SeAMK
SEINÄJOEN AMMATTIKORKEAKOULU
SEINÄJOKI UNIVERSITY OF APPLIED SCIENCES

Valtteri Manninen, Reeta Haaraniemi, Jarmo Luoma & Joni Viitala

IKE-hiilijalanjälkilaskurin kehittäjän opas

Ilmastokestävät elintarvikeprosessit -hanke

Opas
Kevät 2023
Kestävät ruokaratkaisut

ISBN 978-952-7515-35-8

Hanke rahoitetaan REACT-EU-välineen määrärahoista osana Euroopan unionin COVID-19-pandemian johdosta toteuttamia toimia.

SISÄLTÖ

Kuva-, kuvio- ja taulukkoluettelo

Käytetyt termit ja lyhenteet

Allokointi	Kohdentaminen.
Biogeeniset päästöt	Vapautuvat päästöt biomassan hajoamisen tai palamisen seurauksena.
CO_2-ekv.	Hiilidioksidiekvivalentti, joka kuvaa ihmisen tuottamien, ilmastoa lämmittävien kasvihuonekaasujen ilmastovaikutusta.
Ehtorakenne	Ohjelman osa, jonka avulla ohjelman suoritusta voidaan muuttaa sen mukaan, täyttyvätkö määrätyt ehdot vai eivät.
Elinkaari	Tuotteen tai palvelun koko käyttöaika sen tuottamisesta käytöstä poistamiseen.
Fossiiliset päästöt	Vapautuvat päästöt uusiutumattomien päästölähteiden käytön seurauksena.
Funktio	Makron tyyppi (tässä asiayhteydessä), joka eroaa Sub-tyyppisestä makrosta siinä, että se voi palauttaa arvon.
Käyttöliittymä	VBE:ssä laadittu ja käyttötarkoitukseen sopivaksi räätälöity ikkuna, jonka avulla käyttäjä voi syöttää ohjelmalle tietoja.
LCA	Life Cycle Assessment eli elinkaariarviointi.
Makro	Tietokoneohjelma, jonka avulla suoritetaan määrättyjä toimintoja käyttäjän puolesta.
Moduuli	VBA-koodia sisältävä yksikkö VBE:ssä.
Muuttuja	Muuttujiin voidaan varastoida erityyppistä dataa sovelluksen käyttämään muistiin. Muuttujiin voidaan varastoida esimerkiksi kokonaislukuja (Integer), Totuusarvoja (Boolean) ja tekstiä (String).

PEF Product Environmental Footprint eli tuotteen ympäristöjalanjälki. Perustuu elinkaarenarviointimenetelmään, joka sisältää 16 tärkeintä ympäristövaikutusluokkaa.

PEFCR Tuoteryhmäsäännöt, joilla voidaan tehdä tietyille tuotteille laskennan vertailua tuotekilpailutustilanteessa.

Primaaridata Ensisijainen tieto esim. prosessin sähkönkulutuksesta.

Päästökerroin Käytetään vaikutusarviointilaskennassa päästön massana. Esimerkiksi elintarvikkeiden tapauksessa käytetään yleensä päästön massaa kilogramma (kg) suhteessa toiminnalliseen yksikköön kilogrammaan (kg) tuotetta, eli paljonko päästöjä syntyy kilogrammoina kilogrammaa tuotetta kohden. Päästökerrointa valitessa on tärkeää tutustua mitä lähtötietoja päästökertoimen laskentaan on sisällytetty.

Sekundaaridata Toissijainen tieto esim. kirjallisuuslähteestä tai yleisestä tietokannasta.

SFS-EN ISO 14040: -06 Käsittelee ympäristöasioiden hallintaa, elinkaariarvioinnin (LCA) periaatteita ja pääpiirteitä. Ohjaa elinkaariarvioinnin laskennan kriteerejä ja määrittää elinkaareen vaikuttavia tekijöitä.

SFS-EN ISO 14044: -06 Käsittelee ympäristöasioiden hallintaa, elinkaariarvioinnin (LCA) vaatimuksia ja suuntaviivoja.

SFS-EN ISO 14067: -18 Sisältää kasvihuonekaasujen määritelmät, sekä hiilijalanjäljen laskentaa koskevia vaatimuksia, määrittelyjä ja ohjeita.

Silmukkarakenne Ohjelman osa, jonka avulla tiettyä koodin osaa voidaan toistaa useita kertoja määrättyjen ehtojen mukaisesti.

VBA Visual Basic for Applications. Microsoftin sovelluksissa käytettävä ohjelmointikieli makrojen luomiseen.

VBE Visual Basic Editor. Koodieditori VBA:lle.

Vihreä siirtymä	Euroopan Unionin tavoite ilmastoneutraalisuudesta vuoteen 2050 mennessä kaikkien sen jäsenmaiden toteuttamalla ilmastopolitiikalla.
Ympäristöjalanjälki	Määrittelee tuotteen elinkaaren tärkeimmät vaikutusluokat ympäristöön. Ympäristöjalanjälki pystytään määrittämään laskemalla käyttäen apuna PEF-ohjeistusta.

1 HANKKEEN KUVAUS JA TAUSTAT

Hankkeen nimi: Ilmastokestävät Elintarvikeprosessit (IKE)

Toteutusaika: 1.12.2021–31.8.2023

Rahoitus: Euroopan aluekehitysrahasto (EAKR) 80 % ja Seinäjoen Ammattikorkeakoulu Oy 20 %

Hanke rahoitetaan REACT-EU-välineen määrärahoista osana Euroopan unionin COVID-19-pandemian johdosta toteuttamia toimia.

Budjetti: 243 440 €

Toteuttaja: Seinäjoen Ammattikorkeakoulu Oy

Hankkeen taustatietoa: Etelä-Pohjanmaalla on Suomen ruokamaakuntana kova paine ilmastokestävän tuotannon kehittämisessä. Euroopan unionin vihreä siirtymä ja Suomen valtion ilmastotavoitteet hiilineutraalisuudesta vuoteen 2035 mennessä vauhdittavat tätä. Lisäksi kaupalliset toimijat ja kuluttajat ovat entistä ympäristötietoisempia ja haluavat tietoa tuotteiden hiilijalanjäljestä. Pienillä ja keskisuurilla yrityksillä on käytettävissään vähemmän resursseja tuotteidensa ympäristöjalanjäljen laskentaan kuin suuryrityksillä, eikä elintarvikeprosessien ympäristövaikutuksia tai kokonaistehokkuutta ole juurikaan tarkasteltu Etelä-Pohjanmaan pk-yrityksissä.

Seinäjoen Ammattikorkeakoulun Ilmastokestävät elintarvikeprosessit -hankkeen päätavoitteena on ollut luoda Etelä-Pohjanmaan alueen pk-yritysten sekä tutkimus-, kehittämis- ja innovaatiotoimijoiden (TKI) käyttöön laskentatyökalu, jolla elintarviketuotteen ympäristöjalanjälki voidaan laskea. Hankkeen alun tavoitteisiin kuuluivat erilaisten arviointimenetelmien ja prosessien tunnistaminen ympäristö- ja elinkaarivaikutusten arviointiin. Luotua IKE-hiilijalanjälkilaskuria mallinnettiin ja kehitettiin Frami Food Lab -ympäristössä sekä tämän jälkeen testattiin reaaliprosesseilla. Tuloksena saatua hiilijalanjäljen laskentamallia voivat hyödyntää kaikki pk-yritykset ja TKI-toimijat oman toimintansa kehittämisessä. Sen avulla pystytään

samalla tukemaan elintarvikealan pk-yritysten vihreää siirtymää kohti hiilineutraalia elintarviketuotantoa vuoteen 2035 mennessä. Yritykset saavat avoimen IKE-hiilijalanjälkilaskurin avulla mahdollisuuden oman tuotteensa hiilijalanjäljen arviointiin sekä työkalun sisäiseen kehitystyöhönsä kohti ympäristöystävällisempää tuotantoa. Lisäksi kehitettyä IKE-hiilijalanjälkilaskuria on mahdollista edelleen jatkokehittää, päivittää sekä hyödyntää koulutuksessa ja opetuksessa.

2 JOHDANTO

Tämän oppaan tarkoituksena on auttaa IKE-hiilijalanjälkilaskurin mahdollisessa tulevassa kehityksessä. Oppaan laatiminen koettiin tarpeelliseksi, sillä laskurin tietokantoja sekä mahdollisesti laskentaperiaatteita tulee päivittää sitä mukaa, kun uusia ja päivitettyjä päästökertoimia tulee saataville tai hiilijalanjäljen laskentaan liittyvät standardit ja ohjeistukset päivittyvät. Laskuria ohjaavien VBA-makrojen, funktioiden ja käyttöliittymien monimutkaisuuden vuoksi koettiin myös, että näiden ohjelmien toimintaa olisi hyvä avata oppaassa tuleville kehittäjille. Laskurissa on myös havaittu useita potentiaalisia kehityskohteita, joita tässä oppaassa kuvaillaan ja pohditaan. Hankkeen päättymisen vuoksi kaikkia tässä oppaassa esitettyjä kehitystoimenpiteitä ei voitu suorittaa, joten halusimme antaa tulevalle jatkokehitykselle mahdollisimman hyvät lähtökohdat tämän oppaan avulla.

Opas on suunnattu ensisijaisesti henkilöille, joilla on valmiiksi osaamista VBA-ohjelmoinnista. Kuitenkin esimerkiksi päästökerrointietokannan päivittämiseen riittävät normaalit MS Excelin käyttämiseen riittävät taidot, jolloin ohjelmointitaidot eivät ole välttämättömiä. Lisäksi useat laskentaperiaatteiden päivittämiseen liittyvät asiat eivät vaadi välttämättä syvällistä ymmärrystä ohjelmoinnista, sillä jo perusasioiden hallitseminen riittää tämän oppaan avustuksella monien laskurin osioiden muokkaamiseen. Tässä oppaassa ei ohjeisteta tai neuvota VBA:n käyttämiseen tai makrojen luomiseen yleisesti, vaan näistä puhutaan IKE-hiilijalanjälkilaskurin kontekstissa.

Tässä oppaassa selitetään laskurin toimintaa käyttäjän näkökulmasta ainoastaan silloin, kun se on ohjelman toiminnan selittämiseksi välttämätöntä. Tämän oppaan ohella suositellaan luettavaksi ”IKE-hiilijalanjälkilaskurin käyttöopas” -opasta, jossa laskurin toimintaa ja laskentaperiaatteita selitetään paremmin käyttäjän näkökulmasta. Tässä oppaassa tullaan myös viittaamaan tähän oppaaseen.

Opasta lukiessa ja ohjelmaa tutkiessa harjaantuneempi VBA-ohjelmoija (tai kuka vain ohjelmoija) saattaa huomata, että kaikki makrot eivät välttämättä ole koodattu kaikkien yleisten hyvien ohjelmointikäytänteiden mukaisesti, yhtenäisellä tavalla toisiinsa nähden tai ylipäätään kovinkaan taitavasti. Tämä johtuu puhtaasti siitä, että tämän työkalun kehitystyön

aloitushetkillä laskurin ohjelmoinnista vastannut henkilö ei ollut eläessään avannutkaan Visual Basic Editoria. Tästä syystä varsinkin ohjelman vanhimmat makrot eroavat niin tyylissään kuin laadussaan uusimmista, koska näitä koodatessa samanaikaisesti opiskeltiin vielä VBA-ohjelmoinnin perusteita. Ohjelmaan tutustuessa kehityksen kuitenkin näkee ja seuraavasta VBA-projektista tulee varmasti siistimpi ja yhtenäisempi.

Tässä oppaassa on ensin kappaleessa 3 esitelty lyhyesti IKE-hiilijalanjälkilaskurin perusasiat niihin kuitenkaan syvällisemmin perehtymättä. Tässä kappaleessa myös selitetään, miten Excel-työkirjasta saa makrot kytkettyä päälle, kehittäjätyökalut näkyviin ja miten koodieditorin väriteeman saa muokattua tämän oppaan esimerkkejä vastaavaksi. Tämän jälkeen kappaleessa 4 käydään hyvin yksityiskohtaisesti läpi IKE-hiilijalanjälkilaskurin ohjelman toimintaperiaatteet ja rakenteet. Lopuksi kappaleessa 5 pohditaan laajasti ohjelman kehittämiskohteita ja mahdollisia uusia ominaisuuksia.

3 IKE-HIILIJALANJÄLKILASKURI

IKE-Hiilijalanjälkilaskuri on Microsoft Excel -pohjainen laskentamenetelmä tuotteen hiilijalanjäljen laskentaan. Laskentaperiaatteen suunnittelussa on pyritty noudattamaan ISO 14040 (Ympäristöasioiden hallinta. Elinkaariarviointi. Periaatteet ja pääpiirteet), ISO 14044 (Ympäristöasioiden hallinta. Elinkaariarviointi. Vaatimukset ja suuntaviivoja) ja ISO 14067 (Kasvihuonekaasut. Tuotteiden hiilijalanjälki. Hiilijalanjäljen laskemista koskevat vaatimukset ja ohjeet) -standardeja sekä Euroopan komission suositusta: ”Ympäristöjalanjälkeä koskevien menetelmien käyttämisestä tuotteiden ja organisaatioiden elinkaaren ympäristötehokkuuden mittaamiseen ja siitä tiedottamiseen”, josta myöhemmin tässä oppaassa käytetään lyhyempää nimitystä ”PEF-ohjeistus”.

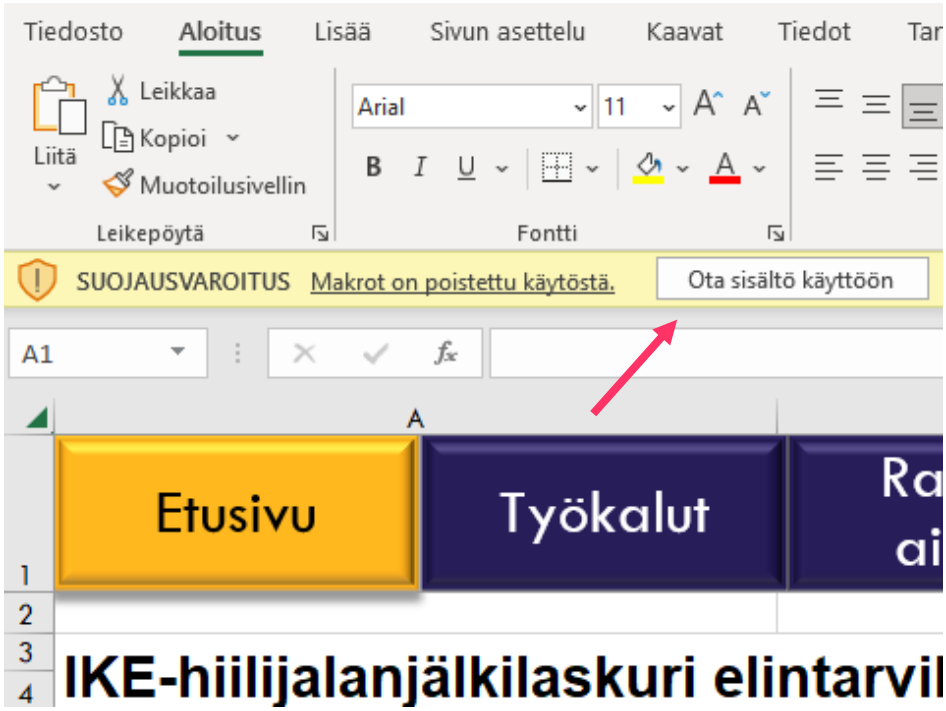
Koska IKE-hiilijalanjälkilaskuri ei ole täysin standardien mukainen, se on tarkoitettu ainoastaan yritysten omaan käyttöön kehitystyöhön. Laskentatuloksia ei siis voi sellaisenaan hyödyntää markkinoinnissa. Menetelmä on kuitenkin haluttu tehdä mahdollisimman läpinäkyväksi eli päästökerrointietokannasta löytyvät kootusti laskennassa käytettävät tiedot. Seinäjoen Ammattikorkeakoulu Oy ei kuitenkaan vastaa tällä työkalulla saaduista tuloksista ja niiden käytöstä.

Laskentatyökalussa on käytetty paljon automatisointia Excelin VBA:n (Visual Basic for Application) avulla, jotta laskennasta voitiin tehdä monipuolisempi ja tarkempi kuitenkin siten, että laskurin käyttäjäystävällisyys ei kärsisi kohtuuttomasti. Tietyissä IKE-hiilijalanjälkilaskurin ominaisuuksissa on hyödynnetty myös Microsoft Wordia ja sen automatisointia esimerkiksi yhteenvetoraportin tulostuksessa ja IKE-hiilijalanjälkilaskurin laskentaoppaan aukaisemisessa. Varsinainen laskenta on laskurissa jaettu viidelle laskentavälilehdelle: raaka-aineet, pakkausmateriaalit, jätteet ja sivuvirrat, energia ja logistiikka. Ennen laskentavälilehtiä laskurista löytyy etusivu- ja työkalut-välilehdet. Laskentavälilehtien jälkeen välilehdet yhteenvedolle, tasetarkastelulle sekä päästökertoimille. Lisäksi laskurissa on piilotettuna datavälilehti, joka sisältää ohjelman kannalta välttämättömiä tietoja, mutta eivät ole käyttäjän näkökulmasta oleellisia.

IKE-hiilijalanjälkilaskuri on tallennettava tietokoneen kovalevylle samaan kansioon yhdessä IKE-hiilijalanjälkilaskurin käyttöoppaan sekä tulostettavan raportointipohjan kanssa, että nämä saadaan toimimaan yhdessä. Ohjeistus- ja raportointiominaisuudet eivät siis toimi, jos laskuri on tallennettu verkkosijaintiin, kuten Microsoft OneDriveen. Itse hiilijalanjälkilaskurin sisältävän Excel-tiedoston tiedostonimellä ei ole merkitystä, mutta laskentaopas tulee olla nimetty tarkasti tiedostonimellä ”IKE-hiilijalanjälkilaskurin käyttöopas.docx” ja raporttipohjan ”IKE-hiilijalanjälkilaskurin raportin pohja.docx”. Mikäli tiedostojen nimissä on yhdenkin merkin virhe, ohjelman tiedostoja käyttävät ominaisuudet eivät toimi.

3.1.1 Makrojen ottaminen käyttöön

Ennen IKE-hiilijalanjälkilaskurin käytön aloittamista on hyväksyttävä laskennassa käytettävien VBA-makrojen käyttö valitsemalla kuvan 1 mukaisesti yläreunaan aukeava ”Ota sisältö käyttöön”, sillä arviointia ei voida tehdä ilman niiden toiminnallisuutta. Käyttöönotto voi vaatia järjestelmänvalvojan käyttöoikeuksia tai vaihtoehtoisesti käyttöoikeuksien sallimista yrityksen tietoturvasta vastaavan toimijan puolesta.



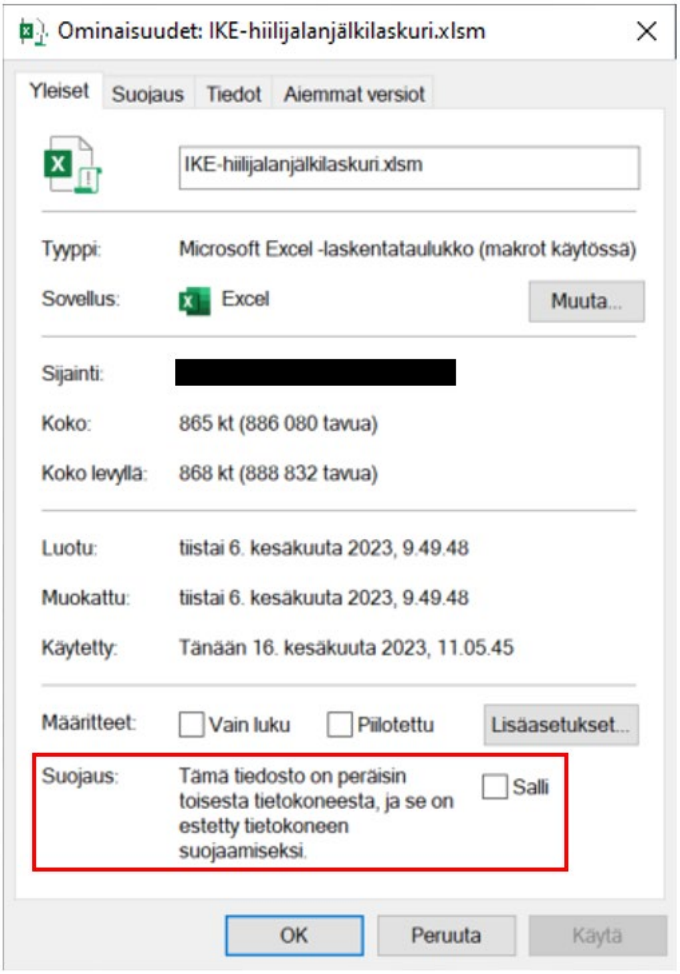
Kuva 1. Makrojen sisällön käyttöönotto.

Jos makroja ei tästä huolimatta saada käyttöön, voidaan käyttää erilaisia Microsoftin tarjoamia ohjeita makrojen käyttöönottamisesta:

1. **tapa** (kuva 2)**:**

https://support.microsoft.com/fi-fi/topic/mahdollisesti-vaarallinen-makro-on-estetty-0952faa0-37e7-4316-b61d-5b5ed6024216

- Avaa tietokoneelta kansio, johon IKE-hiilijalanjälkilaskuri on tallennettu
- Paina tiedoston kohdalla hiiren kakkospainiketta
- Valitse **Ominaisuudet**
- Valitse **Yleiset** -välilehti
- Valitse ikkunan alareunasta **Salli**-valintaruutu
- Valitse **OK**.

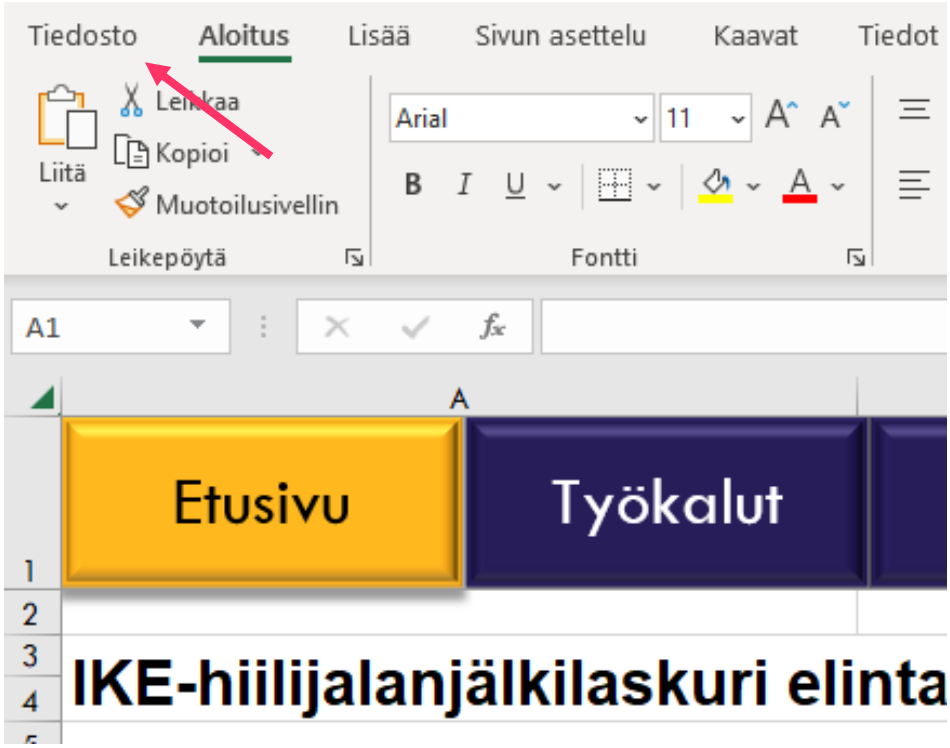


Kuva 2. Tiedoston ominaisuudet.

2. **tapa:**

https://support.microsoft.com/fi-fi/office/makrojen-ottaminen-k%C3%A4ytt%C3%B6%C3%B6n-tai-poistaminen-k%C3%A4yt%C3%B6st%C3%A4-microsoft-365-tiedostoissa-12b036fd-d140-4e74-b45e-16fed1a7e5c6

- Valitse **Tiedosto-välilehti** (kuva 3)

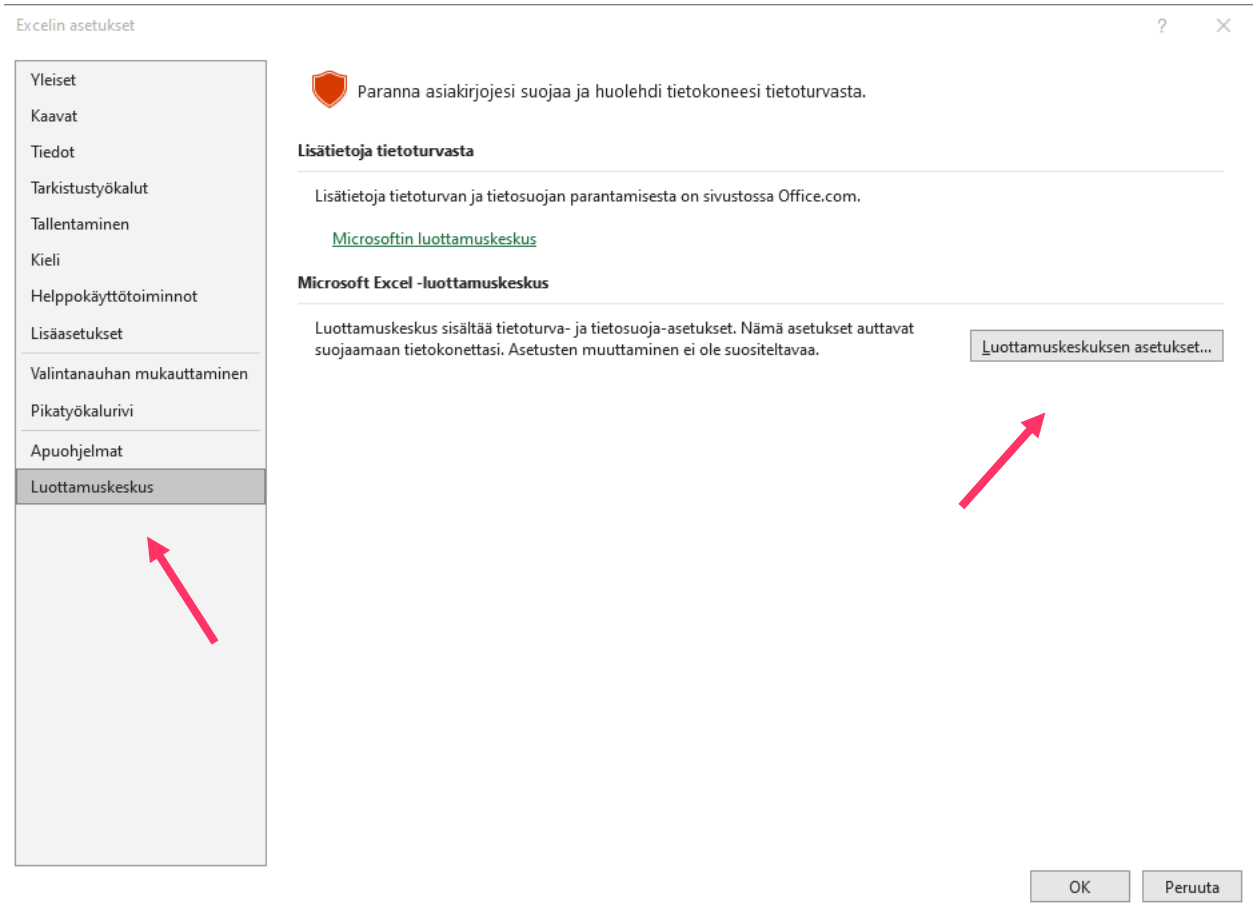


Kuva 3. Tiedosto-välilehti.

– Valitse vasemmasta alakulmasta **Asetukset** (kuva 4)

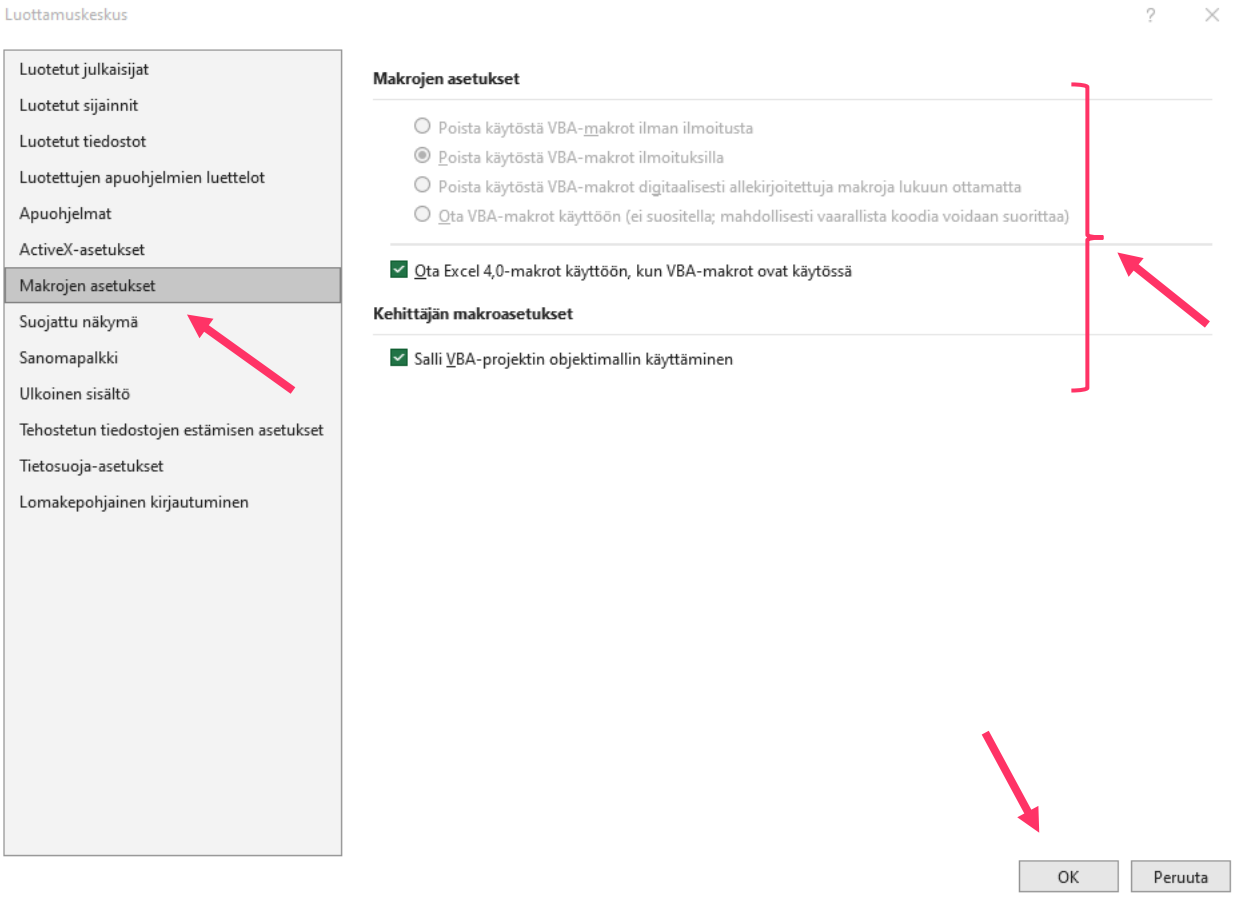
Kuva 4. Asetukset-valikko vasemmassa alakulmassa.

– Valitse **Luottamuskeskus** (kuva 5)
– Valitse **Luottamuskeskuksen asetukset** (kuva 5)



Kuva 5. Luottamuskeskuksen asetusvalikko.

- Valitse **Makrojen asetukset** (kuva 6)
- Valitse tarvittavat hyväksyvät makrojen käyttöönottoasetukset (vaihtelevat eri Excel-versioiden mukaan) (kuva 6)
- Valitse **OK** (kuva 6)



Kuva 6. Makrojen asetusvalikko.

– Tämän jälkeen makrojen käyttö on sallittavissa.

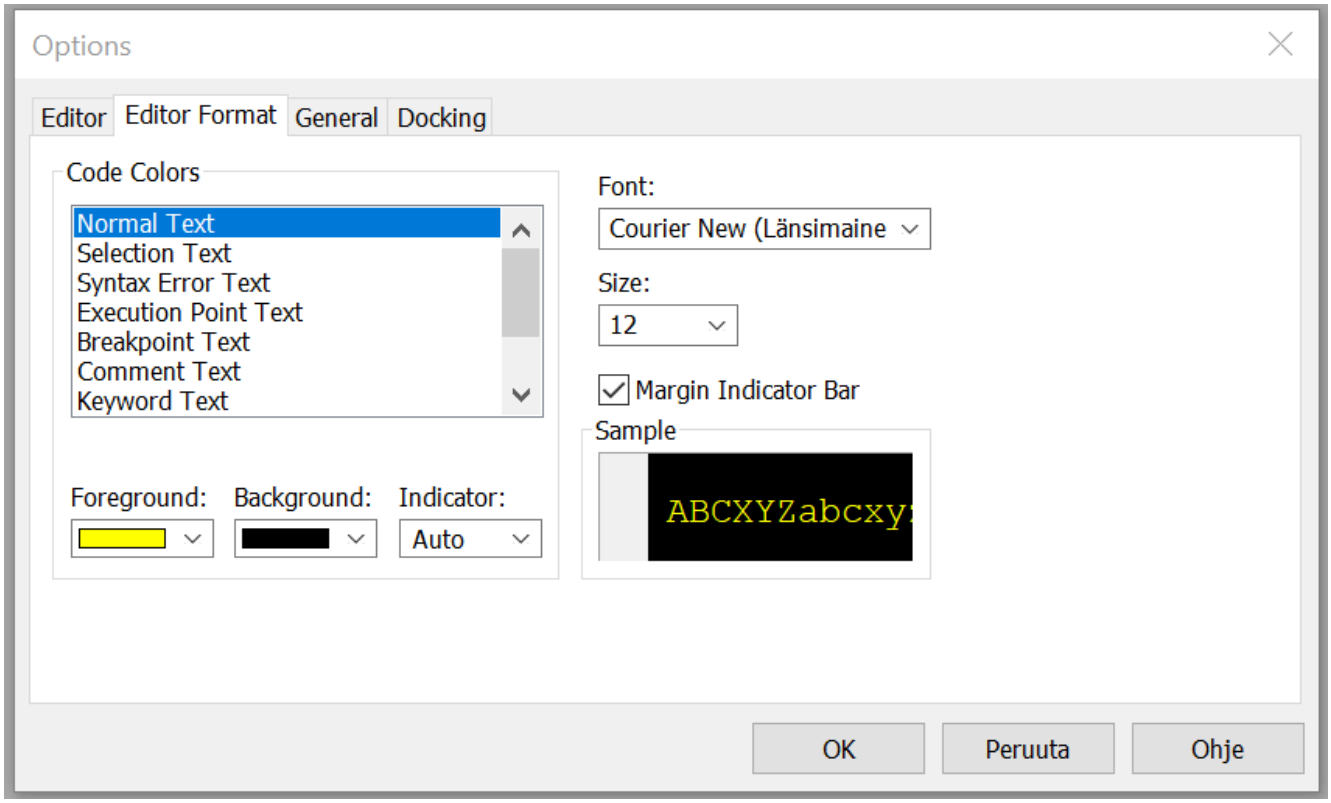
Seinäjoen Ammattikorkeakoulu Oy ei vastaa mahdollisten muiden vahingollisia makroja sisältävien tiedostojen käytöstä aiheutuvista vahingoista.

3.1.2 Visual Basic Editorin ”Dark Mode”

Harmillisesti VBE:ssä (Visual Basic Editorissa) ei varsinaisesti ole olemassa erivärisiä teemoja, joilla koodieditorin taustan tai tekstien värejä saisi näppärästi kerralla muutettua (My Excel Genious, i.a.). Hätä ei ole kuitenkaan tämän näköinen, sillä VBE:n värimaailmaa on mahdollista muokata asetusten kautta. Tässä oppaassa esitellyt koodit näkyvät VBE:n oletusasetuksista poikkeavissa väreissä, joten mikäli tähän oppaaseen syventyy tarkemmin,

suositellaan asetuksia muokattavan samanlaisiksi koodin lukemisen helpottamiseksi. Alla esitellyt asetukset ovat sivustolta My Excel Genius blogitekstistä Coding VBA on the Dark Side.

VBE:n värimaailman muuttaminen alkaa Visual Basicin avaamisesta Excelin Kehitystyökalut lehdeltä tai painamalla Alt+F11. Tätä ennen tulee Kehitystyökalut asetukset olla laitettuna päälle asetuksista Asetukset→Valintanauhan mukauttaminen→Päävälilehdet. Alt+F11 komento ei vaadi erikseen kehitystyökalujen lisäämistä valintanauhaan. Kun Visual Basic on avattu, ylävalikosta valitaan Tools→Options→Editor Format, josta aukeaa kuvan 7 kaltainen ikkuna.



Kuva 7. Visual Basic Editorin ulkoasun muokkaamiseen liittyvät asetukset

Mikäli haluaa käyttää vastaavia väriasetuksia, kuten tässä oppaassa esitellyissä esimerkeissä, tulee Code Colors valikkoon valita taulukon 1 mukaiset asetukset. Fonttina on käytetty kuvassa 7 näkyvää Courier New -fonttia. Toki voit käyttää myös itse haluamiasi asetuksia.

Taulukko *1*. VBE:n eräs vaihtoehto tumman teeman asetuksiksi.

Code Colors	Foreground	Background	Indicator
Normal Text	Yellow	Black	Auto
Selection Text	Auto	Auto	Auto
Syntax Error Text	Red	Black	Auto
Execution Point Text	Black	Yellow	Yellow
Breakpoint Text	White	Cyan (Light Blue)	Cyan (Light Blue)
Comment Text	Bright Green	Black	Auto
Keyword Text	Bright Blue	Black	Auto
Identifier Text	White	Black	Auto
Bookmark Text	Auto	Auto	Bright Blue
Call Return Text	Auto	Auto	Bright Blue

Mikäli nämä asetukset eivät miellytä ja haluat vaihtaa takaisin VBE:n oletusasetuksille, taulukon 2 asetuksilla tämä onnistuu.

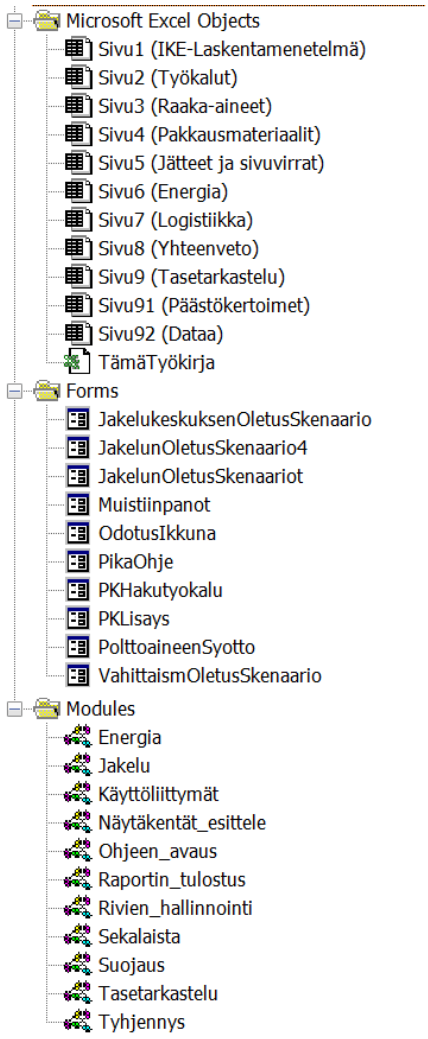
Taulukko *2.* VBE:n väriteeman oletusasetukset

Code Colors	Foreground	Background	Indicator
Normal Text	Black	Auto	Auto
Selection Text	Auto	Auto	Auto
Syntax Error Text	Red	Auto	Auto
Execution Point	Auto	Yellow	Yellow
Breakpoint Text	White	Dark red	Dark red
Comment Text	Dark green	Auto	Auto
Keyword Text	Dark Blue	Auto	Auto
Identifier Text	Auto	Auto	Auto
Bookmark Text	Auto	Auto	Bright blue
Call Return Text	Auto	Auto	Bright green

4 OHJELMA

IKE-hiilijalanjälkilaskuri hyödyntää toimiakseen runsaasti erilaisia makroja, funktioita ja käyttöliittymiä. Näitä käytetään esimerkiksi taulukoiden rivien lisäämiseen ja poistamiseen, suojausten hallintaan, muistiinpanojen kirjoittamiseen, laskentaperiaatteiden valintaan, käyttöoppaan avaamiseen ja moneen muuhun. Laskuria ohjaavaa koodia on verrattain runsaasti, sillä koodia on kirjoitettu Microsoft Excel Objekteihin (välilehdet ja työkirja), käyttöliittymiin sekä omiin moduuleihinsa. Yksittäisiä koodia sisältäviä ”projekteja” on laskurin Excel-tiedostossa yhteensä 32, jotka sisältävät yhteensä tyhjät rivit ja kommentit mukaan luettuna hieman alle 7500 riviä koodia. Kuvassa 8 on kuvakaappaus laajennetusta projekti- ja moduulikansiosta, josta on nähtävissä kaikki laskurissa käytetyt Microsoft Excel Objektit, käyttöliittymät ja moduulit.

Yleisesti laskuria varten kirjoitetussa koodissa on hyödynnetty runsaasti nimettyjä alueita, jolloin haluttuihin soluihin on ollut helppoa viitata, vaikka esimerkiksi taulukoiden sijainnit muuttuisivatkin. Laskurin kehityksen alkutaipaleella ei myöskään ollut täyttä varmuutta laskurin lopullisesta ulkoasusta, joten monet perustoiminnot koodattiin siten, että ne toimivat solujen sijaintien vaihtelusta huolimatta. Esimerkiksi rivien lisäämiseen ja poistamiseen liittyvät makrot tai laskentakaavojen lisäämiset makrojen avulla voivat tästä syystä vaikuttaa tarpeettoman monimutkaisilta.



Kuva 8. IKE-hiilijalanjälkilaskurin ohjelman rakenne.

Microsoft Excel Objektit sisältävät välilehtien ja työkirjan aktivointiin, deaktivointiin, muuttamiseen ja sulkemiseen liittyviä toimintoja. Tyypillisesti nämä liittyvät sivujen asetuksiin, suojauksiin, tai tiettyjen solujen muuttumisesta käynnistettäviin ohjelmiin. Kuvassa 8 näkyvät Microsoft Excel Objektit ovat jaoteltu välilehtien nimien mukaisesti ja listan viimeisenä löytyvä Tämä Työkirja -projekti koskee koko työkirjalle tapahtuvia toimintoja.

Käyttöliittymät eli Forms-projektit sisältävät itse luodut käyttöliittymät ja näiden käyttöliittymien taustalla vaikuttavaa koodia. Näitä käytetään oletusskenaarioiden valinnassa, muistiinpanotyökalussa, päästökertoimien hakutyökalussa, päästökertoimien lisäystyökalussa, pikaohjeen aukaisemisessa sekä polttoaineen syöttämisessä ajoneuvolistaan.

Lopuksi moduulit eli Modules sisältävät kaiken muun koodin, mitä laskurin käyttämiseen tarvitaan. Nämä moduulit sisältävät myös paljon makroja ja funktioita, joita hyödynnetään kutsumalla niitä muista projekteista. Esimerkiksi suojauksien sammuttamiseen ja käynnistämiseen liittyviä makroja kutsutaan lähes kaikista Microsoft Excel Objekteista ja käyttöliittymistä. Moduulit on jaoteltu suurempien kokonaisuuksien perusteella, jotka sisältävät nimettyyn aihepiiriin liittyviä ohjelmia.

Huomioithan, että laskurin koodia hieman muokattiin sieltä täältä vielä tämän oppaan kirjoittamisen jälkeen, mutta merkittäviä muutoksia ei tehty.

4.1 Microsoft Excel Objects

Microsoft Excel Objektit ovat makroja sisältäviä projekteja, jotka aktivoituvat tietyille välilehdille tai koko työkirjalle tapahtuvista toiminnoista. Tässä tapauksessa käytettyjä makroja olivat välilehden aktivoimisesta käynnistyvä Worksheet_Activate, välilehden deaktivoimisesta käynnistyvä Worksheet_Deactivate, välilehden solujen muuttamisesta aktivoituva Worksheet_Change(ByVal Target As Range), työkirjan aktivoimisesta käynnistyvä Workbook_Activate ja työkirjan deaktivoimisesta käynnistyvä Workbook_Deactivate.

4.1.1 Worksheet_Activate

Lähes kaikille välilehdille kohdistetuille projekteille on kirjoitettu koodi, joka käynnistyy välilehden aktivoimisesta. Näillä säädetään yleensä ainakin valikon leveyttä, zoomausta ja säädetään sarakkeiden leveydet AutoFit-toiminnolla. Kuvassa 9 on kuvakaappauksena esimerkki Raaka-aineet-välilehden aukaisemisesta aktivoituvasta makrosta.

```
Option Explicit

Private Sub Worksheet_Activate()

    Call VapautaSivu("Raaka-aineet")

    'Säädetään valikkopalkin leveys ja korkeus sekä ensimmäisen rivin leveys. Tämä tehdään siksi,
    'että jostain syystä valikko muuttaa itsekseen kokoaan.

    ActiveSheet.Shapes("Valikko3").Width = 1500
    ActiveSheet.Shapes("Valikko3").Height = 60
    Rows("1:1").RowHeight = 60

    Range("A1", "F1").EntireColumn.AutoFit
    Columns("A").ColumnWidth = 65

    'Zoomataan näytön kokoon sopivaksi.
    Call zoom
    ActiveWindow.ScrollRow = 1

    'Palautetaan asetukset
    Call KehittajaAsetukset
    Call SuojaaSivu("Raaka-aineet")

End Sub
```

Kuva 9. Raaka-aine-välilehden avaamisesta käynnistyvä makro.

Kuvan 9 makro avaa ensin välilehden suojauksen, tämän jälkeen säätää valikko-objektin leveyden ja korkeuden, säätä ensimmäisen rivin korkeuden, säätää sarakkeiden leveydet, säätää näytön zoomauksen käytetyn näytön koon mukaisesti, kelaa sivun ylälaitaan, palauttaa mahdolliset kehittäjäasetukset ja lopuksi palauttaa suojausasetukset. Kuten kuvassa 9 esitetystä koodista voi huomata, makro kutsuu ulkopuolisia makroja VapautaSivut, zoom, KehittäjäAsetukset ja SuojaaSivu. Nämä makrot sijaitsevat moduuleissa ja niihin tutustutaan tässä oppaassa tarkemmin myöhemmin. Tämän kaltainen ohjelma löytyy kaikilta välilehtiprojekteilta Dataa-välilehteä lukuun ottamatta.

Näiden perustoimintojen lisäksi Yhteenveto ja Tasetarkastelu -välilehdillä välilehden aktivointi käynnistää muitakin toimintoja. Yhteenvetosivulla aktivointi käynnistää sivuvirtojen listaan liittyviä toimenpiteitä, joiden tarkoitus on piilottaa tai päivittää sivuvirtoihin liittyviä välilehden osia. Ohjelma ensiksi tarkistaa, onko laskuriin syötetty yhtään sivuvirtaa. Mikäli ei ole, ohjelma piilottaa yhteenvetosivulta sivuvirtojen taulukon, kuvaajat ja painikkeet. Jos sivuvirtoja taas on vähintään yksi, taulukot, kuvaajat ja painikkeet muutetaan näkyviksi ja sivuvirtojen taulukkoon päivitetään ajantasaiset tiedot. Tasetarkastelu-välilehden avaamisesta aktivoituva makro luonnollisesti päivittää tasetarkastelukaavioon ajantasaiset tiedot etusivulta ja laskentavälilehdiltä.

4.1.2 Worksheet_Deactivate

Välilehden sulkemisesta aktivoituva makro löytyy ainoastaan Päästökertoimet-välilehden projektista. Kuvassa 10 on kuvakaappaus kyseisestä makrosta.

```
Private Sub Worksheet_Deactivate()

    Application.EnableEvents = False
    Application.ScreenUpdating = False

    Call VapautaSivu("Logistiikka")
    Call VapautaSivu("Energia")

    Call KorjaaPudotusvalikko("Energianlahde", "Energia", "PKSahko")
    Call KorjaaPudotusvalikko("MuutLaitteetPolttoaineet", "Energia", "PKPolttoaineet")
    Call KorjaaPudotusvalikko("AjoneuvonTyyppi", "Logistiikka", "PKAjoneuvot")
    Call KorjaaPudotusvalikko("AjoneuvotPolttoaineet", "Logistiikka", "PKPolttoaineet")
    Call KorjaaPudotusvalikko("KylmaKaasut", "Logistiikka", "PKKylmaainekaasut")

    Call SuojaaSivu("Logistiikka")
    Call SuojaaSivu("Energia")

    Application.EnableEvents = True
    Application.ScreenUpdating = True

End Sub
```

Kuva 10. Päästökertoimet-välilehden sulkeutumisesta käynnistyvä makro.

Kuvassa 10 esitetyn makron tarkoitus on korjata laskurin eri välilehdillä sijaitsevien pudotusvalikoiden sisältöjä. Kyseisten korjattavien pudotusvalikoiden tiedot päivittyvät päästökertoimien tietokannan otsikoiden perusteella, joten ilman tätä makroa valikoiden tiedot vanhentuisivat, mikäli päästökertoimia lisättäisiin tai otsikoita muutettaisiin. Makro siis automaattisesti päästökerroinvälilehden tietojen perusteella päivittää tietokannan mukaiseksi sellaiset pudotusvalikot, jotka muutoin vioittuisivat tällaisista muutoksista.

4.1.3 Worksheet_Change

Työkalut, Jätteet ja sivuvirrat, Energia, Logistiikka ja Yhteenveto -välilehdille kohdistuvissa projektimoduuleissa on käytetty Worksheet_Change(ByVal Target As Range) tyyppistä makroa. Tämän makro aktivoituu, mikäli minkä tahansa välilehden solua muutetaan ja saa Range- tyyppisen muuttujan nimellä Target. Tämä Target-muuttuja vastaa aluetta, jonka muuttuminen aktivoi makron.

Worksheet_Change-makroja käytetään pääasiassa niissä tapauksissa, kun halutaan muuttaa esimerkiksi jonkin toisen solun arvoa, kaavaa tai väriä toisen solun muutosten perusteella. Kuvassa 11 on kuvakaappaus Jätteet ja sivuvirrat-välilehden toimintaan vaikuttavasta Worksheet_Change-makrosta.

```
Private Sub Worksheet_Change(ByVal Target As Range)

    Call VapautaSivu("Jätteet ja sivuvirrat")

    Application.ActiveSheet.Unprotect
    If Range("JatJatteetjaSivuvirrat").row < Target.row And Range("JatteetYhteensa").row > Target.row And Target.Count = 1 Then
       If Target.Column = Range("SivVaiJat").Column Then
           If Target.Value = "Sivuvirta" Then
               Target.Offset(0, 2).Interior.Color = xlNone
               Call PoistaMuokkaus(Target.Offset(0, 2), "Jätteet ja sivuvirrat")
           Else:
               Target.Offset(0, 2).Interior.Color = RGB(251, 173, 24)
               Call LisaaMuokkaus(Target.Offset(0, 2), Target.Offset(0, -2), "Jätteet ja sivuvirrat")

           End If
       End If
    End If

    Call SuojaaSivu("Jätteet ja sivuvirrat")

End Sub
```

Kuva 11. Jätteet ja sivuvirrat -välilehden muuttamisesta käynnistyvä makro.

Yllä olevassa esimerkissä makron tarkoitus on muuttaa päästökerroin-sarakkeen väriä ja muokkaamisoikeuksia (Allow Edit Range) sen mukaan, onko Sivuvirta vai jäte?-sarakkeeseen valittu pudotusvalikosta sivuvirta vai jäte.

Koska sivun muokkaamiseen reagoiva makro aktivoituu minkä tahansa solun muokkaamisesta, tulee ohjelman selvittää, mitä solua on muokattu. Tämä on makrossa järjestetty sisäkkäisillä ehtorakenteilla, joista ulommalla selvitetään, onko muokattu alue jätteiden ja sivuvirtojen taulukon ylä- ja alalaitojen välissä ja seuraavaksi sisemmässä ehtorakenteessa onko sarake sama kuin kohdesolulla. Jos muokattu solu ei ole halutulla alueelle, ohjelma ainoastaan suojaa sivun eikä tee mitään muuta. Mikäli taas kahden ulomman ehtorakenteen ehdot täyttyvät, siirrytään kolmanteen ja sisimmäiseen ehtorakenteeseen, jossa tehdään halutut muutokset sen mukaan, mitä kohdesoluun valittiin. Muokkaamisoikeuksien muuttaminen tehdään omilla makroillaan, joita tämä makro kutsuu.

Hyvin samalla tavalla toimivat:

- Työkalut-välilehdellä tätä makrotyyppiä käytetään yksikkömuuntimen tietojen päivittämiseen

- Energia-välilehdellä päästökertoimien päivittämiseen energiajakaumaan, muiden laitteiden taulukkoon sekä höyryn taulukkoon

- Logistiikka-välilehdellä ajoneuvojen taulukossa laskentaperiaatteiden ja päästökertoimien päivittämiseen ajoneuvotyypin perusteella, varastoinnissa solujen värjäämiseen ja muokkausoikeuksien muuttamiseen sekä varastoinnin energianlähteiden ja kylmäainekaasujen päästökertoimien päivittämiseen

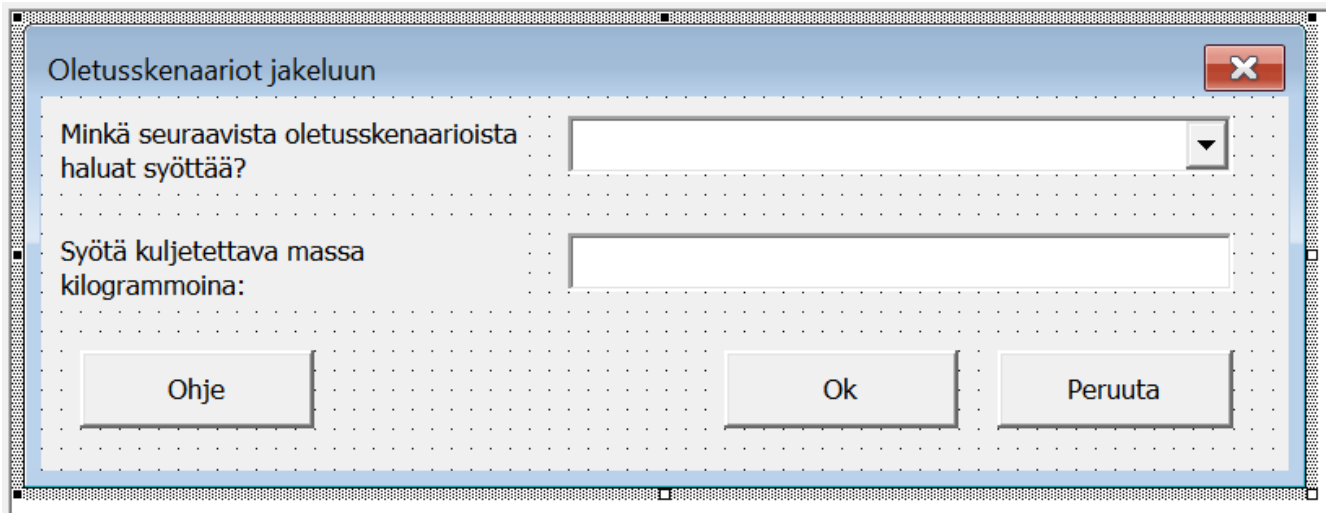
- Yhteenveto-välilehdellä talousallokoinnin ollessa käytössä varmistukseen, että yhteenlasketut prosenttiosuudet ovat alle sata.

4.2 Käyttöliittymät

Laskurissa on luotu muutamia omia käyttöliittymiä, kun käyttäjältä on haluttu kysellä tietoja, joita olisi ollut liian monimutkaista tai vaikeaa kysyä Excelin soluissa tai solujen ominaisuudet eivät salli haluttujen toiminnallisuuksien luomista. Käyttöliittymiä hyödynnettiin erityisesti PEF-ohjeistuksen mukaisten jakelun ja varastoinnin oletusskenaarioiden syöttämisestä.

4.2.1 Oletusskenaariot kuljetuksiin

Seuraavaksi käydään läpi jakelun oletusskenaarion käyttöliittymää ja koodia sen taustalla. Kuvassa 12 on esitetty kuvakaappaus käyttöliittymästä VBE:ssä. Käyttöliittymässä on yksi pudotusvalikko, josta valitaan haluttu oletusskenaario neljästä eri vaihtoehdosta, täytettävä kenttä kuljetettavan massan syöttämiseen sekä kolme painiketta: Ohje, Ok ja Peruuta.



Kuva 12. Oletusskenaariot jakeluun -käyttöliittymä Visual Basic Editorissa.

Kun käyttöliittymä on kuvan 12 mukaisesti avattu VBE:ssä, käyttöliittymän koodiin pääsee kaksoisklikkaamalla jostain kohtaa käyttöliittymää. Käyttöliittymän koodi on jaettu kahdeksaan makroon. Näitä makroja ovat CommandButton1_Click, CommandButton2_Click, CommandButton3_Click, UserForm_Initialize, OletusSkenaario1, OletusSkenaario2, OletusSkenaario3 ja OletusSkenaario4. Kolme ensimmäistä nimensä mukaisesti aktivoituvat käyttöliittymästä löytyvien painikkeiden klikkaamisesta, neljäs on niin sanottu alustusmakro ja neljä viimeistä tuottavat valitun skenaarion tiedot.

Kun käyttöliittymä avataan, ensimmäiseksi aktivoituu UserForm_Initialize-makro. Tämän tehtävä on alustaa tarvittavat lähtötiedot käyttöliittymälle, sillä siinä ei ole automaattisesti valmiina mitään tietoja sen sisältämien ulkoisten objektien lisäksi. Kuvassa 13 on esitetty kuvakaappaus kyseisestä makrosta.

```
Private Sub UserForm_Initialize()

    ' Alkuasetukset. Lisätään pudotusvalikkoon eri kuljetusskenaariot, säädetään ikkunan koko ja sijainti.

    ComboBox1.AddItem "Toimittajalta tehtaalle: Pakkausmateriaalit (ei lasi)"
    ComboBox1.AddItem "Toimittajalta tehtaalle: Pakkausmateriaalit (lasi)"
    ComboBox1.AddItem "Toimittajalta tehtaalle: Muut (mm. raaka-aineet)"
    ComboBox1.AddItem "Tehtaalta loppukäyttäjälle"

    With JakelunOletusSkenaariot
        .Height = 135.5
        .Width = 390
    End With

    Me.Top = Application.Top + (Application.UsableHeight / 2) - (Me.Height / 2)
    Me.Left = Application.Left + (Application.UsableWidth / 2) - (Me.Width / 2)

End Sub
```

Kuva 13. Oletusskenaariot jakeluun -käyttöliittymän alustusmakro.

Kuten kuvasta 13 nähdään, aluksi lisätään käyttöliittymän pudotusvalikkoon halutut vaihtoehdot. (Euroopan komissio, 2021, s. 48–50.) Nämä vaihtoehdot kuvastavat jakelun eri oletusskenaarioita, jotka ovat määritelty PEF-ohjeistuksessa. Tämän jälkeen määritellään käyttöliittymän koko korkeus- ja leveyssuunnissa kuvapisteinä, jotta käyttöliittymä ilmestyisi varmasti oikean kokoisena. Tämän jälkeen määritellään käyttöliittymän sijainti näytöllä, jotta se ilmestyisi aina käytetyn näytön keskelle. Mikäli tätä ei ole määritelty, käyttöliittymä saattaa ilmestyä hankaliin paikkoihin näytöllä tai näytöillä.

Painikkeista aktivoituvat makrot tekevät painikkeiden teksteissä kuvailtuja asioita. Peruuta-painike sulkee käyttöliittymän tekemättä mitään ja Ohje-painike avaa laskurin oppaan oikeasta kohdasta. Oppaan automaattisesta avaamisesta kerrotaan tarkemmin luvussa 4.3.5. Ok-painikkeesta ohjelma lisää syötettyjen tietojen perusteella oikean oletusskenaarion ajoneuvojen taulukkoon. Kuvassa 14 on esitetty kuvakaappaus Ok-painikkeen klikkaamisesta aktivoituvasta koodista.

```
Private Sub CommandButton1_Click()

    ' Syötetään annettujen tietojen perusteella oletusskenaariot kuljetuksista kuljetuslistaan.

    Dim vastaus As String

    Massa = TextBox1.Value

    ' Ensin tarkistus, onko tietoja syötetty ja virheen tarkistus massaan. Jos kaikki on kunnossa,
    ' kutsutaan sopivaa aliohjelmaa valittujen tietojen perusteella.
    If IsNumeric(Massa) = False Or Massa <= 0 Or ComboBox1.Value = "" Then
        If ComboBox1.Value = "" Then
            MsgBox "Et valinnut oletusskenaariota."
            ComboBox1.BackColor = RGB(255, 255, 0)
        Else
            ComboBox1.BackColor = RGB(255, 255, 255)
        End If

        If IsNumeric(Massa) = False Or Massa <= 0 Then
            MsgBox "Massan arvo on virheellinen."
            TextBox1.BackColor = RGB(255, 255, 0)
        Else
            TextBox1.BackColor = RGB(255, 255, 255)
        End If
    Else
        If ComboBox1.Value = "Toimittajalta tehtaalle: Pakkausmateriaalit (ei lasi)" Then
            Call OletusSkenaario1
        ElseIf ComboBox1.Value = "Toimittajalta tehtaalle: Pakkausmateriaalit (lasi)" Then
            Call OletusSkenaario2
        ElseIf ComboBox1.Value = "Toimittajalta tehtaalle: Muut (mm. raaka-aineet)" Then
            Call OletusSkenaario3
        ElseIf ComboBox1.Value = "Tehtaalta loppukäyttäjälle" Then
            Call OletusSkenaario4
        End If
        Massa = TextBox1.Value
        Unload Me
    End If

End Sub
```

Kuva 14. Oletusskenaariot jakeluun -käyttöliittymän Ok-painikkeen makro

Makrossa määritellään aluksi String-tyyppinen muuttuja käyttäjän syöttämää massaa varten ja asetetaan käyttäjän syöttämä arvo siihen. Vaikka massa lähtökohtaisesti on numeerinen arvo, tallennetaan muuttuja aluksi tekstinä esimerkiksi virheentarkistusta varten ja eri kieliasetusten aiheuttamien virheiden välttämiseksi. Tämän jälkeen ohjelma hyödyntää ehtorakennetta virheentarkistukseen ja virheiden puuttuessa sopivan oletusskenaarion valintaan, joka toteutetaan kutsumalla oikean oletusskenaarion toteuttavaa makroa. Virheentarkistuksessa tarkistetaan, onko oletusskenaariota valittu, onko syötetty massan arvo muutettavissa numeeriseksi arvoksi ja onko tämä arvo yli 0.

Kuvassa 15 on kuvakaappaus ensimmäisen oletusskenaarion koodista. (Euroopan komissio, 2021, s.48.) Tällöin ajoneuvojen listaan syötetään PEF-ohjeistuksen mukaisesti tietyt kuljetusvälineet ja matkat. Muut oletusskenaariot toimivat samalla periaatteella viimeistä

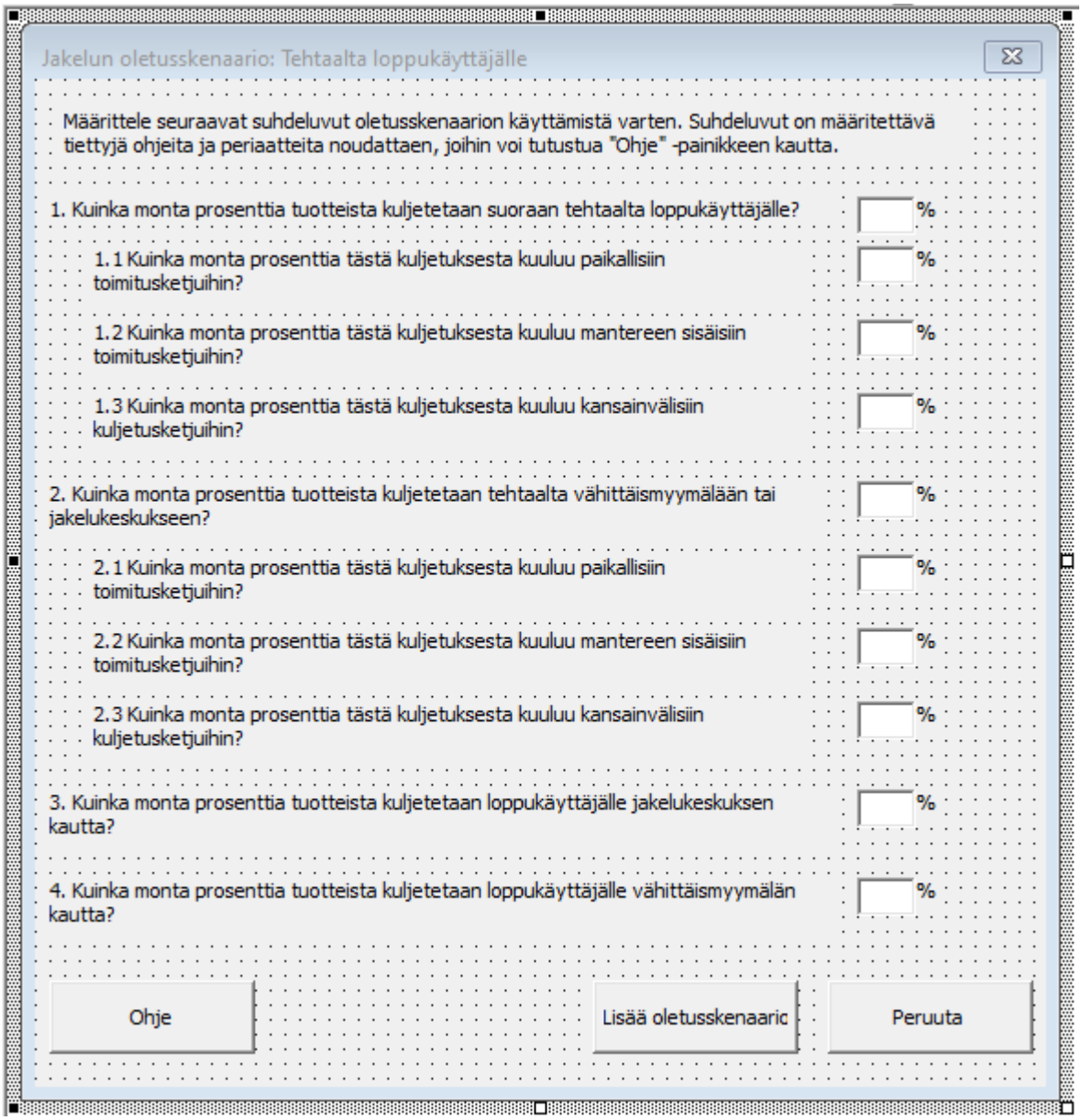
oletusskenaariota lukuun ottamatta, jonka valinta avaa uuden käyttöliittymän uusien tietojen selvittämiseksi.

```
Private Sub OletusSkenaario1()
    '"Toimittajalta tehtaalle: Pakkausmateriaalit (ei lasi)"
    ' 230 km kuorma-autolla (> 32 t, Euro 4)
    ' 280 km junalla (keskimääräinen tavarajuna) ja
    ' 360 km laivalla (proomu).
    'Jos kuorma on massaltaan rajoitettu, on käytettävä 64 prosentin kuormausastetta.

    Call cmdRivinLisaysLog
    Call VarjaaRiviLog

    Application.EnableEvents = False

    With Range("LogistiikkaYhteensa")
        .Offset(-1, 0) = "Puoliperävaunuyhdistelmä (40 t/25 t)"
        .Offset(-1, 1).Value = "Osamatka 2"
        .Offset(-1, 2).Value = 230
        .Offset(-1, 3).Value = TextBox1.Value
        .Offset(-1, 4).Value = 0.64 * 25
    End With
    Call Kuljetukset(Range("LogistiikkaYhteensa").Offset(-1, 0))

    Call cmdRivinLisaysLog
    Call VarjaaRiviLog
    Application.EnableEvents = False

    With Range("LogistiikkaYhteensa")
        .Offset(-1, 0).Value = "Sekatavarajuna, diesel"
        .Offset(-1, 1).Value = "Osamatka 2"
        .Offset(-1, 2).Value = 280
        .Offset(-1, 3).Value = TextBox1.Value / 1000
        .Offset(-1, 4).Value = ""
    End With
    Call Kuljetukset(Range("LogistiikkaYhteensa").Offset(-1, 0))

    Call cmdRivinLisaysLog
    Call VarjaaRiviLog
    Application.EnableEvents = False

    With Range("LogistiikkaYhteensa")
        .Offset(-1, 0).Value = "Autolautta"
        .Offset(-1, 1).Value = "Osamatka 2"
        .Offset(-1, 2).Value = 360
        .Offset(-1, 3).Value = TextBox1.Value / 1000
        .Offset(-1, 4).Value = ""
    End With
    Call Kuljetukset(Range("LogistiikkaYhteensa").Offset(-1, 0))

    Application.EnableEvents = True

End Sub
```

Kuva 15. Oletusskenaariot jakeluun. Oletusskenaario 1: ”Toimittajalta tehtaalle: Pakkausmateriaalit (ei lasi)”

OletusSkenaario1-makro täyttää ajoneuvolistaan yhteensä kolme kuljetusvälinettä. Ajoneuvolistaan täytetään ajoneuvon tyyppi, matkan tyyppi, matkan pituus, kuljetettava massa ja maantieliikenteen tapauksessa oletettu kuormausaste. Lisäksi arvoja vastaavat päästökertoimet ja laskentaperiaatteet lisätään erillisellä makrolla. Ennen jokaista lisäystä ohjelma kutsuu rivinlisäysmakroa ja värjää rivin valkoiseksi omalla makrollaan, jotta oletusskenaario erottuisi käyttäjän itse täyttämistä riveistä. Application.EnableEvents asetus kytketään jokaisessa välissä pois päältä, sillä rivin lisäys makro muuttaa tätä asetusta. Ohjelman lopuksi kytketään Application.EnableEvents asetus päälle.

Makrot OletusSkenaario2 ja OletusSkenaario3 toimivat lähes täysin samalla tavalla, kuten OletusSkenaario1. Kuitenkin viimeisen neljännen OletusSkenaario4-makron toiminta poikkeaa kolmesta ensimmäisestä. Tämä skenaario on sen verran monimutkaisempi, että sitä varten ohjelma aukaisee vielä toisen käyttöliittymän, johon käyttäjän täytyy täyttää lisää tietoja. OletusSkenaario4-makron käynnistyessä avataan siis JakelunOletusSkenaario4-käyttöliittymä, joka vastaa tehtaalta loppukäyttäjälle skenaariosta. Kuvakaappaus käyttöliittymästä VBE:ssä on nähtävissä kuvassa 16.



Kuva 16. Jakelun oletusskenaario: ”Tehtaalta loppukäyttäjälle” -käyttöliittymä

Oletusskenaariossa kysytään käyttäjältä oletusskenaarion kannalta välttämättömiä suhdelukuja, joiden perusteella PEF-ohjeistuksen mukaiset kuljetusvälineet määritellään. (Euroopan komissio, 2021, s. 49–50.) IKE-hiilijalanjälkilaskurin käyttöoppaassa on tarkemmin selitetty, miten näitä suhdelukuja hyödynnetään ja laskentaperiaatteet toimivat kappaleessa 3.7.2. Käyttöliittymää ohjaava koodi on jaettu 13 makroon. Näistä löytyvät luonnollisesti alustusmakro, makrot kolmen eri painikkeen klikkaamiseen, virheidenkäsittelijä ja kaikkiaan 8 makroa erilaisille oletusskenaarion osille. Tässä käyttöliittymässä hyödynnetään

prosenttiluvuille tarkoitettuja moduulin sisäisiä muuttujia, jotka ovat määritelty proseduurien (makrojen, funktioiden, yms) ulkopuolella moduulin alussa.

Ohje ja Peruuta-painikkeet sekä alustusmakro ovat toiminnaltaan sen verran yksinkertaisia, että tässä ei pureuduta niihin tarkemmin. Näiden sijaan käydään ensimmäiseksi läpi Lisää Skenaario-painikkeen aktivoiva LisaaSkenaarioPainike_Click-makron toimintaa. Kuvakaappaus makrosta on nähtävissä kuvasta 17.

```
Private Sub LisaaSkenaarioPainike_Click()

    ' Lisätään skenaario tuotteen kuljetuksesta loppukäyttäjälle. Ohjelman käyttäjän
    ' tulee määritellä kuinka kuljetus jakautuu vähittäismyymälöille, jakelukeskuksille
    ' ja suorille kuljetuksille sekä kuinka kuljetukset jakautuvat paikallisiin,
    ' mantereen sisäisiin ja kansainvälisiin toimitusketjuihin.

    Application.ScreenUpdating = False

    On Error Resume Next
    Pros1 = Prosentti1.Value / 100
    Pros11 = Prosentti11.Value / 100
    Pros12 = Prosentti12.Value / 100
    Pros13 = Prosentti13.Value / 100
    Pros2 = Prosentti2.Value / 100
    Pros21 = Prosentti21.Value / 100
    Pros22 = Prosentti22.Value / 100
    Pros23 = Prosentti23.Value / 100
    Pros3 = Prosentti3.Value / 100
    Pros4 = Prosentti4.Value / 100
    On Error GoTo 0

    'Virheiden tarkistus
    If VirheenTarkistus = True Then
        GoTo Skippaus
    End If

    ' Riippuen käyttäjän tekemistä valinnoista lomakkeessa, kutsutaan sopivia aliohjelmia.
    If Pros11 > 0 Then
        Call OletusSkenaario411
    End If
    If Pros12 > 0 Then
        Call OletusSkenaario412
    End If
    If Pros13 > 0 Then
        Call OletusSkenaario413
    End If
    If Pros21 > 0 Then
        Call OletusSkenaario421
    End If
    If Pros22 > 0 Then
        Call OletusSkenaario422
    End If
    If Pros23 > 0 Then
        Call OletusSkenaario423
    End If
    If Pros3 > 0 Then
        Call OletusSkenaario43
    End If
    If Pros4 > 0 Then
        Call OletusSkenaario44
    End If

    Unload Me


Skippaus:

    Call SuojaaSivu("Logistiikka")
    Application.EnableEvents = True
    Application.ScreenUpdating = True


End Sub
```

Kuva 17. LisaaSkenaarioPainike_Click-makro

Aivan ensimmäiseksi makrossa lisätään arvot tarvittaviin muuttujiin, jotka on siis määritelty makron ulkopuolella jo valmiiksi. Tämä määritys on tehty siksi, että samoja muuttujia voitaisiin käyttää suoraan muissakin saman moduulin makroissa ilman, että muuttujien arvot nollaantuvat. Tämän jälkeen suoritetaan virheentarkistus erillisellä VirheenTarkistus-funktiolla, jonka saadessa arvon True ohjelman toiminta siirretään varsinaisen skenaarioiden lisäämisen ja käyttöliittymän sulkemisen ohi. Jos VirheenTarkistus-funktio saa arvon False, virhettä ei löydetty ja ohjelma jatkaa skenaarioiden makroja kutsuvaan ehtorakenteeseen. Ehtorakenteessa ohjelma käy jokaisen prosenttiluvun sisältävän muuttujan läpi, ja kutsuu sopivaa oletusskenaariomakroa, mikäli luku on suurempaa kuin 0.

Seuraavaksi käydään läpi LisaaSkenaarioPainike_Click-makrossa kutsuttava VirheenTarkistus-funktion toimintaa läpi. Kuvissa 18 ja 19 ja 20 on esitelty kuvakaappaukset kyseisen makron koodista.

VirheenTarkistus-funktion periaatteena on palauttaa Boolean arvona True, jos käyttäjän antamissa tiedoissa on virheitä ja False, jos virheitä ei ole. Arvon ollessa True funktio myös värjää niiden syötettävien laatikoiden taustat keltaisella ja antaa käyttäjälle virheilmoituksen väärin merkatuista tiedoista. Tällä on pyritty siihen, että väärin syötettyjen tietojen tapauksessa käyttäjälle ilmaistaan selkeästi, mitä tulee korjata. Funktion toiminta alkaa muuttujien määrittämisellä ja arvojen määrittämisellä prosenttilukujen muuttujiin. Lisäksi alustetaan Virhe-muuttujaan arvo False ja VirheIlmoitus muuttujaan otsikkoteksti.

Tämän jälkeen testataan ensimmäiseksi, ovatko syötetyt tiedot numeerisia ja ovatko ne nollan ja sadan väliltä. Tämä toteutetaan For Each -silmukalla, joka käy kaikki käyttöliittymän osat läpi. Silmukan sisällä on kaksi sisäkkäistä ehtorakennetta, joista ulompi testaa onko käsiteltävä osio prosenttiluvun oletusarvoisesti sisältävä TextBox-objekti vai jokin muu. Jos kyseessä on TextBox, seuraava ehtorakenne testaa numeerisuuden ja arvon suuruuden. Mikäli virheitä ilmenee, muutetaan TextBox-laatikon taustaväri keltaiseksi ja määritetään Virhe muuttujan arvoksi True. Muussa tapauksessa taustaväri muutetaan valkoiseksi.

Silmukan jälkeen on oma ehtorakenteensa VirheIlmoitus-muuttujan uudelleen määritykselle. (Euroopan komissio, 2021, s.49–50.) Virheilmoitukseen lisätään edellisten tekstien perään

teksti siitä, että annettujen suhdelukujen tietotyypeissä on virheitä. Tämän jälkeen käydään pitkähköjen ehtorakenteiden avulla kaikki mahdolliset virheet läpi, jotta tiedetään, että kaikki annetut suhdeluvut ovat toisiinsa nähden syötetty oikein. Näihin ehtoihin ja sääntöihin voi perehtyä tarkemmin IKE-hiilijalanjälkilaskurin käyttöoppaasta kappaleesta 3.7.2, PEF-ohjeistuksesta tai tarkastelemalla kuvissa 18, 19 ja 20 esitettyjä koodeja tarkemmin. Funktio aina virheen löytäessään värjää virheitä sisältävien laatikoiden taustat ja lisää virheilmoitukseen tekstiä.

```
Private Function VirheenTarkistus() As Boolean

    ' Virheentarkistusta käyttäjän syöttämistä tiedoista. Koska mahdollisia virheitä on paljon
    ' ja lomake on kohtalaisen monimutkainen koostetaan virheilmoitukseen erikseen kaikki, mitä
    ' täyttäessä on tehty väärin.

    Dim Virhe As Boolean
    Dim VirheIlmoitus As String
    Dim ctrl As Control

    On Error Resume Next
    Pros1 = Prosentti1.Value / 100
    Pros11 = Prosentti11.Value / 100
    Pros12 = Prosentti12.Value / 100
    Pros13 = Prosentti13.Value / 100
    Pros2 = Prosentti2.Value / 100
    Pros21 = Prosentti21.Value / 100
    Pros22 = Prosentti22.Value / 100
    Pros23 = Prosentti23.Value / 100
    Pros3 = Prosentti3.Value / 100
    Pros4 = Prosentti4.Value / 100
    On Error GoTo 0

    Virhe = False
    VirheIlmoitus = "Suhdeluvuissa ilmeni seuraavia virheitä:"

    'Onko kaikki arvot numeerisia
    For Each ctrl In Me.Controls
        If TypeName(ctrl) = "TextBox" Then
            If IsNumeric(ctrl.Value) = False Or ctrl.Value < 0 Or ctrl.Value > 100 Then
                ctrl.BackColor = RGB(255, 255, 0)
                Virhe = True
            Else
                ctrl.BackColor = RGB(255, 255, 255)
            End If
        End If
    Next ctrl

    If Virhe = True Then
        VirheIlmoitus = VirheIlmoitus & Chr(13) & Chr(13) & _
        "Annettujen suhdelukujen tietotyypeissä on virheitä"
    End If

    '1. ja 2. tulee olla yhteensä 100%
    If Pros1 + Pros2 <> 1 Then
        VirheIlmoitus = VirheIlmoitus & Chr(13) & Chr(13) & _
        "Kohtien 1. ja 2. suhdelukujen tulee olla yhteensä 100%."
        Virhe = True
        Prosentti1.BackColor = RGB(255, 255, 0)
        Prosentti2.BackColor = RGB(255, 255, 0)
    Else
        Prosentti1.BackColor = RGB(255, 255, 255)
        Prosentti2.BackColor = RGB(255, 255, 255)
    End If
```

Kuva 18. VirheenTarkistus-funktio (1/3)

```
    '1.1, 1.2 ja 1,3 tulee olla yhteensä 100% jos 1. on muuta kuin 0%
    If Pros1 > 0 And Pros11 + Pros12 + Pros13 <> 1 Then
        VirheIlmoitus = VirheIlmoitus & Chr(13) & Chr(13) & _
        "Kohtien 1.1 ja 1.2 ja 1.3 suhdelukujen tulee olla " & _
        "yhteensä 100%, jos kohdan 1. arvo on suurempi, kuin 0%."
        Virhe = True
        Prosentti11.BackColor = RGB(255, 255, 0)
        Prosentti12.BackColor = RGB(255, 255, 0)
        Prosentti13.BackColor = RGB(255, 255, 0)
    ElseIf Pros1 = 0 And Pros11 + Pros12 + Pros13 <> 0 Then
        VirheIlmoitus = VirheIlmoitus & Chr(13) & Chr(13) & _
        "Kohtien 1.1 ja 1.2 ja 1.3 suhdelukujen tulee olla 0%, " & _
        "jos kohdan 1. arvo on 0%."
        Virhe = True
        Prosentti11.BackColor = RGB(255, 255, 0)
        Prosentti12.BackColor = RGB(255, 255, 0)
        Prosentti13.BackColor = RGB(255, 255, 0)
    Else
        Prosentti11.BackColor = RGB(255, 255, 255)
        Prosentti12.BackColor = RGB(255, 255, 255)
        Prosentti13.BackColor = RGB(255, 255, 255)
    End If

    '2.1, 2.2 ja 2.3 tulee olla yhteensä 100% jos 2. on muuta kuin 0%
    If Pros2 > 0 And Pros21 + Pros22 + Pros23 <> 1 Then
        VirheIlmoitus = VirheIlmoitus & Chr(13) & Chr(13) & _
        "Kohtien 2.1 ja 2.2 ja 2.3 suhdelukujen tulee olla yhteensä " & _
        "100%, jos kohdan 2. arvo on suurempi, kuin 0%."
        Virhe = True
        Prosentti21.BackColor = RGB(255, 255, 0)
        Prosentti22.BackColor = RGB(255, 255, 0)
        Prosentti23.BackColor = RGB(255, 255, 0)
    ElseIf Pros2 = 0 And Pros21 + Pros22 + Pros23 <> 0 Then
        VirheIlmoitus = VirheIlmoitus & Chr(13) & Chr(13) & _
        "Kohtien 2.1 ja 2.2 ja 2.3 suhdelukujen tulee olla 0%, " & _
        "jos kohdan 2. arvo on 0%."
        Virhe = True
        Prosentti21.BackColor = RGB(255, 255, 0)
        Prosentti22.BackColor = RGB(255, 255, 0)
        Prosentti23.BackColor = RGB(255, 255, 0)
    Else
        Prosentti21.BackColor = RGB(255, 255, 255)
        Prosentti22.BackColor = RGB(255, 255, 255)
        Prosentti23.BackColor = RGB(255, 255, 255)
    End If
```

Kuva 19. VirheenTarkistus-funktio (2/3)

```
    '3. ja 4. tulee olla yhteensä yhtä paljon, kuin 2.
    If Pros3 + Pros4 <> Pros2 Then
        VirheIlmoitus = VirheIlmoitus & Chr(13) & Chr(13) & _
        "Kohtien 3. ja 4. suhdelukujen tulee olla yhteensä " & _
        "suuruudeltaan yhtä suuri, kuin kohta 2."
        Virhe = True
        Prosentti3.BackColor = RGB(255, 255, 0)
        Prosentti4.BackColor = RGB(255, 255, 0)
    Else
        Prosentti3.BackColor = RGB(255, 255, 255)
        Prosentti4.BackColor = RGB(255, 255, 255)
        Prosentti2.BackColor = RGB(255, 255, 255)
    End If

    If Virhe = True Then
        MsgBox VirheIlmoitus, vbOKOnly, "Virheilmoitus"
        VirheenTarkistus = True
    End If

End Function
```

Kuva 20. VirheenTarkistus-funktio (3/3)

Kuvassa 20 nähdään funktion lopussa, että mikäli Virhe-muuttuja on funktion toiminnan aikana saanut arvon True, esitetään käyttäjälle ponnahdusikkunana kaikki VirheIlmoitus-muuttujaan kerätty teksti ja muutetaan funktion arvoksi True.

VirheenTarkistus-funktion jälkeen LisaaSkenaarioPainike_Click-makrossa käydään ehtorakenteiden avulla läpi kaikki prosenttiluvut, ja kutsutaan sopivaa oletusskenaarion täyttävää makroa, mikäli arvo on suurempaa kuin nolla. Nämä oletusskenaarioita täyttävät makrot ovat rakenteeltaan hyvin samankaltaisia, joten käydään tässä esimerkin omaisesti läpi ainoastaan yksi oletusskenaariomakro. Kuvassa 21 on esitetty makron OletusSkenaario411 koodi, joka siis vastaa skenaariosta 1.1, jossa tuote kuljetetaan suoraan tehtaalta loppukäyttäjälle paikallisena toimitusketjuna.

```
Private Sub OletusSkenaario411()
    ' 1.1 Tehtaalta suoraan loppukäyttäjälle:
    ' a) paikallinen toimitusketju X %: 1 200 km kuorma-autolla (> 32 t, Euro 4)

    Call cmdRivinLisaysLog
    Call VarjaaRiviLog
    Application.EnableEvents = False

    Call VapautaSivu("Logistiikka")

    With Range("LogistiikkaYhteensa")
        .Offset(-1, 0).Value = "Puoliperävaunuyhdistelmä (40 t/25 t)"
        .Offset(-1, 1).Value = "Osamatka 3"
        .Offset(-1, 2).Value = 1200
        .Offset(-1, 3).Value = Pros1 * Pros11 * Massa
        .Offset(-1, 4).Value = 0.64 * 25
    End With
    Call Kuljetukset(Range("LogistiikkaYhteensa").Offset(-1, 0))

    Application.EnableEvents = True

End Sub
```

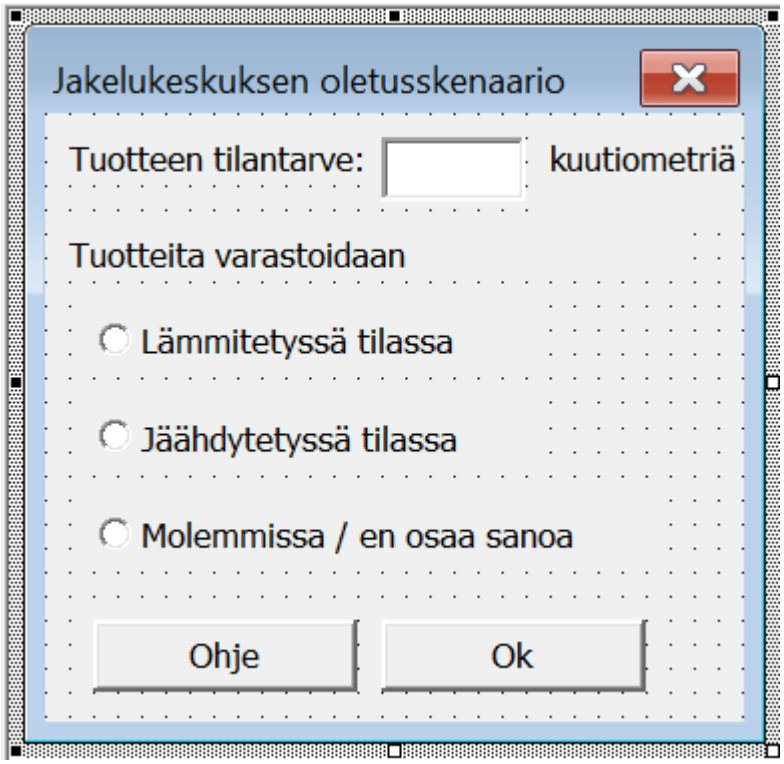
Kuva 21. OletusSkenaario411-makro

Oletusskenaariossa 1.1 ajoneuvojen taulukkoon lisätään mantereen sisäinen kuljetus 1200 kilometriä kuorma-autolla. (Euroopan komissio, 2021, s. 50.) Makro aluksi lisää taulukkoon rivin, värjää sen valkoiseksi, jotta se erottuu käyttäjän itse täyttämistä riveistä, muuttaa tapahtuma-asetukset ja vapauttaa sivun. Tämän jälkeen uudelle riville vain syötetään oletusskenaarion mukaiset tiedot, joissa käytetään parhaiten PEF-ohjeistuksen linjaamaa ajoneuvoa vastaavaa kuljetusvälinettä, oikeaa osamatkaa ja kuormausasteena käytetään PEF-ohjeistuksen määräämää 64 prosentin kuormausastetta. Laskennalliseksi kuorman massaksi määritellään se osuus kuljetettavan tuotteen massasta, joka vastaa oletusskenaarion osuutta kuljetuksista. Lopuksi vielä kutsutaan Kuljetukset-makroa, joka määrittää päästökertoimet ja laskentakaavat oikein riville.

4.2.2 Oletusskenaariot varastointiin

Varastointiin liittyviä oletusskenaarioita on kaksi, joten käyttöliittymät ovat erikseen jakelukeskukselle ja vähittäismyymälälle. PEF-ohjeistus määrittelee näille erilaisia arvoja oletusskenaarioihin, joten nämä oli helpointa toteuttaa erillisinä käyttöliittyminä. Käydään ensin läpi

jakelukeskuksen oletusskenaarion käyttöliittymä, josta on nähtävissä kuvakaappaus VBE:ssä kuvassa 22.



Kuva 22. Jakelukeskuksen oletusskenaarion käyttöliittymä VBE:ssä

Jakelukeskuksen oletusskenaarion käyttöliittymään siis syötetään tuotteen tilantarve kuutiometreinä, ja valitaan, miten tuote on varastoitu. Näiden tietojen perusteella ohjelma sitten syöttää PEF-ohjeistuksen mukaisen oletusskenaarion taulukkoon. Käyttöliittymän taustalla vaikuttaa kuusi makroa, joista kolme ovat eri skenaarioiden toteuttavia makroja, yksi on alustusmakro, yksi ohjeen avausmakro ja yksi Ok-painikkeeseen reagoiva makro. Ohjeen avaaminen ja käyttöliittymän alustus on jo käyty läpi, joten aloitetaan käyttöliittymän koodin käsittely Ok-painikkeeseen reagoivasta Ok_Click-makrosta. Kuvakaappaus makron koodista on nähtävissä kuvassa 23.

```
Private Sub Ok_Click()

    ' Oletusskenaario varastointi jakelukeskuksessa. Tästä napista syötetään tiedot listaan.

    'Virheentarkistus
    If IsNumeric(TextBox1.Value) = False Or TextBox1.Value < 0 Then
        TextBox1.BackColor = RGB(255, 255, 0)
        MsgBox "Tuotteen tilantarpeen arvo on virheellinen."
        Exit Sub
    Else
        TextBox1.BackColor = RGB(255, 255, 255)
    End If

    ' Jos virhettä ei havaita, muutetaan kaikki taustavärit valkoisiksi.
    OptionButton1.BackColor = RGB(255, 255, 255)
    OptionButton2.BackColor = RGB(255, 255, 255)
    OptionButton3.BackColor = RGB(255, 255, 255)

    'Riippuen siitä, mikä valintanappula oli valittuna, ajetaan sopiva aliohjelma.
    If OptionButton1 = True Then
        Call LammminJakeluvarasto
    ElseIf OptionButton2 = True Then
        Call KylmaJakeluvarasto
    ElseIf OptionButton3 = True Then
        Call LamminjaKylmaJakeluvarasto
    Else:
        MsgBox "Varaston tyyppiä ei valittu."
        OptionButton1.BackColor = RGB(255, 255, 0)
        OptionButton2.BackColor = RGB(255, 255, 0)
        OptionButton3.BackColor = RGB(255, 255, 0)
        Exit Sub
    End If

    Unload Me

End Sub
```

Kuva 23. Jakelukeskuksen oletusskenaarion käyttöliittymän Ok_Click makro

Ok_Click-makro aloittaa virheentarkistuksella, jossa testataan, onko tuotteen tilantarve ilmoitettu numeerisena arvona, ja onko se suurempaa kuin nolla. Mikäli näin ei ole, laatikon taustaväri värjätään keltaiseksi, käyttäjälle annetaan virheilmoitus ja makron toiminta lopetetaan. Muussa tapauksessa taustat värjätään valkoiseksi ja ohjelman toimintaa jatketaan ehtorakenteeseen, joka valitsee kutsuttavan oletusskenaarion toteuttavan makron sen perusteella, mikä kohta käyttöliittymästä oli valittuna. Mikäli mitään varastointityyppiä ei valittu, käyttäjälle annetaan virheilmoitus, valintapainikkeiden taustat värjätään keltaiseksi ja ohjelman toiminta lopetetaan. Jos kaikki on kuitenkin mennyt toimintaperiaatteen mukaan, ehtorakenteen jälkeen käyttöliittymä suljetaan ja oletusskenaario on valmis.

Käyttöliittymä voi siis käyttää kolmea vaihtoehtoista oletusskenaarion toteuttavaa makroa: LamminJakeluvarasto, KylmaJakeluVarasto ja LamminjaKylmäJakuluvarasto. Tässä oppaassa näistä esitellään ensimmäinen, joka on rakenteeltaan yksinkertaisin. Kaksi muuta

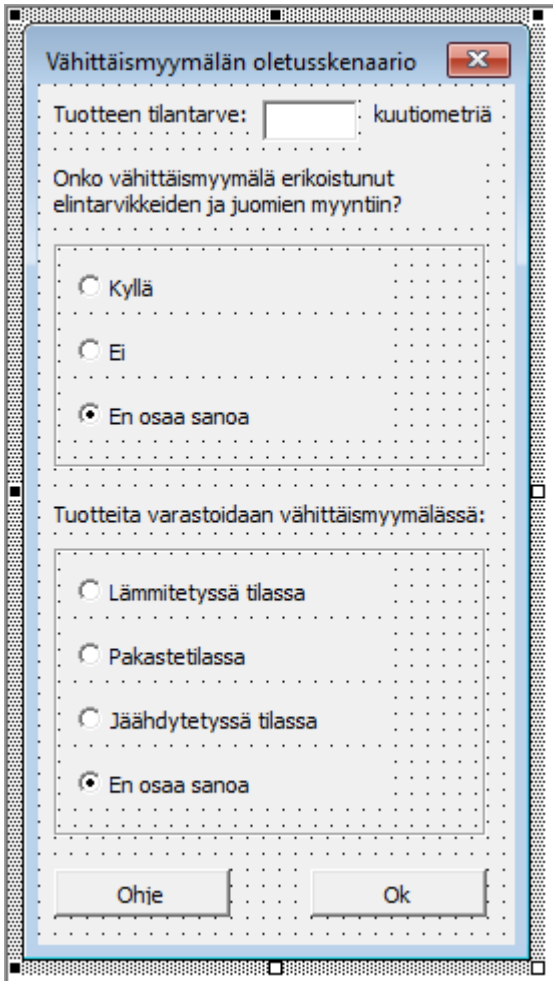
toimivat samalla periaatteella, mutta niissä lisätään yhden rivin sijaan useita varastoinnin taulukossa sekä kylmäainekaasujen taulukossa. Kuvassa 24 on esitetty kuvakaappaus Lammin-Jakeluvarasto-makron koodista.

```
Sub LammminJakeluvarasto()

    ' Syötetään PEF-standardin mukaiset tiedot lämpimän jakeluvaraston tapauksessa.

    Call cmdRivinLisaysVarast
    Call VarjaaRiviVarasto
    Call VapautaSivu("Logistiikka")

    Application.EnableEvents = False
    With Range("VarastointiYhteensa")
        .Offset(-1, 0).Value = "Jakelukeskus"
        .Offset(-1, 1).Value = "Tuotteiden varasto"
        .Offset(-1, 2).Value = "Lämmin varasto"
        .Offset(-1, 3).Value = CSng(TextBox1.Value)
        '5 metriä korkeat hyllyt ja puollet lattiapinta-alasta voidaan täyttää.
        .Offset(-1, 4).Value = 24000
        ' Tämä on tarkistettava. Standardissa 52 viikkoa epäsuorasti ilmaistuna
        .Offset(-1, 6).Value = 7
        .Offset(-1, 7).Formula = "=(60000/7)*" & _
        SarakeKirjain(Range("VarastointiYhteensa").Offset(-1, 6).Column) & _
        Range("VarastointiYhteensa").row - 1
        ' 101,42 kWh vastaa standardin ilmoittamaa 10Nm3 maakaasua. LHV 36,51 MJ/m3
        .Offset(-1, 8).Value = 101.42
        Application.EnableEvents = True
        .Offset(-1, 9).Value = "Maakaasu"
        Application.EnableEvents = False
    End With

    Call Varastointi(Range("VarastointiYhteensa").Offset(-1, 2))
    Call SuojaaSivu("Logistiikka")
    Application.EnableEvents = True

End Sub
```

Kuva 24. LamminJakeluvarasto-makro

Toiminnaltaan LamminJakeluvarasto on hyvin yksinkertainen makro ja sen toimintaperiaate on tuttu kuljetusten oletusskenaarioista. (Euroopan komissio, 2021, s.50–51.) Se lisää aluksi uuden rivin varastojen listaan, värjää kyseisen rivin ja vapauttaa sivun suojaukset. Tämän jälkeen riville lisätään PEF-ohjeistuksen mukaiset oletusarvot. Osa oletusarvoista on jouduttu valitsemaan käyttämällä omaa tulkintaa, mutta näistä on tarkemmin kerrottu IKE-hiilijalanjälkilaskurin käyttöoppaassa kappaleissa 3.7.5 ja 3.7.6. Tietojen lisäämisen jälkeen lisätään Varastointi-makrolla tarvittavat laskentakaavat, suojataan sivu ja palautetaan asetukset.

Toinen varastointiin liittyvä käyttöliittymä liittyy varastointiin vähittäismyymälässä. Vaikka käyttöliittymä ja sen periaatteet eivät eroa paljon jakelukeskuksessa varastoinnista, ovat käyttöliittymät PEF-ohjeistuksen eroavaisuuksien takia hieman erinäköisiä. Myös koodin rakenne jostain syystä haluttiin toteuttaa hieman eri tavalla. Kuvassa 25 on esitetty kuvakaappaus käyttöliittymästä VBE:ssä.



Kuva 25. Varastoinnin vähittäismyymälän oletusskenaarion käyttöliittymä VBE:ssä

Vähittäismyymälän oletusskenaarion käyttöliittymässä kysytään myös tuotteen tilantarve kuutiometreinä, mutta myös vähittäismyymälän erikoistumista elintarvikkeiden myyntiin. (Euroopan komissio, 2021, s. 51.) Lisäksi varastointityyppiin on lisätty vaihtoehto ”Pakastetila”, koska tämä on jostain syystä PEF-ohjeistuksissa erikseen määritetty vähittäismyymälöiden tapauksessa.

Vähittäismyymälän oletusskenaarion käyttöliittymän koodi on jaettu vain kolmeen makroon: ohjeen avausmakroon, alustusmakroon ja Ok-painikkeeseen reagoivaan Ok_Click-makroon. Tämä makro eroaa huomattavasti jakelukeskuksen oletusskenaarion käyttöliittymän vastaavasta makrosta. Kuvakaappaus kyseisen makron alkupuoliskosta on nähtävissä kuvasta 26.

```
Private Sub Ok_Click()

    'Virheentarkistus
    If IsNumeric(TextBox1.Value) = False Or TextBox1.Value < 0 Then
        TextBox1.BackColor = RGB(255, 255, 0)
        MsgBox "Tuotteen tilantarpeen arvo on virheellinen."
        Exit Sub
    Else
        TextBox1.BackColor = RGB(255, 255, 255)
    End If

    OptionButton1.BackColor = RGB(255, 255, 255)
    OptionButton2.BackColor = RGB(255, 255, 255)
    OptionButton3.BackColor = RGB(255, 255, 255)

    Dim ekulutus As Integer
    Dim vtyyppi As String
    Dim vltyyppi As String
    Dim KylAine As Boolean
    Dim Osuus As Single

    KylAine = False
    Osuus = 1
    vtyyppi = "Tuotteiden varasto"
    If OptionButton1 = True Then
        If OptionButton5 = True Then
            ekulutus = 400
        ElseIf OptionButton6 = True Then
            ekulutus = 150
        ElseIf OptionButton7 = True Then
            ekulutus = 300
        Else
            MsgBox "Erikoistumista ei valittu."
            OptionButton5.BackColor = RGB(255, 255, 0)
            OptionButton6.BackColor = RGB(255, 255, 0)
            OptionButton7.BackColor = RGB(255, 255, 0)
        End If
    ElseIf OptionButton2 = True Then
        KylAine = True
        ekulutus = ekulutus + 2700
    ElseIf OptionButton3 = True Then
        KylAine = True
        ekulutus = ekulutus + 1900
    ElseIf OptionButton4 = True Then
        KylAine = True
        Osuus = 0.2
        If OptionButton5 = True Then
            ekulutus = 400 * 0.8 + (400 + 1900) * 0.2
        ElseIf OptionButton6 = True Then
            ekulutus = 150 * 0.8 + (150 + 1900) * 0.2
        ElseIf OptionButton7 = True Then
            ekulutus = 300 * 0.8 + (300 + 1900) * 0.2
        Else
            MsgBox "Erikoistumista ei valittu."
            OptionButton5.BackColor = RGB(255, 255, 0)
            OptionButton6.BackColor = RGB(255, 255, 0)
            OptionButton7.BackColor = RGB(255, 255, 0)
        End If
    Else:
        MsgBox "Varaston tyyppiä ei valittu."
        OptionButton1.BackColor = RGB(255, 255, 0)
        OptionButton2.BackColor = RGB(255, 255, 0)
        OptionButton3.BackColor = RGB(255, 255, 0)
        OptionButton4.BackColor = RGB(255, 255, 0)
        Exit Sub
    End If
```

Kuva 26. Vähittäismyymälän oletusskenaarion käyttöliittymän Ok_Click-makro (1/2)

Ok_Click-makron toimintaperiaate vähittäismyymälän oletusskenaariossa eroaa jakelukeskuksen vastaavasta siten, että myös oletusskenaario syötetään tästä makrosta ilman vastaavia aliohjelmia. Käyttäjän syötteiden perusteella muutetaan arvoja sopiviin muuttujiin, joiden avulla sitten oletusskenaarion mukaiset tiedot täytetään.

Makron ensimmäisessä osassa tarkistetaan käyttäjän syötteet virheiden varalta samaan tapaan, kuten jakelukeskuksen käyttöliittymän vastaavassa makrossa. (Euroopan komissio, 2021, s. 51.) Tämän jälkeen määritetään tarvittavat muuttujat ja alustetaan näille arvoja. Kun alustavat arvot KylAine ja vtyyppi-muuttujille on määrätty, aloitetaan ensiksi PEF-ohjeistuksen mukaisen energiankulutuksen selvittäminen annetuilla vaihtoehdoilla. Aluksi kulutuksen niin sanottu pohja-arvo määritetään vähittäismyymälän erikoistumisen perusteella, jonka jälkeen siihen lisätään kulutusta, mikäli käytössä on jäähdytysjärjestelmiä. Jos käytössä on sekä lämmitettyä että jäähdytettyä tilaa, oletetaan 20 prosenttia tilasta jäähdytetyksi. Energiankulutuksen määrittämisen jälkeen siirrytään muiden tietojen määrittämiseen, joista on lisää kuvassa 27.

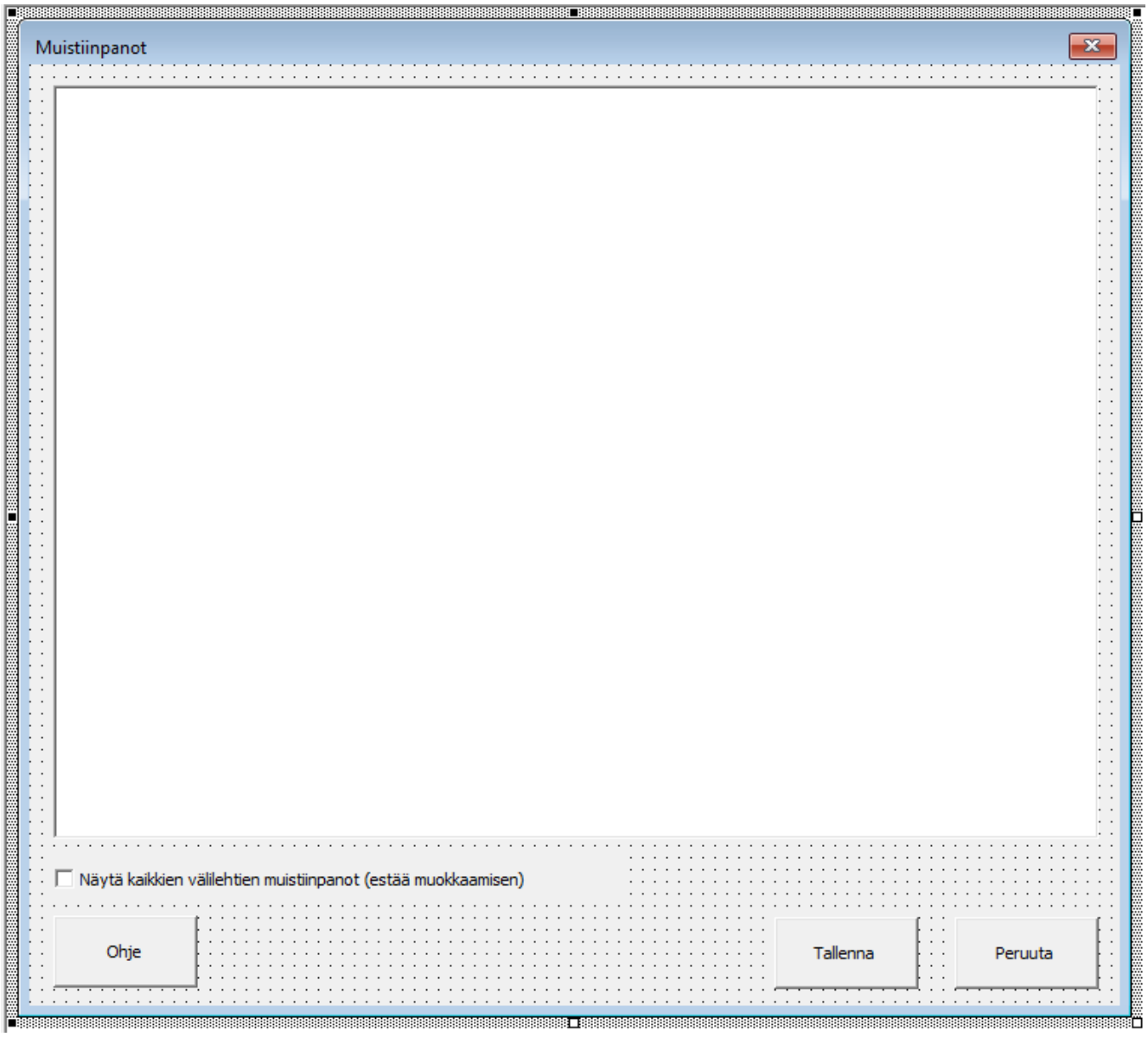
```
    Call cmdRivinLisaysVarast
    Call VarjaaRiviVarasto
    Call VapautaSivu("Logistiikka")
    Application.EnableEvents = False

    If OptionButton1 = True Then
        vltyyppi = "Lämmin varasto"
    ElseIf OptionButton2 = True Or OptionButton3 = True Then
        vltyyppi = "Jäähdytetty varasto"
    ElseIf OptionButton4 = True Then
        vltyyppi = "Lämmin varasto"
    End If


    With Range("VarastointiYhteensa")
        .Offset(-1, 0).Value = "Vähittäismyymälä"
        .Offset(-1, 1).Value = vtyyppi
        .Offset(-1, 2).Value = vltyyppi
        .Offset(-1, 3).Value = CSng(TextBox1.Value)
        .Offset(-1, 4).Value = 2000 '2 metriä korkeat hyllyt ja puolet lattiapinta-alasta voidaan täyttää.
        .Offset(-1, 6).Value = 7 ' Tämä on tarkistettava. Standardissa 52 viikkoa epäsuorasti ilmaistuna
        .Offset(-1, 7).Formula = "=(2000/7)*" & _
        SarakeKirjain(Range("VarastointiYhteensa").Column + 6) & Range("VarastointiYhteensa").row - 1
        .Offset(-1, 8).Value = ekulutus ' 101,42 kWh vastaa standardin ilmoittamaa 10Nm3 maakaasua. LHV 36,51 MJ/m3
        Application.EnableEvents = True
        .Offset(-1, 9).Value = "Suomen keskiarvo"
        Application.EnableEvents = False
    End With

    Call Varastointi(Range("VarastointiYhteensa").Offset(-1, 2))

    If KylAine = True Then
        Call cmdRivinLisaysKylmaAineet
        Call VarjaaRiviKylmaAineet
        Call VapautaSivu("Logistiikka")
        Application.EnableEvents = False

        With Range("KylmaKaasutYhteensa")
            .Offset(-1, 0).Value = "R404A"
            .Offset(-1, 1).Value = "Tuotteiden varasto"
            .Offset(-1, 2).Value = CSng(TextBox1.Value)
            .Offset(-1, 3).Value = 7
            .Offset(-1, 4).Value = 2000
            .Offset(-1, 5).Value = "=(2000/7)*" & SarakeKirjain(Range("KylmaKaasutYhteensa").Column + 3) & _
            Range("KylmaKaasutYhteensa").row - 1
            .Offset(-1, 7).Value = 3.922
        End With

    End If

    Unload Me

    Call SuojaaSivu("Logistiikka")
    Application.EnableEvents = True

End Sub
```

Kuva 27. Vähittäismyymälän oletusskenaarion käyttöliittymän Ok_Click-makro (2/2)

Energiankulutuksen määrittämisen jälkeen lisätään varastoinnin taulukkoon rivi, värjätään se, vapautetaan sivu ja muutetaan tapahtumien asetukset. Sitten testataan ehtorakenteen avulla, mikä varaston tyyppi on valittuna. Tämän jälkeen tuttuun tapaan lisätään uudelle riville tietoja aiempien määritysten mukaisesti. Tämän lisäksi testataan, onko jäähdytysjärjestelmiä käytössä, jolloin lisätään myös tarvittava kylmäainekaasujen kulutus. Lopuksi suljetaan käyttöliittymä, suojataan sivu ja palautetaan asetukset.

4.2.3 Muistiinpanot

IKE-hiilijalanjälkilaskurin kehityksen yhteydessä huomattiin, että käyttäjillä on usein tarve kirjoittaa muistiinpanoja syöttämiin tietoihin tai laskentaan liittyen. Koska laskuri sisältää taulukoiden ulkopuolisissa soluissa dataa ja kaavoja ja solujen paikat saattavat muuttua, koettiin järkeväksi kehittää laskuriin toiminto muistiinpanojen kirjoittamista varten. Tämä toiminto päätettiin toteuttaa käyttöliittymän avulla, josta on nähtävissä kuvakaappaus kuvassa 28.



Kuva 28. Muistiinpanot-käyttöliittymä

Muistiinpanot-työkalun suurelle valkoiselle tekstikentälle käyttäjän on mahdollista kirjoittaa vapaasti omia muistiinpanojaan välilehtikohtaisesti. Käyttäjä voi myös halutessaan tarkastella kaikkia muistiinpanoja, mutta tällöin muokkaaminen on estetty. Jotta muistiinpanot tallentuvat, käyttäjän tulee painaa Tallenna-painiketta ennen käyttöliittymän sulkemista.

Muistiinpanot-käyttöliittymä sisältää kuusi makroa, joista kaksi toimivat käyttöliittymän alustuksessa, yksi Näytä kaikkien välilehtien muistiinpanot -valintaruudun muuttumisen yhteydessä, ja kolme muuta kolmen alalaidassa esiintyvän painikkeen klikkausten yhteydessä. Kuvassa 29 esitellään kaksi käyttöliittymän alustuksesta vastaavaa makroa.

```
Private Sub UserForm_Initialize()

    With Muistiinpanot
        .Height = 498.5
        .Width = 556
    End With

    Me.Top = Application.Top + (Application.UsableHeight / 2) - (Me.Height / 2)
    Me.Left = Application.Left + (Application.UsableWidth / 2) - (Me.Width / 2)

    Call TekstinAlustus

End Sub
```

```
Private Sub TekstinAlustus()

    If ActiveSheet.Name = "IKE-Laskentamenetelmä" Then
        TextBox1.Value = ThisWorkbook.Sheets("Dataa").Range("Muistiinpanot").Offset(1, 1).Value
    ElseIf ActiveSheet.Name = "Työkalut" Then
        TextBox1.Value = ThisWorkbook.Sheets("Dataa").Range("Muistiinpanot").Offset(2, 1).Value
    ElseIf ActiveSheet.Name = "Raaka-aineet" Then
        TextBox1.Value = ThisWorkbook.Sheets("Dataa").Range("Muistiinpanot").Offset(3, 1).Value
    ElseIf ActiveSheet.Name = "Pakkausmateriaalit" Then
        TextBox1.Value = ThisWorkbook.Sheets("Dataa").Range("Muistiinpanot").Offset(4, 1).Value
    ElseIf ActiveSheet.Name = "Jätteet ja sivuvirrat" Then
        TextBox1.Value = ThisWorkbook.Sheets("Dataa").Range("Muistiinpanot").Offset(5, 1).Value
    ElseIf ActiveSheet.Name = "Energia" Then
        TextBox1.Value = ThisWorkbook.Sheets("Dataa").Range("Muistiinpanot").Offset(6, 1).Value
    ElseIf ActiveSheet.Name = "Logistiikka" Then
        TextBox1.Value = ThisWorkbook.Sheets("Dataa").Range("Muistiinpanot").Offset(7, 1).Value
    ElseIf ActiveSheet.Name = "Yhteenveto" Then
        TextBox1.Value = ThisWorkbook.Sheets("Dataa").Range("Muistiinpanot").Offset(8, 1).Value
    ElseIf ActiveSheet.Name = "Tasetarkastelu" Then
        TextBox1.Value = ThisWorkbook.Sheets("Dataa").Range("Muistiinpanot").Offset(9, 1).Value
    ElseIf ActiveSheet.Name = "Päästökertoimet" Then
        TextBox1.Value = ThisWorkbook.Sheets("Dataa").Range("Muistiinpanot").Offset(10, 1).Value
    End If

End Sub
```

Kuva 29. Muistiinpanot-käyttöliittymän alustusmakro

Aluksi normaalissa automattisesti käyttöliittymän avaamisen yhteydessä käynnistyvässä Userform_Initialize-makrossa säädetään tavalliseen tapaan käyttöliittymän sijainti ja koko, mutta näiden lisäksi kutsutaan TekstinAlustus-makroa, joka on esitetty myös kuvassa 29.

Tämän makron tarkoituksena on lisätä oikea teksti käyttöliittymän tekstilaatikkoon sen mukaan, mikä välilehti on avoinna. Tiedot ovat tallennettuina laskurin Dataa-välilehdelle tiettyihin soluihin, joista laskuri osaa ne käydä noutamassa.

Seuraavaksi esitellään Näytä kaikkien välilehtien muistiinpanot-painikkeesta aktivoituva CheckBox1_Click-makro, joka listaa käyttöliittymän tekstikenttään kaikkien välilehtien muistinpanot valinnan ollessa aktiivinen. Kuvassa 30 on esitetty kuvakaappaus kyseisestä makrosta.

```
Private Sub CheckBox1_Click()

    Dim strTeksti As String
    Dim rngAlue As Range
    Dim i As Integer

    VapautaSivu ("Dataa")

    Set rngAlue = ThisWorkbook.Sheets("Dataa").Range("Muistiinpanot").CurrentRegion

    If CheckBox1 = True Then
        For i = 2 To 11
            If Not rngAlue.Cells(i, 2).Value = "" Then
                strTeksti = strTeksti & CStr(rngAlue.Cells(i, 1).Value) & _
                ": " & CStr(rngAlue.Cells(i, 2).Value) & Chr(13) & Chr(13)
            End If
        Next i
        TextBox1.Value = strTeksti
        TextBox1.Enabled = False
        CommandButton1.Enabled = False
    Else
        TextBox1.Enabled = True
        CommandButton1.Enabled = True
        Call TekstinAlustus
    End If

    SuojaaSivu ("Dataa")

End Sub
```

Kuva 30. Muistiinpanot käyttöliittymän CheckBox1_Click -makro

CheckBox1_Click-makron alussa määritellään muuttujat ja vapautetaan Dataa-välilehti. Tämän jälkeen määritellään rngAlue-muuttujaan Dataa-välilehdeltä muistinpanoihin liittyvien solujen alue. Sitten ehtorakenteen avulla testataan, onko valintaruutu valittuna vai ei. Mikäli on, käydään For-silmukkarakenteen avulla rngAlue-alueen tietoja läpi ja lisätään strTeksti-muuttujaan silmukan kierroksella käsiteltävän välilehden nimi, välilehteä koskevat muistiinpanot ja kaksi rivinvaihtoa. Silmukan jälkeen lisätään strTeksti-muuttujan sisältämä teksti

käyttöliittymän tekstikenttään, estetään tekstikentän muokkaaminen ja estetään Tallenna-painikkeen painaminen. Jos valintaruutu ei ollut valittuna, tekstikentän muokkaaminen avataan, Tallenna-painike vapautetaan ja kutsutaan jo aiemmin esiteltyä TekstinAlustus-makroa. Ehtorakenteen jälkeen enää suojataan Dataa-välilehti.

Muistiinpanot-käyttöliittymän painikkeista Peruuta- ja Ohje-painikkeet nimiensä mukaisesti joko sulkevat käyttöliittymän tai avaavat ohjeen oikeasta kohtaa OhjeenAukaisu makron avulla. Kuitenkin Tallenna-painikkeen toimintaan liittyy hieman enemmän kuin yksi rivi koodia, joka on esitetty kuvassa 31.

```
Private Sub CommandButton1_Click()

    VapautaSivu ("Dataa")

    Dim rngAlue As Range
    Dim cell As Range

    Set rngAlue = ThisWorkbook.Sheets("Dataa").Range("Muistiinpanot").CurrentRegion

    For Each cell In rngAlue.Columns(1).Cells
        If cell.Value = ActiveSheet.Name Then
            cell.Offset(0, 1).Value = TextBox1.Value
            Exit For
        End If
    Next

    SuojaaSivu ("Dataa")

End Sub
```

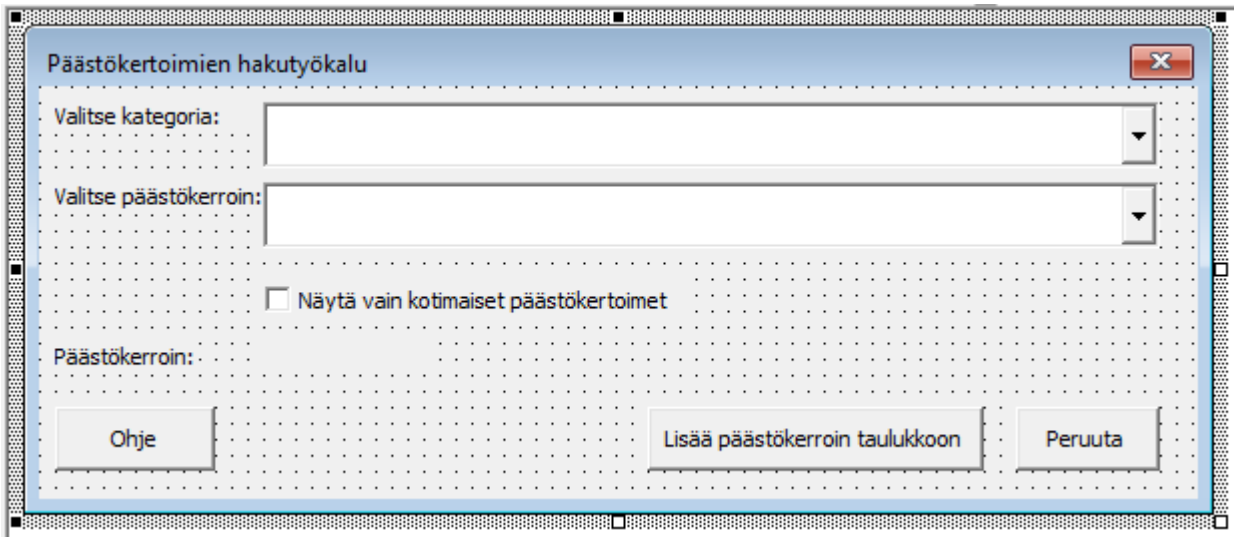
Kuva 31. Muistiinpanot käyttöliittymän CommanButton1_Click -makro

Muistiinpanot-käyttöliittymän CommandButton1_Click-makron toiminta on varsin yksinkertainen. Ensin vapautetaan Dataa-välilehti, määritellään muuttujat ja määritellään rngAlue-muuttujaan Dataa-välilehdeltä muistiinpanoja käsittelevä alue. Tämän jälkeen käydään For Each-silmukkarakenteen avulla kaikki alueen solut läpi, jolloin oikean välilehden nimen osuessa kohdalle viereiseen soluun tallennetaan sillä hetkellä käyttöliittymän tekstikentässä olevat tiedot. Lopuksi vielä suojataan Dataa-välilehti.

4.2.4 Päästökertoimien hakutyökalu

IKE-hiilijalanjälkilaskurissa on kaksi päästökertoimien hallintaan tarkoitettu työkalua: hakutyökalu ja lisäystyökalu. Päästökertoimen hakutyökalu on tarkoitettu yksittäisten päästökertoimien löytämiseen Päästökertoimet-välilehden tietokannasta ja sitä käytetään useimmissa laskentavälilehtien taulukoissa. Erityisen hyödyllinen se on raaka-aineiden päästökertoimien etsintään silloin, kun halutaan käyttää tietokantojen geneerisiä päästökertoimia. Päästökertoimien lisäystyökalua voidaan käyttää taas omien päästökertoimien lisäämiseen tietokantaan, jolloin nämä päästökertoimet ovat käytettävissä laskurissa.

Aloitetaan näistä kahdesta työkalusta monikäyttöisemmällä ja oleellisemmalla eli päästökertoimien hakutyökalusta. Käyttöliittymä sisältää pudotusvalikot ensin päästökertoimien kategorialle ja sitten itse päästökertoimille, pelkkien kotimaisten päästökertoimien näyttämiseen tarkoitetun valintaruudun, paikan itse päästökertoimelle sekä Ohje-, Lisää päästökerroin taulukkoon- ja Peruuta-painikkeet. Kuvakaappaus päästökertoimien hakutyökalun käyttöliittymästä on nähtävissä kuvasta 32.



Kuva 32. PKHakutyokalu-käyttöliittymä.

PKHakutyokalu-käyttöliittymä vaatii toimiakseen 8 erilaista makroa. Nämä sisältävät alustusmakron, pudotusvalikoiden muuttumiseen reagoivat makrot, valintaruudun klikkaamiseen reagoivan makron, painikkeisiin reagoivat makrot sekä virheentarkistajan. Aloitetaan

käyttöliittymän makrojen läpikäyminen alustusmakrosta UserForm_Initialize, joka on nähtävissä kuvassa 33.

```
Private Sub UserForm_Initialize()

    Dim nm As Name
    Dim sh As Worksheet
    On Error Resume Next

    Set sh = ThisWorkbook.Sheets("Päästökertoimet")

    For Each nm In Names
        If Left(CStr(nm.RefersTo), 16) = "=Päästökertoimet" And _
        Left(CStr(nm.Name), 18) = "Päästökertoimet!PK" Then
            ComboBox1.AddItem (sh.Range(nm.Name).Value)
        End If
    Next nm


    If ActiveSheet.Name = "Raaka-aineet" Then
        ComboBox1.Value = "Raaka-aineet"
        CheckBox1.Visible = True
    ElseIf ActiveSheet.Name = "Pakkausmateriaalit" Then
        ComboBox1.Value = "Pakkausmateriaalit"
        CheckBox1.Visible = False
    ElseIf ActiveSheet.Name = "Päästökertoimet" Then
        ComboBox1.Value = "Ajoneuvot"
        CheckBox1.Visible = False
        LisaaKerroin.Visible = False
    ElseIf ActiveSheet.Name = "Jätteet ja sivuvirrat" And Not _
    ActiveSheet.Shapes("PKHakuJAtNappi").Fill.ForeColor.RGB = RGB(218, 217, 215) Then
        ComboBox1.Value = "Jätelajikkeet"
        CheckBox1.Visible = False
    ElseIf ActiveSheet.Name = "Jätteet ja sivuvirrat" And Not _
    ActiveSheet.Shapes("PKHakuPesNappi").Fill.ForeColor.RGB = RGB(218, 217, 215) Then
        ComboBox1.Value = "Pesuaineet"
        CheckBox1.Visible = False
    Else
        ComboBox1.Value = "Ajoneuvot"
        CheckBox1.Visible = False
    End If

    With PKHakutyokalu
        .Height = 182.5
        .Width = 445
    End With

    Me.Top = Application.Top + (Application.UsableHeight / 2) - (Me.Height / 2)
    Me.Left = Application.Left + (Application.UsableWidth / 2) - (Me.Width / 2)

End Sub
```

Kuva 33. PKHakutyokalu-käyttöliittymän UserForm_Initialize-makro

Käyttöliittymän alustusmakro sisältää normaalien sijainnin ja koon määrittämisen lisäksi päästökategoriat sisältävän pudotusvalikon eli ComboBox-tietojen alustuksen. Tämä tehdään muuttujien määrityksen jälkeen For Each -silmukan avulla, jossa kaikki työkirjan nimetyt

alueet käydään yksitellen läpi. Mikäli nimetty alue sijaitsee Päästökertoimet-välilehdellä ja sen nimen kahdeksantoista ensimmäistä merkkiä ovat "Päästökertoimet!PK", jolloin tämä nimi viittaa Päästökertoimet-välilehdellä johonkin päästökategorian otsikkoon, lisätään alueen solun sisältämä teksti pudotusvalikkoon. Tällä tavalla saadaan muodostettua pudotusvalikkoon ajantasainen lista kaikista mahdollisista päästökategorioista, vaikka niitä olisi tietokannasta poistunut tai tullut lisää. Tämän jälkeen vielä muutetaan käyttöliittymän ulkoasua ja toimintoja aktiivisena olevan välilehden mukaisesti. Esimerkiksi valintaruutu kotimaisille päästökertoimille asetetaan näkyville ainoastaan raaka-aineiden tapauksessa ja Lisää päästökerroin taulukkoon-painike poistetaan käytöstä Päästökertoimet-välilehdellä.

Alustusmakron jälkeen käsitellään ensimmäisen pudotusvalikon muuttumiseen reagoiva ComboBox1-makro. Makron koodi on nähtävissä kuvassa 34. Makron tarkoitus on päivittää toiseen pudotusvalikkoon ensimmäiseen pudotusvalikkoon valitun kategorian mukaiset päästökertoimet.

```
Private Sub ComboBox1_Change()

    Dim OnOlemassa As Boolean
    OnOlemassa = False

    Set sh = ThisWorkbook.Sheets("Päästökertoimet")
    For Each nm In Names
        If Left(CStr(nm.RefersTo), 16) = "=Päästökertoimet" And _
        Left(CStr(nm.Name), 18) = "Päästökertoimet!PK" Then
            If sh.Range(nm.Name).Value = ComboBox1.Value Then
                Set rngTaul = sh.Range(nm.Name).CurrentRegion
                OnOlemassa = True
            End If
        End If
    Next nm

    If OnOlemassa = False Then Exit Sub

    ComboBox2.Clear
    If ComboBox1.Value = "Raaka-aineet" Then
        CheckBox1.Visible = True
        If CheckBox1.Value = False Then
            For i = 2 To rngTaul.Rows.Count
                ComboBox2.AddItem (rngTaul(i, 2).Value & " - " & rngTaul(i, 4).Value)
            Next i
        Else
            For i = 2 To rngTaul.Rows.Count
                If rngTaul(i, 4).Value = "Suomi" Then
                    ComboBox2.AddItem (rngTaul(i, 2).Value & " - " & rngTaul(i, 4).Value)
                End If
            Next i
        End If
    Else
        CheckBox1.Visible = False
        CheckBox1.Value = False
        For i = 2 To rngTaul.Rows.Count
            ComboBox2.AddItem (rngTaul(i, 2).Value)
        Next i
    End If

End Sub
```

Kuva 34. PKHakutyokalu-käyttöliittymän ComboBox1_Change-makro

ComboBox1_Change-makron toiminta alkaa muuttujien määrittämisellä, jonka jälkeen määritetään haluttujen päästökertoimien alue Range-tyyppiseen rngTaul-muuttujaan For Each -silmukkaa hyödyntäen. Samaan tapaan kuin alustusmakrossa silmukassa käydään läpi työkirjan nimettyjä alueita ja samojen ehtojen perusteella etsitään päästökertoimien otsikkoalueita. Kun tällainen otsikkoalue löydetään, sitä verrataan ensimmäiseen pudotusvalikkoon valittuun arvoon. Jos nämä nimet täsmäävät, määritetään tämän otsikkoalueen päästökertoimien alue CurrentRegion-ominaisuuden avulla rngTaul-muuttujaan. Lisäksi OnOlemassa-muuttujan arvo muutetaan arvosta False arvoon True, jotta tiedetään alueen löytyneen. Välittömästi For Each -silmukan jälkeen onkin lyhyt ehtorakenne, joka lopettaa makron toiminnan heti, mikäli aluetta ei löytynyt.

Tämän jälkeen toisen pudotusvalikon tiedot tyhjennetään siltä varalta, että siellä on valmiiksi tallennettuna vanhoja tietoja. Kun tiedot on tyhjennetty, aloitetaan uusien tietojen lisääminen. Päästökertoimien lisääminen pudotusvalikkoon toteutetaan suhteellisen yksinkertaisella For-silmukalla, jossa käydään rngTaul-muuttujaan tallennettua aluetta läpi rivi kerrallaan, ja lisätään päästökertoimet. Tässä kohtaa ohjelma myös hyödyntää ehtorakennetta, sillä raaka-aineiden tapauksessa tulee ottaa myös huomioon, onko Näytä vain kotimaiset päästökertoimet -valintaruutu valittuna. Tällöin listaan lisätään vain ne rivit, joiden maatiedoissa lukee ”Suomi”. Raaka-aineiden tapauksessa myös lisätään maatieto pudotusvalikkoon vietävien päästökertoimien nimien perään.

Kun käyttäjä valitsee toisesta pudotusvalikosta haluamansa päästökertoimen, aktivoituu ComboBox2_Change-makro. Tämän makron tehtävänä on käyttäjän valinnan perusteella lisätä käyttöliittymän ”Päästökerroin:” tekstin viereen valitun päästökertoimen arvo. Kuvassa 35 on esitetty kuvakaappaus kyseisen makron koodista.

```
Private Sub ComboBox2_Change()

    Set sh = ThisWorkbook.Sheets("Päästökertoimet")
    For Each nm In Names
        If Left(CStr(nm.RefersTo), 16) = "=Päästökertoimet" And _
        Left(CStr(nm.Name), 18) = "Päästökertoimet!PK" Then
            If sh.Range(nm.Name).Value = ComboBox1.Value Then
                Set rngTaul = sh.Range(nm.Name).CurrentRegion
            End If
        End If
    Next nm

    If ComboBox1.Value = "Raaka-aineet" Then
        For i = 1 To rngTaul.Rows.Count
            If (rngTaul(i, 2).Value & " - " & rngTaul(i, 4).Value) = ComboBox2.Value Then
                Label4.Caption = rngTaul(i, 3).Value
            End If
        Next i
    Else
        For i = 1 To rngTaul.Rows.Count
            If rngTaul(i, 2).Value = ComboBox2.Value Then
                Label4.Caption = rngTaul(i, 3).Value
            End If
        Next i
    End If

End Sub
```

Kuva 35. PKHakutyokalu-käyttöliittymän ComboBox2_Change-makro.

ComboBox2_Change-makro alkaa täysin vastaavalla For Each -silmukalla, kuten Combobox1_Change-makro ja alustusmakrot. Ensin siis määritellään Päästökertoimet-välilehdeltä alue, jossa halutut päästökertoimet sijaitsevat. Tämän jälkeen ehtorakenteen avulla päätellään, onko kyseessä raaka-aineiden päästökertoimien lista vai jokin muu, jotta ohjelma osaa poimia päästökertoimen oikeasta sarakkeesta. Sopivan päästökertoimen otsikon löytyessä päästökertoimen arvo lisätään käyttöliittymään oikeaan kohtaan.

Käyttöliittymän painikkeista Ohje ja Peruuta eivät ole kovin monimutkaisia ja vastaavat makrot on jo selitetty tässä oppaassa, joten niihin ei pureuduta tässä kohtaa tarkemmin. Sen sijaan Lisää Päästökerroin taulukkoon -painikkeesta aktivoituva LisaaKerroin_Click-makro on esitetty kuvassa 36.

```
Private Sub LisaaKerroin_Click()
    'Tällä ohjelmalla etsityn päästökertoimen voi halutessaan lisätä listaan.
    ' Tällöin päästökertoimen otsikko ja itse päästökerroin liätään listan automaattisesti,
    ' ja muut tiedot jäävät käyttäjän täytettäviksi. Käytettävissä vain raaka-aineille,
    ' pakkausmateriaaleille ja jätteille.

    If VirheTarkistus = True Then
        GoTo ohitus
    End If

    If ActiveSheet.Name = "Raaka-aineet" Then
        Call cmdRivinLisaysRaa
        Call VapautaSivu("Raaka-aineet")
        Range("RaakaaineetYhteensa").Offset(-1, 0).Value = ComboBox2.Value
        Range("RaakaaineetYhteensa").Offset(-1, 3).Value = CSng(Label4.Caption)
        Call SuojaaSivu("Raaka-aineet")
    ElseIf ActiveSheet.Name = "Pakkausmateriaalit" Then
        Call cmdRivinLisaysPakkaus
        Call VapautaSivu("Pakkausmateriaalit")
        Range("PakkauksetYhteensa").Offset(-1, 0).Value = ComboBox2.Value
        Range("PakkauksetYhteensa").Offset(-1, 5).Value = CSng(Label4.Caption)
        Call SuojaaSivu("Pakkausmateriaalit")
    ElseIf ActiveSheet.Name = "Jätteet ja sivuvirrat" And Not _
    ActiveSheet.Shapes("PKHakuJAtNappi").Fill.ForeColor.RGB = RGB(218, 217, 215) Then
        Call cmdRivinLisaysJat
        Call VapautaSivu("Jätteet ja sivuvirrat")
        Range("JatteetYhteensa").Offset(-1, 0).Value = ComboBox2.Value
        Call VapautaSivu("Jätteet ja sivuvirrat")
        Range("JatteetYhteensa").Offset(-1, 4).Value = CSng(Label4.Caption)
        Call SuojaaSivu("Jätteet ja sivuvirrat")
    ElseIf ActiveSheet.Name = "Jätteet ja sivuvirrat" And Not _
    ActiveSheet.Shapes("PKHakuPesNappi").Fill.ForeColor.RGB = RGB(218, 217, 215) Then
        Call cmdRivinLisaysPes
        Call VapautaSivu("Jätteet ja sivuvirrat")
        Range("PesuAineetYhteensa").Offset(-1, 0).Value = ComboBox2.Value
        Call VapautaSivu("Jätteet ja sivuvirrat")
        Range("PesuAineetYhteensa").Offset(-1, 3).Value = CSng(Label4.Caption)
        Call SuojaaSivu("Jätteet ja sivuvirrat")
    Else

    End If

    Unload Me

ohitus:

End Sub
```

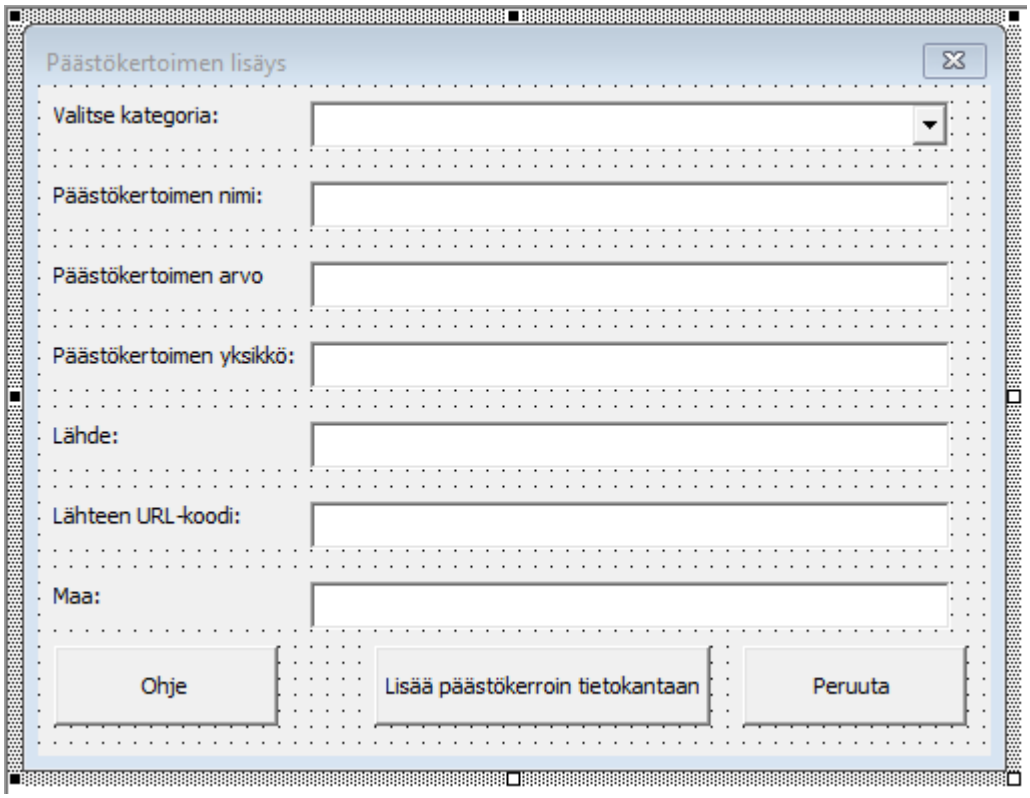
Kuva 36. PKHakutyokalu-käyttöliittymän LisaaKerroin_Click-makro.

LisaaKerroin_Click-makron suoritus alkaa käyttäjän syötteiden tarkistamisella virheiden varalta erillisellä VirheenTarkistus-makrolla. Mikäli virhe löydetään, ohjelman suoritus ohitetaan, mutta muutoin jatketaan normaalisti. Makro selvittää ensin ehtorakenteen avulla, millä välilehdellä ollaan ja mihin taulukkoon päästökerrointa ollaan lisäämässä. tämän jälkeen kutsutaan sopivaa rivin lisäysmakroa, vapautetaan oikea välilehti, lisätään päästökerroin sekä nimi ja suojataan sivu. Ehtorakenteen jälkeen käyttöliittymä suljetaan.

4.2.5 Päästökertoimen lisäystyökalu

Toisin kuin päästökertoiminen hakutyökalu, päästökertoimen lisäystyökalu on käytössä ainoastaan Päästökertoimet-välilehdellä. Sen tarkoituksena on mahdollistaa käyttäjälle omien päästökertoimien lisääminen tietokantaan siten, että päästökerroin menee varmasti oikeaan paikkaan oikealla tavalla ilman pelkoa tietokannan rikkoutumisesta.

PKLisays-käyttöliittymässä kysytään käyttäjältä pudotusvalikon avulla haluttua päästökategoriaa, minne päästökerroin halutaan lisätä. Käyttäjän tulee lisäksi kirjoittaa tekstikenttiin päästökertoimen nimi, arvo, yksikkö, lähdeviite, lähteen URL-koodi, sekä raaka-aineiden tapauksessa maa. Käyttöliittymän alalaidasta löytyvät painikkeet ohjeen avaamiselle, päästökertoimen lisäämiselle ja peruuttamiselle. Kuvakaappaus käyttöliittymästä on nähtävissä kuvassa 37.



Kuva 37. PKLisays-käyttöliittymä VBE:ssä.

Käyttöliittymän toiminta on jaoteltu kuuteen makroon: alustusmakroon, virheidenkäsittelyyn, pudotusvalikon muutokseen reagoivaan makroon sekä kolmeen painikkeeseen reagoiviin makroihin. Käydään ensin läpi alustusmakron toiminta, jonka koodi on esitelty kuvassa 38.

```
Private Sub UserForm_Initialize()

    Dim nm As Name
    Dim sh As Worksheet
    On Error Resume Next

    Set sh = ThisWorkbook.Sheets("Päästökertoimet")

    For Each nm In Names
        If Left(CStr(nm.RefersTo), 16) = "=Päästökertoimet" And _
        Left(CStr(nm.Name), 18) = "Päästökertoimet!PK" Then
            ComboBox1.AddItem (sh.Range(nm.Name).Value)
        End If
    Next nm

    On Error GoTo 0

    Label7.Visible = False
    TextBox6.Visible = False

    With PKLisays
        .Height = 279.5
        .Width = 368
    End With

    Me.Top = Application.Top + (Application.UsableHeight / 2) - (Me.Height / 2)
    Me.Left = Application.Left + (Application.UsableWidth / 2) - (Me.Width / 2)

    TextBox1.SetFocus

End Sub
```

Kuva 38. PKLisays-käyttöliittymän UserForm_Initialize-makro.

PKLisays-käyttöliittymän alustusmakro on hyvin samankaltainen päästökertoimien hakutyökalun makroista löytyvien ohjelmien kanssa. Muuttujien määritysten jälkeen lisätään pudotusvalikkoon päästökategorioiden otsikot tuttuun tapaan For Each -silmukalla, jonka jälkeen piilotetaan alustavasti käyttöliittymän alin kysymys maatietoon liittyen. Lopuksi vielä suoritetaan normaalit käyttöliittymän kokoon ja sijaintiin liittyvät toimenpiteet sekä asetetaan fokus ensimmäiseen tekstiruutuun.

Käyttöliittymän pudotusvalikon eli ComboBox1 muuttamisesta aktivoituva ComboBox1_Change-makro on toiminnaltaan hyvin yksinkertainen, sillä sen ainoa tehtävä on tarkistaa, onko pudotusvalikkoon valittu vaihtoehto ”Raaka-aineet”. Jos on, muutetaan alustusmakrossa piilotettu maatietoa kysyvä osio näkyväksi käyttäjälle. Kuvassa 39 on nähtävissä kuvakaappaus käyttöliittymän ComboBox1_Change-makrosta.

```
Private Sub ComboBox1_Change()

    If ComboBox1.Value = "Raaka-aineet" Then
        Label7.Visible = True
        TextBox6.Visible = True
    Else
        Label7.Visible = False
        TextBox6.Visible = False
        TextBox6.Value = ""
    End If

End Sub
```

Kuva 39. PKLisays- käyttöliittymän ComboBox1_Change-makro.

Päästökertoimen lisäystyökalun syötteiden oikeellisuuden varmistamiseksi käyttöliittymän taustalta löytyy oma VirheTarkistus-funktio, joka tarkistaa kaikki käyttäjän käyttöliittymälle antamat syötteet tyypillisimpien virheiden varalta. Funktiolle määritellään alustava arvo False, joka muutetaan funktion toiminnan aikana arvoon True, mikäli virhe löydetään. Koska käyttöliittymän syötteissä saattaa olla useampia erilaisia virheitä, koostetaan VirheIlmoitus-muuttujaan kaikki löydetyt virheet.

Varsinainen virheiden etsintä suoritetaan ehtorakenteella, joka käy käyttöliittymän osiot yksi kerrallaan läpi. Näistä testataan, onko tekstikenttään annettu mitään syötettä ja onko syöte annettu oikeassa muodossa (numeerinen tai ei numeerinen). Mikäli virhe löydetään, värjätään kyseisen kentän tausta keltaiseksi, lisätään VirheIlmoitus-muuttujaan sopiva lisäys ja muutetaan Virhe muuttujan arvoksi True. Ehtorakenteen jälkeen funktio antaa käyttäjälle

koostetun virheilmoituksen, mikäli virheitä on löytynyt. Kuva funktion koodista on nähtävissä kuvassa 40.

```
Private Function VirheTarkistus() As Boolean

    Dim VirheIlmoitus As String
    Dim Virhe As Boolean

    VirheTarkistus = False
    Virhe = False
    VirheIlmoitus = "Syötteissä ilmeni seuraavia virheitä:"

    If ComboBox1.Value = "" Then
        VirheIlmoitus = VirheIlmoitus & Chr(13) & Chr(13) & _
        "Päästökerroinkategoriaa ei ole valittu tai siinä on virhe."
        ComboBox1.BackColor = RGB(255, 255, 0)
        VirheTarkistus = True
        Virhe = True
    Else
        ComboBox1.BackColor = RGB(255, 255, 255)
    End If
    If TextBox1.Value = "" Or IsNumeric(TextBox1.Value) = True Then
        VirheIlmoitus = VirheIlmoitus & Chr(13) & Chr(13) & _
        "Päästökertoimen nimeä ei ole syötetty tai nimi on virheellinen"
        TextBox1.BackColor = RGB(255, 255, 0)
        VirheTarkistus = True
        Virhe = True
    Else
        TextBox1.BackColor = RGB(255, 255, 255)
    End If
    If TextBox2.Value = "" Or IsNumeric(TextBox2.Value) = False Then
        VirheIlmoitus = VirheIlmoitus & Chr(13) & Chr(13) & _
        "Päästökertoimen arvo on virheellinen."
        TextBox2.BackColor = RGB(255, 255, 0)
        VirheTarkistus = True
        Virhe = True
    Else
        TextBox2.BackColor = RGB(255, 255, 255)
    End If
    If TextBox3.Value = "" Or IsNumeric(TextBox3.Value) = True Then
        VirheIlmoitus = VirheIlmoitus & Chr(13) & Chr(13) & _
        "Päästökertoimen yksikkö on virheellinen."
        TextBox3.BackColor = RGB(255, 255, 0)
        VirheTarkistus = True
        Virhe = True
    Else
        TextBox3.BackColor = RGB(255, 255, 255)
    End If
    If TextBox4.Value = "" Or IsNumeric(TextBox4.Value) = True Then
        VirheIlmoitus = VirheIlmoitus & Chr(13) & Chr(13) & _
        "Lähdettä ei ole syötetty tai se on virheellinen"
        TextBox4.BackColor = RGB(255, 255, 0)
        VirheTarkistus = True
        Virhe = True
    Else
        TextBox4.BackColor = RGB(255, 255, 255)
    End If
    If TextBox5.Value = "" Then
        VirheIlmoitus = VirheIlmoitus & Chr(13) & Chr(13) & _
        "Lähteen linkkiä ei ole syötetty tai se on virheellinen."
        TextBox5.BackColor = RGB(255, 255, 0)
        VirheTarkistus = True
        Virhe = True
    Else
        TextBox5.BackColor = RGB(255, 255, 255)
    End If

    If Virhe = True Then
        MsgBox VirheIlmoitus, vbOKOnly, "Virheilmoitus"
    End If

End Function
```

Kuva 40. PKLisays-käyttöliittymän VirheTarkistus-makro.

Käyttöliittymän painikkeista Peruuta- ja Ohje-painikkeiden aktivoivia makroja ei käydä yksinkertaisuutensa takia läpi tarkemmin, mutta Lisää päästökerroin tietokantaan-painikkeen toiminta selitetään seuraavaksi. Makron koodi on esitetty kuvassa 41.

Lisaa_Click-makron toiminta alkaa muuttujien määrityksellä, jonka jälkeen käyttäjän syötteet tarkistetaan VirheTarkistus-funktion avulla. Mikäli virheitä ei havaita, siirrytään tuttuun For Each -silmukkaan, jossa määritellään rngOtsikko-muuttujaan alue oikean päästökategorian otsikon sijainnista. Tämän jälkeen tätä aluetta käytetään uuden päästökertoimen lisäämiseksi halutun listan pohjalle .End(xlDown)-ominaisuutta hyödyntäen. Myös lähteen hyperlinkki liitetään .Hyperlinks.Add-metodia käyttäen. Lopuksi vielä muotoillaan tekstien fonttia sopivammiksi ennen kuin käyttöliittymä suljetaan.

```
Private Sub Lisaa_Click()

    Dim sh As Worksheet
    Dim nm As Name
    Dim rngOtsikko As String

    If VirheTarkistus = True Then
        GoTo ohitus
    End If

    Set sh = ThisWorkbook.Sheets("Päästökertoimet")
    For Each nm In Names
        If Left(CStr(nm.RefersTo), 16) = "=Päästökertoimet" And _
        Left(CStr(nm.Name), 18) = "Päästökertoimet!PK" Then
            If sh.Range(nm.Name).Value = ComboBox1.Value Then
                rngOtsikko = nm.Name
            End If
        End If
    Next nm

    With sh.Range(rngOtsikko).End(xlDown)
        .Offset(1, 0).EntireRow.Insert
        .Offset(1, -1).Value = TextBox4.Value
        .Offset(1, 0).Value = TextBox1.Value
        If ComboBox1.Value = "Raaka-aineet" Then
            .Offset(1, 2).Value = TextBox6.Value
            .Offset(1, 1).Value = CSng(TextBox2.Value)
            .Offset(1, 3).Value = TextBox3.Value
        Else
            .Offset(1, 1).Value = TextBox2.Value
            .Offset(1, 2).Value = TextBox3.Value
        End If
    End With

    sh.Hyperlinks.Add Range(rngOtsikko).End(xlDown).Offset(0, -1), _
    Address:=TextBox5.Value

    With sh.Range(rngOtsikko).End(xlDown)
        .Offset(0, -1).Font.Color = RGB(0, 0, 0)
        .Offset(0, -1).Font.Underline = False
        .Offset(0, -1).Font.Size = sh.Range(rngOtsikko).End(xlDown).Font.Size
        .Offset(0, -1).Font.Name = sh.Range(rngOtsikko).End(xlDown).Font.Name
    End With

    Unload Me

ohitus:

End Sub
```

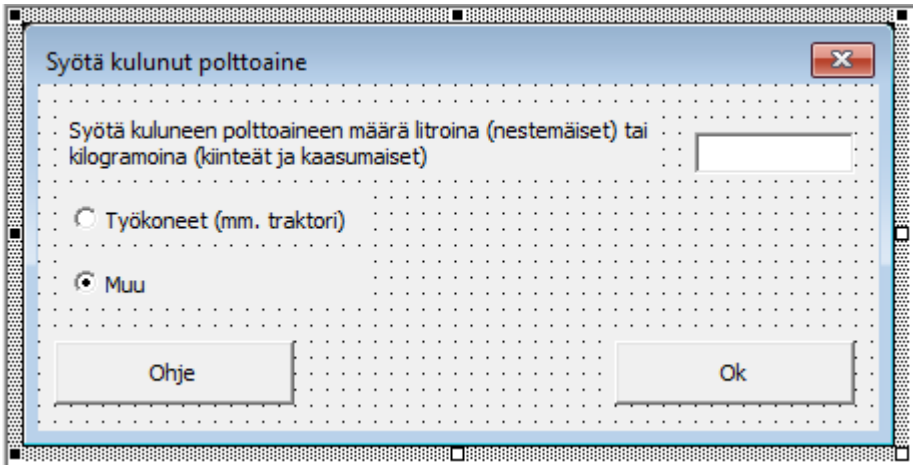
Kuva 41. PKLisays-käyttöliittymän Lisaa_Click-makro.

4.2.6 Muut käyttöliittymät

Monimutkaisempien käyttöliittymien ohella IKE-hiilijalanjälkilaskuri käyttää muutamaa yksinkertaisempaa käyttöliittymää. Etusivulta aktivoituva ”Pikaohje” tuodaan käyttöliittymän avulla, koska siinä on liikaa merkkejä VBA:n omalle MsgBox-funktiolle. PikaOhje-käyttöliittymä sisältääkin ainoastaan alustusmakron, jolla käyttöliittymän koko ja sijainti säädetään oikein.

Samaan tapaan OdotusIkkuna-käyttöliittymä ainoastaan avautuu raportin tulostustoiminnon suorituksen ajaksi näkyville ja sulkeutuu ohjelman valmistuessa. Tässä tapauksessa käyttöliittymää käytettiin MsgBox-funktion sijaan siitä syystä, että se voidaan sulkea koodin avulla automaattisesti Unload OdotusIkkuna-komennolla ilman käyttäjän klikkausta toisin kuin MsgBox.

Viimeisenä ”yksinkertaisena” käyttöliittymänä IKE-hiilijalanjälkilaskuri sisältää ajoneuvojen polttoaineen määrän syöttämiseen manuaalisesti tarkoitetun PolttoaineenSyotto-käyttöliittymän. Tämä luotiin siksi, että VBA:n oma InputBox-funktio ei ihan ollut riittävä tarpeellisten tietojen kysymiseen käyttäjältä. Käyttöliittymässä halutaan kysyä käyttäjältä polttoaineen kulutuksen lisäsi, onko kyseessä työkone vai jokin muu ajoneuvo. Kuvassa 42 on esitetty kuvakaappaus kyseisen käyttöliittymän koodista.



Kuva 42. PolttoaineenSyotto-käyttöliittymä VBE:ssä.

Käyttöliittymän koodi pitää sisällään ainoastaan neljä makroa; alustusmakron, virhekäsittelijän ja makrot painikkeiden klikkauksille. Alustusmakro UserForm_Initialize ei sisällä

käyttöliittymän koon ja sijainnin määrityksen lisäksi muita toimintoja, Ohje-painikkeesta aktivoituva makro ainoastaan kutsuu ohjeen avausmakroa ja VirheKasittely-funktio käy läpi hyvin yksinkertaisella ehtorakenteella tyypillisimmät virheet käyttäjän syötteestä tekstikenttään. Ok-painikkeesta aktivoituvan CommandButton2_Click-makron koodi on nähtävissä kuvassa 43.

```
Private Sub CommandButton2_Click()

    If VirheKasittely = True Then
        Exit Sub
    End If

    With Range("LogistiikkaYhteensa").Offset(-1, 0)
        If .Value = "" Then
            If OptionButton2 Then
                .Value = "Muu"
            Else
                .Value = "Työkoneet (mm. traktori)"
            End If
            .Offset(0, 7).Value = TextBox1.Value
            .Offset(0, 2).Value = 100
        Else
            Call cmdRivinLisaysLog
            If OptionButton2 Then
                .Offset(1, 0).Value = "Muu"
            Else
                .Offset(1, 0).Value = "Työkoneet (mm. traktori)"
            End If
            .Offset(1, 7).Value = TextBox1.Value
            .Offset(1, 2).Value = 100
        End If
    End With

    Unload Me

End Sub
```

Kuva 43. PolttoaineenSyotto käyttöliittymän CommandButton2_Click-makro

CommandButton2_Click-makrossa suoritetaan ensin virheiden käsittely omalla funktiollaan, jonka jälkeen ehtorakenteella selvitetään, onko ajoneuvojen taulukossa jo valmiina tyhjää riviä. Jos on, lisätään tarvittavat tiedot sille, jotta käyttäjän pyytämä polttoaineen kulutus täyttyy. Jos taas riviä ei valmiina ole, lisätään se ensin ja täytetään tiedot vasta sitten. Tämän jälkeen käyttöliittymä suljetaan.

4.3 Moduulit

Moduulit sisältävät suurimman osan laskuria varten tehdyistä makroista. Moduulit on jaettu sen mukaan, minkä tyyppisiä tai mihin asiaan liittyviä makroja ne sisältävät. Toiset makrot ovat suunniteltu hyvin spesifejä käyttötarkoituksia varten, kun taas toiset ovat hyvin monikäyttöisiä. Esimerkiksi energiamoduulin makroja käytetään vain tiettyihin yksittäisiin toimintoihin tietyissä paikoissa, kun taas tiettyjä suojausmoduulin makroja kutsutaan lähes kaikista muista makroista.

4.3.1 Energia

Energiamoduuli sisältää laskurin Energia-välilehden toiminnan kannalta oleellisia makroja. Makroja on tässä moduulissa yhteensä kahdeksan, ja ne ovat energiajakauman hallinnointiin liittyvät OmaEnergiajakauma_Napsauta, imgKäytäOmaaPäästökerrointa_Napsauta ja imgKäytäOletusarvoja_Napsauta, tietojen syöttämiseen liittyvät cmdSyotakWh ja cmdSyotaPolttoaineMuutLaitteet sekä Energia-välilehden Worksheet_Change-makron kautta kutsuttavina makroina toimivat MuutLaitteet(rngKohde As Range), EnergiaJakauma(rngKohde As Range).

Energiajakauman hallinnoinnin kolme makroa liittyvät energiavälilehden toimintoon, jossa käyttäjä voi valita käytetyn sähköenergian päästökertoimen muodostamisperiaatteen kolmesta eri vaihtoehdosta: Käytä omaa energiajakaumaa, Käytä omaa päästökerrointa ja Käytä oletusarvoja. Kuvassa 44 on esitetty kuvakaappaus OmaEnergiaJakauma_Napsauta -makrosta, joka aktivoituu, kun käyttäjä klikkaa Käytä omaa energiajakaumaa -painiketta.

```
Public Sub OmaEnergiajakauma_Napsauta()
    ' Tämä makro mahdollistaa käyttäjälle oman energiajakauman luomisen mm. sähköyhtiön
    ' ilmoittaman energiajakauman perusteella.

    Call VapautaSivu("Energia")
    Application.EnableEvents = False
    ThisWorkbook.Sheets("Yhteenveto").ChartObjects("EnergiajakaumaPiirakkaKaavio").Visible = True

    'Määritellään muuttujat
    Dim ylanurkka As Range
    Dim alanurkka As Range
    Dim EnergiaJakauma As Range

    ' Asetetaan muuttujiin tarvittavat alueet. Käytännössä Määritetään energiajakaumaa koskevan
    ' listan alue piilottelua ja paljastamista varten.
    Set ylanurkka = Range("Energianlahde").Offset(-1, 3)
    Set alanurkka = Range("KokPaastokerroin").Offset(-1, 3)
    Set EnergiaJakauma = Range(ylanurkka, alanurkka)

    ' Paljastetaan näkyviin energiajakaumaa käsittelevä taulukko ja nappulat.
    EnergiaJakauma.EntireRow.Hidden = False
    ActiveSheet.Shapes("EJakPoistoNappi").Visible = True
    ActiveSheet.Shapes("EJakLisaysNappi").Visible = True

    ' Muutetaan alarivin teksti
    Range("KokPaastokerroin").Value = "Päästökerroin annetulla energiajakaumalla:"

    ' Muutetaan valintanappuloiden värit siten, että valittu nappula on eri värinen,
    ' kuin kaksi ei-valittua.
    ActiveSheet.Shapes("imgKäytäOmaaEnergiajakaumaa").Fill.ForeColor.RGB = RGB(251, 173, 24)
    ActiveSheet.Shapes("imgKäytäOmaaPäästökerrointa").Fill.ForeColor.RGB = RGB(32, 23, 81)
    ActiveSheet.Shapes("imgKäytäOletusarvoja").Fill.ForeColor.RGB = RGB(32, 23, 81)
    ActiveSheet.Shapes("imgKäytäOmaaEnergiajakaumaa").TextFrame.Characters.Font.Color = RGB(0, 0, 0)
    ActiveSheet.Shapes("imgKäytäOmaaPäästökerrointa").TextFrame.Characters.Font.Color = RGB(255, 255, 255)
    ActiveSheet.Shapes("imgKäytäOletusarvoja").TextFrame.Characters.Font.Color = RGB(255, 255, 255)

    ' Lisätään päästökertoimen ja kokonaisprosentin soluihin oikeat summakaavat.
    With Range("KokPaastokerroin")
        .Offset(0, 4).Formula = "=SUM(" & SarakeKirjain(Range("Energianlahde").Column + 4) & _
        Range("Energianlahde").row + 1 & ":" & SarakeKirjain(Range("Energianlahde").Column + 4) & _
        Range("KokPaastokerroin").row - 1 & ")"
        .Offset(0, 5).Formula = "=SUM(" & SarakeKirjain(Range("Energianlahde").Column + 5) & _
        Range("Energianlahde").row + 1 & ":" & SarakeKirjain(Range("Energianlahde").Column + 5) & _
        Range("KokPaastokerroin").row - 1 & ")"
        .Offset(0, 1).Formula = "=SUM(" & SarakeKirjain(Range("Energianlahde").Column + 1) & _
        Range("Energianlahde").row + 1 & ":" & SarakeKirjain(Range("Energianlahde").Column + 1) & _
        Range("KokPaastokerroin").row - 1 & ")"
        .Offset(0, 4).Font.Color = RGB(0, 0, 0)
        .Offset(0, 5).Font.Color = RGB(0, 0, 0)
    End With

    Call SuojaaSivu("Energia")
    Application.EnableEvents = True

End Sub
```

Kuva 44. Oman energiajakauman syöttämiseen liittyvän painikkeen makro.

Kuvan 44 mukaisen makron tarkoituksena on siis muuttaa valinnan mukaisesti painikkeiden värit sekä paljastaa täytettävä taulukko ja sen painikkeet käyttäjälle. Aluksi makrossa vapautetaan sivu ja estetään Worksheet_Change-makron tarpeeton aktivoituminen asettamalla Application.EnableEvents-asetus pois päältä. Tämän jälkeen määritellään yhteenvetosivun energiajakaumaa kuvaava kaavio näkyväksi, ja sen jälkeen määritellään tarvittavat muuttujat. Muuttujien määrityksen jälkeen paljastetaan energiajakaumaan liittyvät taulukot ja painikkeet,

muutetaan alarivin teksti ja muutetaan painikkeiden värit sen mukaan, mikä vaihtoehto on valittuna. Lopuksi vielä lisätään päästökertoimien ja kokonaisprosentin soluihin alariville oikeat summakaavat.

Edellisestä makrosta voi huomata, että kaavojen asettamiseen liittyvät koodirivit vaikuttavat tarpeettoman pitkiltä ja monimutkaisilta. Tämä johtuu siitä, että koodi on muodostettu ottamaan huomioon energiajakauman taulukon mahdollinen siirtyminen. Makro siis toimii täydellisesti, vaikka energiajakauman taulukko siirrettäisiin täysin eri kohtaan samalla välilehdellä.

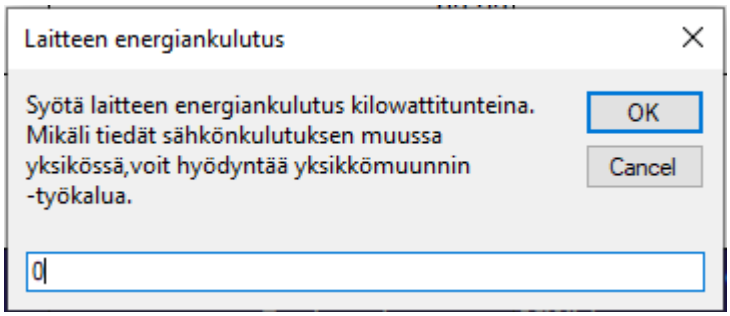
Kaksi muuta sähköenergian päästökertoimen määrittämiseen liittyvää makroa imgKäytäOmaaPäästökerrointa_Napsauta ja imgKäytäOletusarvoja_Napsauta toimivat hyvin pitkälti samalla periaatteella, kuten OmaEnergiaJakauma_Napsauta-makro. Oleellisena erona näissä kahdessa muussa energiajakauman taulukko piilotetaan painikkeiden ja kaavioiden ohella. Lisäksi oman päästökertoimen kanssa joudutaan kysymään kerrointa käyttäjältä ja tarkistamaan syöte virheellisten syötteiden varalta.

Seuraava energiamoduulin makro on cmdSyotakWh, joka aktivoituu energiavälilehdellä sähkölaitteiden listan yläpuolelta löytyvästä painikkeesta. Tämän makron avulla käyttäjä voi lisätä haluamansa sähkölaitteen sähkönkulutuksen suoraan kilowattitunteina, jos hän ei esimerkiksi tiedä laitteen tehoa ja käyttöaikaa. Kuvakaappaus makrosta on esitetty kuvassa 45.

```
Public Sub cmdSyotakWh()

    Call VapautaSivu("Energia")

    Dim syöte As String
    Dim message As String
    Dim title As String
    Dim kWh As Double

    Call NapinKlikkaus1(ActiveSheet.Shapes("LisaakWhNappi"))

    message = "Syötä laitteen energiankulutus kilowattitunteina. Mikäli tiedät sähkönkulutuksen muussa yksikössä," & _
              "voit hyödyntää yksikkömuunnin -työkalua."
    title = "Laitteen energiankulutus"

    syöte = InputBox(message, title, 0)
    If StrPtr(syöte) = 0 Then
        GoTo Loppu
    ElseIf syöte = vbNullString Then
        MsgBox ("Et syöttänyt kelvollista arvoa.")
        GoTo Loppu
    ElseIf IsNumeric(syöte) = False Then
        MsgBox ("Et syöttänyt kelvollista arvoa.")
        GoTo Loppu
    ElseIf syöte <= 0 Then
        MsgBox ("Et syöttänyt kelvollista arvoa.")
        GoTo Loppu
    Else
        kWh = CDbl(syöte)
    End If

    With Range("SahkolaitteetYhteensa").Offset(-1, 0)
        If .Value = "" And .Offset(0, 1).Value = "" And .Offset(0, 2).Value = "" Then
            .Offset(0, 1).Value = kWh
            .Offset(0, 2).Value = 60
        Else
            Call cmdRivinLisaysEner
            .Offset(1, 1).Value = kWh
            .Offset(1, 2).Value = 60
        End If
    End With

Loppu:
    Call NapinKlikkaus2(ActiveSheet.Shapes("LisaakWhNappi"))
    Call SuojaaSivu("Energia")

End Sub
```

Kuva 45. Sähkölaitteen kuluttaman kilowattituntien määrän lisäysmakro.

Yllä olevan makron perusidea on syöttää sähkölaitteiden taulukkoon sellaiset tiedot, joilla käyttäjän syöttämä sähkönkulutus täyttyy. Aluksi määritellään tarvittavat muuttujat, jonka jälkeen kutsutaan napin klikkauksen väritehosteita varten erillistä makroa. Tämän jälkeen määritellään String-muuttujiin tekstit, joita käytetään InputBox:issa kysyttäessä käyttäjältä sähkönkulutusta. Tämän jälkeen määritellään syöte-muuttujaan arvo InputBox-metodin avulla. InputBox on VBA:n metodi, joka luo automaattisesti yksinkertaisen käyttöliittymän, jolle käyttäjä voi syöttää tietoja. Kuvassa 46 on kuvakaappaus kyseisestä InputBox:sta.



Kuva 46. Laitteen energiankulutuksen lisäämistä varten tarkoitettu InputBox-ikkuna.

InputBox-metodin huonoja puolia ovat esimerkiksi, että se ei ole visuaalisesti kovin kaksinen käyttöliittymä ja siihen virheenkäsittelyn koodaaminen on jostain syystä hankalaa verrattuna itse luotuun käyttöliittymään. Tästä syystä koodissa selvitetään seuraavaksi virheenkäsittelyn keinoin, onko käyttäjä syöttänyt arvoksi nollan, onko käyttäjä jättänyt syötteen tyhjäksi, onko käyttäjä painanut ikkunan sulkemispainiketta tai Cancel-painiketta, onko syöte jotain muuta, kuin numeerinen arvo ja onko syötetty arvo negatiivinen. Jos syöte on positiivinen numero, syötetään taulukkoon tarvittavat arvot. Energia-moduulista löytyy lähes täysin vastaava makro: cmdSyotaPolttoaineMuutLaitteet muiden laitteiden polttoaineen syöttämiselle suoraan litroissa tai kilogrammoissa.

Viimeiset kolme makroa energiamoduulissa ovat oikeiden päästökertoimien määrittämistä varten, ja niitä kutsutaan energiavälilehden Worksheet_Change(ByVal Target As Range)-makrosta, josta on selitetty yleisesti kappaleessa 4.1.3. Kuvassa 47 on kuvakaappaus EnergiaJakauma(rngKohde As Range) -makrosta, jonka avulla päivitetään päästökertoimia ja tarkistetaan prosenttiosuuksien oikeellisuutta käyttäjän muutoksien mukaisesti.

```
Sub EnergiaJakauma(rngKohde As Range)

    Dim shSheetKertoimet As Worksheet
    Dim rngLista As Range
    Dim icol As Integer
    Dim irow As Integer

    Call VapautaSivu("Energia")
    Call VapautaSivu("Päästökertoimet")

    Set shSheetKertoimet = ThisWorkbook.Sheets("Päästökertoimet")
    Set rngLista = shSheetKertoimet.Range("PKSahko").CurrentRegion 'Energiamuotojen pääästökertoimet.

    Application.EnableEvents = False
    Application.ScreenUpdating = False

    On Error Resume Next

     ' Mikäli muokattu solu oli energianlähde, haetaan valittua energiamuotoa vastaava päästökerroin
     '"Päästökertoimet" -välilehdeltä oikeasta kohtaa.
     If rngKohde.Column = Range("Energianlahde").Column Then
         icol = rngLista.Columns(2).Column
         For irow = 1 To rngLista.Rows(rngLista.Rows.Count).row
             If rngLista(irow, icol) = rngKohde Then
                     rngKohde.Offset(0, 2).Value = rngLista(irow, icol + 1) 'Päästökerroin
                     rngKohde.Offset(0, 3).Value = rngLista(irow, icol + 5) 'Biogeenisten päästöjen osuus
             End If
         Next irow
     ' Mikäli muokattu solu oli prosentiosuus, tarkistetaan, ettei osuuksien summa ylitä 100%. Mikäli näin
     ' on päässyt käymään, annetaan tästä virheilmoitus ja korjataan osuudeksi 0%.
     ElseIf rngKohde.Column = Range("Energianlahde").Column + 1 Then
         If Range("KokPaastokerroin").Offset(0, 1).Value > 1 Then
             MsgBox ("Energiajakauman yhteenlasketut osuudet ovat yli 100 prosenttia. Korjaa osuudet.")
             rngKohde.Value = 0
         End If
     End If

    On Error GoTo 0

    Call SuojaaSivu("Energia")
    Call SuojaaSivu("Päästökertoimet")

    Application.EnableEvents = True
    Application.ScreenUpdating = True

End Sub
```

Kuva 47. Energiajakauman päästökertoimista vastaava makro.

Yllä esitetty makro alkaa muuttujien määrittämisellä, sivujen suojauksien vapauttamisella, ja asetusten muuttamisella. Näiden jälkeen käytetään komentoa On Error Resume Next. Tämä komento aiheuttaa sen, että ohjelman normaalisti pysähtyessä virheeseen, ohjelma jatkaa automaattisesti suorittamista seuraavalta riviltä tästä välittämättä. Tämä ei yleisesti ole kovin suositeltava metodi, mutta tässä tapauksessa sitä uskalletaan käyttää, koska tiedetään varmuudella koodin tietyissä tapauksissa aiheuttavan virheen, joka ei kuitenkaan voi aiheuttaa vahinkoa laskurille. Tämä mahdollinen virhe voi sattua seitsemän riviä alempana kohdassa If rngLista(irow, icol) = rngKohde Then. Tällä rivillä on mahdollista, että vertailuoperaattorin eri puolten tietotyypit eivät täsmää. Tällöin ohjelma normaalisti pysähtyisi Type mismatch-tyyppiseen erroriin, mutta On Error Resume Next komennon ansiosta ohjelma ei tästä käyttäjälle ilmoita ja jatkaa eteenpäin. Kun tietotyypit eivät vastaa toisiaan, ei myöskään ehtorakenteen vaatima ehto toteudu ja näin ollen virheen ohittaminen ei aiheuta vahinkoa.

Virheen ohituksen jälkeen makrossa selvitetään ehtorakenteella, onko muokattu alue energianlähdesarakkeessa, prosenttiosuussarakkeessa vai ei kummassakaan.

Mikäli muokattu solu oli energianlähdesarakkeessa, selvitetään For-silmukkarakenteen avulla valitulle energianlähteelle vastaavat päästökertoimet. Energianlähteet valitaan pudotusvalikosta, joka perustuu tietokannasta löytyvään listaan energianlähteistä. Silmukassa käydään siis läpi tietokannan energialähteiden listaa, ja valitaan ehtorakenteen avulla sopiva, kun tämä tulee vastaan. Energiajakauman taulukkoon lisätään tällöin vastaava päästökerroin sekä biogeenisten päästöjen prosentuaalinen osuus. Jos muokattu solu taas oli prosenttiosuussarakkeessa, ohjelma vain tarkistaa, ylittääkö prosenttiosuuksien summa 100 prosenttia. Mikäli näin on, käyttäjä saa tästä ilmoituksen ja ohjelma korjaa käyttäjän muokkaaman solun nollaksi.

Tämän jälkeen palautetaan virheenkäsittelyasetus komennolla On Error GoTo 0. Tämä on hyvin tärkeää, sillä muutoin On Error Resume Next-asetus jäisi päälle, eikä Excel jatkossa enää ilmoittaisi ohjelman virheistä. Lopuksi palautetaan välilehtien suojaukset ja palautetaan asetukset.

Kaksi muuta makroa: MuutLaitteet(rngKohde As Range) ja HoyryPK(rngKohde As Range) toimivat samalla periaatteella, kuten EnergiaJakauma(rngKohde As Range).

4.3.2 Jakelu

Jakelumoduuli sisältää nimensä mukaisesti logistiikkavälilehdelle liittyviä makroja, joita kutsutaan pääasiallisesti logistiikkavälilehden Worsheet_Change-makrosta tai moduulin sisäisistä makroista. Moduuli sisältää 5 makroa: Varastointi(rngKohde As Range), Kuljetukset(rngKohde As Range), Kylmaainekaasut(rngKohde As Range), KuljetustenPaastojenKaava(AjoneuvonTyyppi As String, rivi As Integer) As String ja tkmKValineet() As Integer. Näistä kolme ensimmäistä ovat Worsheet_Change-makrosta kutsuttavia makroja ja kaksi jälkimmäistä Kuljetukset(rngKohde As Range)-makrosta kutsuttavia makroja.

Varastointi(rngKohde As Range) on hyvin samankaltainen makro, kuten vastaavat Worksheet_Change-makrot energiamoduulissa. Samalla tyylillä tutkitaan ehtorakenteen avulla, missä muutos tehtiin ja toimitaan sen mukaisesti, kuten kuvan 48 kuvakaappauksesta voidaan nähdä. Ensimmäisessä ehtorakenteen vaiheessa etsitään tietokannasta vastaava päästökertoimen arvo, mikäli varastointilistan energiamuotoa on muutettu. Toisessa vaiheessa muutetaan laskentakaavoja varaston tyypin mukaisesti, sekä muutetaan solujen värjäyksiä sen mukaan, mitä tietoja käyttäjältä tarvitaan laskentaan.

```
Public Function Varastointi(rngKohde As Range)

    Call VapautaSivu("Logistiikka")
    Call VapautaSivu("Päästökertoimet")

    ' Määritellään muuttujat
    Dim shKertoimet As Worksheet
    Dim rngLista As Range
    Dim lrow As Long
    Dim lcol As Long
    Dim VSar As Integer
    Dim R As Integer

    Application.EnableEvents = False

    If rngKohde.Column = Range("Varastointi").Column + 9 Then

        ' Etsitään energiamuotoa muutettaessa vastaava päästökerroin Päästökertoimet-välilehdeltä.
        Set shKertoimet = ThisWorkbook.Sheets("Päästökertoimet")
        Set rngLista = shKertoimet.Range("PKSahko").CurrentRegion

        On Error Resume Next

        lcol = rngLista.Columns(2).Column
        For lrow = 2 To rngLista.Rows.Count
            If rngLista(lrow, lcol) = rngKohde.Value Then
                    rngKohde.Offset(0, 2).Value = rngLista(lrow, lcol + 1) / 1000
                    rngKohde.Offset(0, 5).Value = rngLista(lrow, lcol + 5)
                    Exit For
            End If
        Next lrow

        On Error GoTo 0
    ElseIf rngKohde.Column = Range("Varastointi").Column + 2 Then
        VSar = Range("Varastointi").Column
        R = rngKohde.row
        If rngKohde.Value = "Lämmin varasto" Then
            rngKohde.Offset(0, 2).Interior.Color = rngKohde.Interior.Color
            rngKohde.Offset(0, 3).Interior.Color = RGB(255, 255, 255)
            rngKohde.Offset(0, 8).Formula2 = "=IFERROR(" & SarakeKirjain(VSar + 8) & R & "*" & _
            SarakeKirjain(VSar + 4) & R & "*(" & SarakeKirjain(VSar + 6) & R & "/365)*(" & _
            SarakeKirjain(VSar + 3) & R & "/" & SarakeKirjain(VSar + 7) & R & ")," & Chr(34) & Chr(34) & ")"
        ElseIf rngKohde.Value = "Jäähdytetty varasto" Then
            rngKohde.Offset(0, 2).Interior.Color = RGB(255, 255, 255)
            rngKohde.Offset(0, 3).Interior.Color = rngKohde.Interior.Color
            rngKohde.Offset(0, 8).Formula2 = "=IFERROR(" & SarakeKirjain(VSar + 8) & R & "*" & _
            SarakeKirjain(VSar + 5) & R & "*(" & SarakeKirjain(VSar + 6) & R & "/365)*(" & _
            SarakeKirjain(VSar + 3) & R & "/" & SarakeKirjain(VSar + 7) & R & ")," & Chr(34) & Chr(34) & ")"
        ElseIf rngKohde.Value = "Lämmittämätön varasto" Then
            rngKohde.Offset(0, 2).Interior.Color = RGB(255, 255, 255)
            rngKohde.Offset(0, 3).Interior.Color = RGB(255, 255, 255)
            rngKohde.Offset(0, 8).Value = 0
        End If
    End If

    Application.EnableEvents = True

    Call SuojaaSivu("Logistiikka")
    Call VapautaSivu("Päästökertoimet")

End Function
```

Kuva 48. Varastoinnin päästökertoimien ja kaavojen määrittämiseen tarkoitettu makro.

Mikäli varastoinnin päästöjen laskentakaavaa halutaan muokata, täytyy muuttaa soluun rngKohde.Offset(0,8) tehtäviä muutoksia, sillä näissä .Formula2-ominaisuuden avulla muutetaan solun kaavaa. Esimerkiksi useammalle riville kirjoitetulla komennolla: rngKohde.Offset(0, 8).Formula2 = "=IFERROR(" & SarakeKirjain(VSar + 8) & R & "*" & SarakeKirjain(VSar + 4) & R & "*(" & SarakeKirjain(VSar + 6) & R & "/365)*(" &

SarakeKirjain(VSar + 3) & R & "/" & SarakeKirjain(VSar + 7) & R & ")," & Chr(34) & Chr(34) & ")" voidaan lisätä energiakulutuksen soluun kaava: =IFERROR(I18*E18*(G18/365)*(D18/H18);""). Äkkiseltään tämä vaikuttaa tarpeettoman monimutkaiselta ja sitä se tietystä näkökulmasta onkin. Kaavan muodostamiseen hyödynnetään toisesta moduulista kutsuttavaa makroa SarakeKirjain, joka muuttaa kokonaisluvun järjestysnumeroa vastaavaksi aakkoseksi. Tällä tavalla kaavaan saadaan oikeat sarakkeet, vaikka taulukkoa siirreltäisiin eri paikkaan. Lisäksi komennossa käytetään muuttujia VSar ja R, joista ensimmäinen kuvastaa taulukon vasemman reunan sarakkeen numeroa ja jälkimmäinen käsiteltävän rivin numeroa.

Kuljetukset(rngKohde As Range) ja Kylmaainekaasut(rngKohde As Range) -makrot ovat sisällöltään hieman erilaisia edelliseen verrattuna, mutta rakenteeltaan ja periaatteeltaan ne ovat hyvin samankaltaiset. Näistä kahdesta ensimmäinen on jonkin verran muita monimutkaisempi, sillä kuljetusten päästöjen laskentaan on sisällytetty useita eri vaihtoehtoisia laskentaperiaatteita. Tästä syystä kyseistä makroa on yksinkertaistettu viemällä kuljetusten päästöjen kaava omaan funktioonsa.

Kuljetusten päästöjen kaavat määrittelevä funktio on nimetty toimintansa mukaisesti: KuljetustenPaastojenKaava(AjoneuvonTyyppi As String, rivi As Integer) As String. Kuvassa 49 on esitetty kuvakaappaus kyseisestä koodista.

```
Private Function KuljetustenPaastojenKaava(AjoneuvonTyyppi As String, rivi As Integer) As String

    ' Tällä funktiolla voidaan määrittää oikea kaava kuljetusten päästöjen lakentaan kolmesta
    ' vaihtoehdosta: Ajoneuvoa ei ole valittu -> ei kaavaakaan, Ajoneuvon tyypiksi on valittu
    ' "Muu" tai "Työkoneet (mm. traktori)" -> lasketaan päästöt polttoaineen kulutukseen
    ' perustuen, Ajoneuvon tyyppi ei ole "Muu", "Työkoneet (mm. traktori)" tai tyhjä ja
    ' ajoneuvolle on saatavissa päästökertoimet tyhjälle ja täydelle kuormalle erikseen ->
    ' lasketaan päästöt kuorman tietoihin perustuen(PEF-standardin määräämä tapa) Ajoneuvon
    ' tyyppi ei ole "Muu" tai tyhjä ja ajoneuvolle ei ole saatavissa päästökertoimet tyhjälle
    ' ja täydelle kuormalle erikseen -> lasketaan tonnikilometreihin perustuvalla kaavalla.
    ' Korjataan samalla valintojen perusteella solujen värejä ja poistetaan mahdollisesti
    ' roikkumaan jääneitä ylimääräisiä tietoja.

    Call VapautaSivu("Logistiikka")

    Dim rngLista As Range
    Dim icol As Integer
    Dim irow As Integer
    Dim VSar As Integer
    Dim R As Integer
    Dim raja As Integer

    Set rngLista = ThisWorkbook.Sheets("Päästökertoimet").Range("PKAjoneuvot").CurrentRegion
    VSar = Range("LogistiikkaYhteensa").Column
    R = rivi
    raja = tkmKValineet
    icol = rngLista.Columns(2).Column
    For irow = 1 To rngLista.Rows.Count
        If rngLista(irow, icol).Value = AjoneuvonTyyppi Then
            Exit For
        End If
    Next irow

    If AjoneuvonTyyppi = "" Then
        KuljetustenPaastojenKaava = "Ei kuljetustyyppiä valittuna"
    ElseIf AjoneuvonTyyppi = "Muu" Or AjoneuvonTyyppi = "Työkoneet (mm. traktori)" Then
        KuljetustenPaastojenKaava = "=IFERROR(" & SarakeKirjain(VSar + 2) & R & "*0.01*" & _
        SarakeKirjain(VSar + 7) & R & "*" & SarakeKirjain(VSar + 3) & R & "*0.001/" & _
        SarakeKirjain(VSar + 4) & R & "*" & SarakeKirjain(VSar + 9) & R & ",0)"
        Cells(R, Range("AjoneuvonTyyppi").Column + 5).ClearContents
    ElseIf Left(AjoneuvonTyyppi, 11) = "Henkilöauto" Then
        KuljetustenPaastojenKaava = "=IFERROR(" & SarakeKirjain(VSar + 2) & R & "*" & _
        SarakeKirjain(VSar + 9) & R & "*" & SarakeKirjain(VSar + 3) & R & "*0.001/" & _
        SarakeKirjain(VSar + 4) & R & ",0)"
    ElseIf irow >= raja Then ' Laiva, rautatie, lennot
        KuljetustenPaastojenKaava = "=IFERROR(" & SarakeKirjain(VSar + 2) & R & "*" & _
        SarakeKirjain(VSar + 3) & R & "*0.001*" & SarakeKirjain(VSar + 9) & R & ",0)"
        Cells(R, Range("LogKuormanMassa").Column).Interior.Color = xlNone
        Cells(R, Range("AjoneuvonTyyppi").Column + 5).ClearContents
    Else
        KuljetustenPaastojenKaava = "=IFERROR(" & SarakeKirjain(VSar + 2) & R & "*" & _
        SarakeKirjain(VSar + 9) & R & "*(" & SarakeKirjain(VSar + 3) & R & _
        "*0.001/" & SarakeKirjain(VSar + 4) & R & "),0)"
        Cells(R, Range("LogKuormanMassa").Column).Interior.Color = _
        Cells(R, Range("AjoneuvonTyyppi").Column).Interior.Color
    End If

End Function
```

Kuva 49. Kuljetusten päästöjen laskentaan käytettävien kaavojen määrittämiseen tarkoitettu funktio.

Funktiossa on muuttujien määrittelyjen jälkeen For-silmukka, jonka tarkoituksena on etsiä, monennellako rivillä tietokannan kuljetusvälineiden päästökertoimien luettelossa haluttu ajoneuvo sijaitsee. Tätä tietoa tarvitaan siksi, että tietokannassa kuljetusvälineet on järjestetty siten, että ensimmäisenä ovat ne ajoneuvot, joille päästökerroin on saatavilla erikseen

tyhjälle ja täydelle kuormalle. Tämä saadaan selville myöhemmin vertaamalla rivinnumeroa raja-muuttujaan, joka saa arvonsa tkmValineet-makron kautta. Tämä makro selvittää, missä kohtaa listaa päästökertoimet muuttuvat tonnikilometriperusteisiksi. Tätä tietoa käytetään siis siihen, että tiedetään mitä tietoja valitusta kuljetusvälineestä on saatavilla. Tämä ei ole välttämättä paras tai tehokkain tapa toteuttaa laskentaperiaatteen valintaa, mutta se toimii niin kauan, kuin tietokannan rakenne pysyy samanlaisena.

Seuraavaksi ehtorakenteen avulla selvitetään, minkä tyyppinen kuljetusväline valittiin ja sen mukaan tallennetaan funktion String-muuttujaksi haluttu kaava. Kaava on erilainen muille ajoneuvoille ja työkoneille, henkilöautoille, tonnikilometriperusteisen päästökertoimen omaaville kuljetusvälineille (laivat, raideliikenne ja lennot) sekä kuljetusvälineille, joiden päästökertoimet tiedetään erikseen tyhjälle ja täydelle kuormalle. Nämä kaavat ja laskentaperiaatteet ovat esitetty tarkemmin IKE-hiilijalanjälkilaskurin laskentaoppaassa kappaleessa 3.7.3.

4.3.3 Käyttöliittymät

Käyttöliittymät-moduuli on hyvin yksinkertainen, sillä se sisältää pääasiassa makroja, joiden tehtävä on reagoida käyttäjän painikkeen painamiseen ja aukaista näkyväksi oikea käyttöliittymä. Moduulista löytyvät siis keskitetysti kaikki käyttöliittymien avaamiseen käytettävät makrot. Kuvassa 50 on esitetty kuvakaappaus JakelunOletusSkenaariotNäytä-makrosta, joka avaa JakelunOletusSkenaariot-käyttöliittymän.

```
Sub JakelunOletusSkenaariotNäytä()

    Call VapautaSivu("Logistiikka")
    Call NapinKlikkaus1(ActiveSheet.Shapes("AjoOletusSkenaarioNappi"))

    Application.ScreenUpdating = False

    JakelunOletusSkenaariot.Show

    Application.ScreenUpdating = True

    Call NapinKlikkaus2(ActiveSheet.Shapes("AjoOletusSkenaarioNappi"))
    Call SuojaaSivu("Logistiikka")

End Sub
```

Kuva 50. JakelunOletusSkenaariotNäytä-makro.

Makrossa ensin vapautetaan sen välilehden suojaus, jolla makron käynnistänyt painike sijaitsee. Tämän jälkeen kutsutaan makroa, joka luo ”klikkausefektin” painetulle painikkeelle. Sitten suljetaan näytön päivitys, avataan käyttöliittymä, palautetaan näytön päivitys, palautetaan painike alkuperäiseen ulkomuotoonsa ja suojataan välilehti.

Tällaisten käyttöliittymien avaamiseen liittyvien makrojen lisäksi moduulista löytyy kolme oletusskenaarioiden käyttöön kutsuttavaa rivien värjäysmakroa, joiden tarkoitus on nimensä mukaisesti värjätä rivi valkoiseksi oletusskenaarioita käytettäessä. Kuvassa 51 on esitetty esimerkkinä kuvakaappaus makrosta, joka värjää kuljetusvälineiden listassa rivin valkoiseksi.

```
Sub VarjaaRiviLog()

    Call VapautaSivu("Logistiikka")

    With Range("LogistiikkaYhteensa")
        .Offset(-1, 0).Interior.Color = RGB(255, 255, 255)
        .Offset(-1, 1).Interior.Color = RGB(255, 255, 255)
        .Offset(-1, 2).Interior.Color = RGB(255, 255, 255)
        .Offset(-1, 3).Interior.Color = RGB(255, 255, 255)
        .Offset(-1, 4).Interior.Color = RGB(255, 255, 255)
    End With

    Call SuojaaSivu("Logistiikka")

End Sub
```

Kuva 51. Kuljetusvälineiden rivin valkoiseksi värjäävä VarjaaRiviLog-makro.

4.3.4 Näytäkentät_esittele

Näytäkentät_esittele-moduulin sisältää vain yhteen yksittäiseen toimintoon liittyviä makroja. Moduulin makrot liittyvät laskurin etusivulta löytyvän Piilota prosessitiedot / Näytä prosessitiedot -kytkimen toimintaan, jonka tarkoituksena on mahdollistaa laskurista tarkkojen prosessitietojen piilottaminen esimerkiksi esitystarkoituksissa. Moduuli sisältää 5 makroa: Naytto_Esittely_Click, Esittele, NaytaKentat, fEsittely(YNurk As Range, ANurk As Range) ja fNaytaKentat(YNurk As Range, ANurk As Range, Optional VariAlue As Range = Nothing, Optional Vari As String = "").

Ensimmäinen makro Naytto_Esittely_Click reagoi käyttäjän suorittamaan painikkeen klikkaukseen. Riippuen siitä, missä asennossa kytkin oli, makro valitsee ehtorakenteen avulla kutsuttavan makron ja muuttaa painikkeiden väriä ja tekstiä tämän mukaisesti. Kuvassa 52 on esitetty kuvankaappaus kyseisestä makrosta.

```
Sub Naytto_Esittely_Click()

    Call VapautaKaikki
    If ActiveSheet.Shapes("SensuuriNappula").Fill.ForeColor.RGB = RGB(251, 173, 24) Then
        ActiveSheet.Shapes("SensuuriNappula").Fill.ForeColor.RGB = RGB(218, 217, 215)
        ActiveSheet.Shapes("SensuuriNappula").TextFrame.Characters.Text = "Piilota prosessitiedot"
        Call NaytaKentat
    ElseIf ActiveSheet.Shapes("SensuuriNappula").Fill.ForeColor.RGB = RGB(218, 217, 215) Then
        ActiveSheet.Shapes("SensuuriNappula").Fill.ForeColor.RGB = RGB(251, 173, 24)
        ActiveSheet.Shapes("SensuuriNappula").TextFrame.Characters.Text = "Näytä prosessitiedot"
        Call Esittele
    End If
    Call SuojaaKaikki

End Sub
```

Kuva 52. Prosessitietojen piilottamisen tai näyttämisen käynnistävä makro.

Kuten edellisestä makrosta voi huomata, makro kutsuu painikkeen vallitsevan tilan mukaan joko NaytaKentat- tai Esittele-makroa. Nämä makrot ovat niin pitkiä, etteivät ne mahtuisi kokonaisena kuvankaappauksena järkevästi tähän oppaaseen, joten kuvassa 53 on esitetty Esittele makron alkuosa, josta makron toimintaperiaate on kyllä hyvin ymmärrettävissä.

```
Sub Esittele()

    Call VapautaKaikki
    Application.EnableEvents = False

    Dim shRaa As Worksheet
    Dim shPak As Worksheet
    Dim shJat As Worksheet
    Dim shEner As Worksheet
    Dim shLog As Worksheet
    Dim shYht As Worksheet
    Dim shTase As Worksheet

    Set shRaa = ThisWorkbook.Sheets("Raaka-aineet")
    Set shPak = ThisWorkbook.Sheets("Pakkausmateriaalit")
    Set shJat = ThisWorkbook.Sheets("Jätteet ja sivuvirrat")
    Set shEner = ThisWorkbook.Sheets("Energia")
    Set shLog = ThisWorkbook.Sheets("Logistiikka")
    Set shYht = ThisWorkbook.Sheets("Yhteenveto")
    Set shTase = ThisWorkbook.Sheets("Tasetarkastelu")

' Raaka-aineet
    ' määrä, osuus reseptistä, ja päästöt

    Call fEsittely(shRaa.Range("RaaMaara").Offset(1, 0), shRaa.Range("RaakaaineetYhteensa").Offset(0, 2))
    Call fEsittely(shRaa.Range("RaaPaastot").Offset(1, 0), shRaa.Range("RaakaaineetYhteensa").Offset(0, 4))

' Pakkausmateriaalit
    'määrä, massa, montako tuotetta, päästöt

    Call fEsittely(shPak.Range("PakMaara").Offset(1, 0), shPak.Range("PakkauksetYhteensa").Offset(0, 4))
    Call fEsittely(shPak.Range("PakPaastot").Offset(1, 0), shPak.Range("PakkauksetYhteensa").Offset(0, 6))
' Jätteet ja sivuvirrat
    'Syntyvaihe, sivuvirta/jäte, Määrä, päästöt

    Call fEsittely(shJat.Range("JatSyntyvaihe").Offset(1, 0), shJat.Range("JatteetYhteensa").Offset(0, 3))
    Call fEsittely(shJat.Range("JatPaastot").Offset(1, 0), shJat.Range("JatteetYhteensa").Offset(0, 5))

    Call fEsittely(shJat.Range("PesuAineet").Offset(1, 1), shJat.Range("PesuAineetYhteensa").Offset(0, 2))
    Call fEsittely(shJat.Range("PesuAineet").Offset(1, 4), shJat.Range("PesuAineetYhteensa").Offset(0, 4))
```

Kuva 53. Osittainen Esittele-makro

Esittele-makron tarkoituksena on piilottaa tiedot määrätyltä alueelta. Tämä suoritetaan käyttämällä fEsittely-funktiota, josta tarkemmin seuraavassa kappaleessa. Makron alussa vapautetaan kaikki laskurin työkirjan suojaukset, estetään Microsoft Excel Objects-makrojen toiminta ja määritellään muuttujat. Tämän jälkeen aloitetaan haluttujen alueiden piilottaminen välilehti kerrallaan, joista kuvassa 53 on nähtävissä Raaka-aineet-, Pakkausmateriaalit- ja Jätteet ja sivuvirrat -välilehtien piilotukset. Tämän jälkeen makrossa käsitellään samaan tyyliin muutkin välilehdet Päästökertoimet- ja Dataa-välilehtiä lukuun ottamatta. Kutsuttavalle funktiolle annetaan piilotettavasta alueesta sijaintitieto Range-muuttujina vasemmasta ylänurkasta ja oikeasta alanurkasta.

fEsittely-funktio on toimintaperiaatteeltaan hyvin yksinkertainen. Kuten edellisessä kappaleessa mainittiin, se saa syötteinä kaksi sijaintia, peitettävän alueen ylä- ja alanurkan. Näiden perusteella funktio muodostaa kokonaisen alueen PeitettavaAlue, jonka funktio ”peittää” muuttamalla alueen solujen taustavärin ja tekstien fontin värin saman sävyisellä harmaalla

värillä sekä poistaa solujen rajat. Tällöin näyttää siltä, ikään kuin alue olisi peitetty harmaalla laatikolla. Kuvakaappaus fEsittely-funktiosta on nähtävissä kuvasta 54.

```
Private Function fEsittely(YNurk As Range, ANurk As Range)

    Dim PeitettavaAlue As Range

    Set PeitettavaAlue = Range(YNurk, ANurk)

    PeitettavaAlue.Interior.Color = RGB(128, 128, 128)
    PeitettavaAlue.Font.Color = RGB(128, 128, 128)
    PeitettavaAlue.Borders.LineStyle = xlNone


End Function
```

Kuva 54. fEsittely-funktio.

Haluttujen alueiden peittäminen on jokseenkin yksinkertaista, kun verrataan näiden peitettyjen alueiden palauttamiseen näkyviksi. Kaikki solut eivät ole saman värisiä, ja nämä värit eivät välttämättä vaihtele minkään suoraviivaisen logiikan mukaisesti. Kuvassa 55 on esitetty osittainen kuvakaappaus NäytäKentat-makrosta.

```
Sub NaytaKentat()

    Call VapautaKaikki
    Application.EnableEvents = False

    Dim shRaa As Worksheet
    Dim shPak As Worksheet
    Dim shJat As Worksheet
    Dim shEner As Worksheet
    Dim shLog As Worksheet
    Dim shYht As Worksheet
    Dim shTase As Worksheet

    Set shRaa = ThisWorkbook.Sheets("Raaka-aineet")
    Set shPak = ThisWorkbook.Sheets("Pakkausmateriaalit")
    Set shJat = ThisWorkbook.Sheets("Jätteet ja sivuvirrat")
    Set shEner = ThisWorkbook.Sheets("Energia")
    Set shLog = ThisWorkbook.Sheets("Logistiikka")
    Set shYht = ThisWorkbook.Sheets("Yhteenveto")
    Set shTase = ThisWorkbook.Sheets("Tasetarkastelu")

    ' Raaka-aineet
    Call fNaytaKentat(shRaa.Range("RaaMaara").Offset(1, 0), shRaa.Range("RaakaaineetYhteensa"). _
    Offset(0, 1), VariAlue:=shRaa.Range("RaaRaakaaineet").Offset(1, 0))
    Call fNaytaKentat(shRaa.Range("RaaMaara").Offset(1, 2), shRaa.Range("RaakaaineetYhteensa"). _
    Offset(0, 3), VariAlue:=shRaa.Range("RaaRaakaaineet").Offset(1, 0))
    Call fNaytaKentat(shRaa.Range("RaaMaara").Offset(1, 1), shRaa.Range("RaakaaineetYhteensa"). _
    Offset(0, 2), VariAlue:=shRaa.Range("RaaRaakaaineet").Offset(1, 6))
    Call fNaytaKentat(shRaa.Range("RaaPaastot").Offset(1, 0), shRaa.Range("RaakaaineetYhteensa"). _
    Offset(0, 4), VariAlue:=shRaa.Range("RaaRaakaaineet").Offset(1, 6))
    Range(shRaa.Range("RaakaaineetYhteensa"), shRaa.Range("RaakaaineetYhteensa").Offset(0, 5)). _
    BorderAround Weight:=xlThick

    ' Pakkausmateriaalit
    Call fNaytaKentat(shPak.Range("PakMaara").Offset(1, 0), shPak.Range("PakkauksetYhteensa"). _
    Offset(0, 3), VariAlue:=shPak.Range("Pakkausmateriaalit").Offset(1, 0))
    Call fNaytaKentat(shPak.Range("PakPaastot").Offset(1, -2), shPak.Range("PakkauksetYhteensa"). _
    Offset(0, 4), Vari:="Valkoinen")
    Call fNaytaKentat(shPak.Range("PakPaastot").Offset(1, -1), shPak.Range("PakkauksetYhteensa"). _
    Offset(0, 5), Vari:="Oranssi")
    Call fNaytaKentat(shPak.Range("PakPaastot").Offset(1, 0), shPak.Range("PakkauksetYhteensa"). _
    Offset(0, 6), VariAlue:=shPak.Range("Pakkausmateriaalit").Offset(1, 8))
    Range(shPak.Range("PakkauksetYhteensa"), shPak.Range("PakkauksetYhteensa").Offset(0, 7)). _
    BorderAround Weight:=xlThick
    Range(shPak.Range("PakkauksetYhteensa"), shPak.Range("PakkauksetYhteensa").Offset(0, 7)). _
    Interior.Color = RGB(255, 255, 255)
```

Kuva 55. Osittainen NaytaKentat-makro.

Kuvasta 55 voi nähdä, että se on hyvin samankaltainen Esittele-makron kanssa. Näiden oleellinen ero liittyy kutsuttavan fNaytaKentat-funktion toimintaan, joka on fEsittely-funktioon verrattuna huomattavasti monimutkaisempi. fNaytaKentat vaatii syötteenä samat tiedot kuin fEsittely (ylä- ja alanurkkien sijainnit) sekä näiden lisäksi mahdollisesti vapaaehtoisina muuttujina VariAlue-nimisen Range-muuttujan ja Vari-nimisen String-muuttujan. Kuvassa 56 on esitetty kuvakaappaus fNaytaKentat-funktiosta.

```
Private Function fNaytaKentat(YNurk As Range, ANurk As Range, _
Optional VariAlue As Range = Nothing, Optional Vari As String = "")

    Dim NaytettavaAlue As Range
    Dim rivi As Range
    Dim irow As Integer
    Dim fVari As String
    Set NaytettavaAlue = Range(YNurk, ANurk)
    If Vari = "" Then
        For irow = 1 To NaytettavaAlue.Rows.Count
            Set rivi = Range(NaytettavaAlue(irow, 1), NaytettavaAlue _
            (irow, NaytettavaAlue.Columns.Count))

            Vari = VariAlue.Offset(irow - 1, 0).Interior.Color

            fVari = VariAlue.Offset(irow - 1, 0).Font.Color
            rivi.Interior.Color = Vari
            rivi.Font.Color = fVari
        Next irow
    ElseIf Vari = "Varaston tyyppi" Then
        For irow = 1 To NaytettavaAlue.Rows.Count
            Set rivi = Range(NaytettavaAlue(irow, 1), NaytettavaAlue _
            (irow, NaytettavaAlue.Columns.Count))
            If VariAlue(irow) = "Lämmin varasto" Then
                rivi(1, 1).Interior.Color = VariAlue.Offset(irow - 1, 0).Interior.Color
                rivi(1, 2).Interior.Color = RGB(255, 255, 255)
            ElseIf VariAlue(irow, 1) = "Jäähdytetty varasto" Then
                rivi(1, 2).Interior.Color = VariAlue.Offset(irow - 1, 0).Interior.Color
                rivi(1, 1).Interior.Color = RGB(255, 255, 255)
            ElseIf VariAlue(irow, 1) = "Lämmittämätön varasto" Then
                rivi.Interior.Color = RGB(255, 255, 255)
            Else
                rivi.Interior.Color = RGB(255, 255, 255)
            End If
            rivi.Font.Color = RGB(0, 0, 0)
        Next
    Else
        For irow = 1 To NaytettavaAlue.Rows.Count
            Set rivi = Range(NaytettavaAlue(irow, 1), NaytettavaAlue _
            (irow, NaytettavaAlue.Columns.Count))
            If Vari = "Oranssi" Then
                rivi.Interior.Color = RGB(251, 173, 24)
            ElseIf Vari = "Valkoinen" Then
                rivi.Interior.Color = RGB(255, 255, 255)
            ElseIf Vari = "Keltainen" Then
                rivi.Interior.Color = RGB(255, 255, 0)
            End If
            rivi.Font.Color = RGB(0, 0, 0)

            If Not VariAlue Is Nothing Then
                If VariAlue.Offset(irow - 1, -15).Value = "Muu" Or VariAlue.Offset _
                (irow - 1, -15).Value = "Työkoneet (mm. traktori)" Then
                    VariAlue.Offset(irow - 1, -7).Interior.Color = RGB(251, 173, 24)
                    VariAlue.Offset(irow - 1, -8).Interior.Color = RGB(251, 173, 24)
                End If
            End If
        Next irow
    End If
    NaytettavaAlue.Borders.LineStyle = xlContinuous

End Function
```

Kuva 56. fNaytaKentat-funktio.

fNaytaKentat-funktio hyödyntää toiminnassaan sekä ehto- että silmukkarakenteita, jotta kaikkien solujen värit saataisiin palautettua siten, kuten ne alun perin ennen peittämistä olivatkin. Paljastettavan alueen määrittämiseen käytettävät ”nurkkasolut” toimivat kuten fEsittely-funktiossakin. Funktio koostuu suuresta ehtorakenteesta, jonka tarkoituksena on

ensiksi määrittää tekniikka, jolla solujen värit palautetaan. Tässä ehtorakenteessa hyödynnetään kahta valinnaista syötettä VariAlue ja Vari. Ensiksi testataan, onko Vari-muuttujan arvo tyhjä (””). Jos on, määritetään värit VariAlue-muuttujan mukaisesti. Tällöin koko paljastettava alue käydään For-silmukan avulla läpi rivi kerrallaan, ja palautetaan solujen taustavärit ja fonttien värit vastaaviksi, kuten VariAlue-sarakkeessa.

Jos Vari-muuttujaan on tallennettu teksti ”Varaston tyyppi”, käytetäänkin värien määrittämisessä hieman erilaista lähestymistapaa. Tämä johtuu siitä, että tätä vaihtoehtoa käytetään nimenomaan varastojen taulukossa logistiikkavälilehdellä, jolloin tiettyjen solujen värit riippuvat valitusta varastotyypistä. Tällöin selvitetään VariAlue-muuttujan arvo, josta ohjelma päättelee solujen oikeat värit.

Mikäli Vari-muuttujan arvo ei ole tyhjä eikä ”Varaston tyyppi”, käytetään ehtorakenteen viimeistä Else-lohkoa. Tässä samalla tavalla For-silmukassa rivi kerrallaan värjätään Vari-muuttujan arvon mukaisesti solut joko oranssiksi, valkoiseksi tai keltaiseksi. Keltaista ei laskurin viimeisimmässä versiossa enää käytetä, vaan solut ovat joko valkoisia tai oransseja. Else-lohkon sisällä on vielä lopuksi erikoistilannetta varten uusi ehtorakenne, jossa testataan, onko VariAlue-muuttujaan tallennettu jotakin. Tätä tarvitaan kuljetusvälineiden listan tapauksessa, jossa kuljetusvälineeksi on valittu vaihtoehto ”Muu” tai ”Työkoneet (mm. traktori)”. Aivan funktion lopuksi vielä muutetaan näytettävän alueen rajat näkyviksi.

4.3.5 Ohjeen_avaus

Ohjeen_avaus -moduuli on keskittynyt myös tarkasti yhden toiminnon suorittamiseen, eli Word-pohjaisen ohjetiedoston ”IKE-hiilijalanjälkilaskurin käyttöopas” avaamiseen juuri oikeasta kohdasta. Laskurissa on jokaisella välilehdellä sopivissa kohdissa pyöreitä ja punaisia infonappuloita, joiden avulla käyttäjä voi navigoida helposti oikeaan kohtaan tiedostossa. Näitä infonappuloita on laskurissa yhteensä 29 kappaletta, joista osa sijaitsee käyttöliittymien sisällä harmaana ja neliskanttisena Ohje-painikkeena.

Moduulissa on jokaiselle info- tai ohjepainikkeelle oma makro, joka aktivoituu käyttäjän klikatessa painiketta. Nämä makrot eivät tee käytännössä mitään muuta, kuin kutsuvat

OhjeenAukaisu(Hakusana As String)-funktiota, jossa varsinainen ohjeen aukaiseminen tehdään. Kuvassa 57 on esitetty kuvakaappaus tästä funktiosta.

```
Function OhjeenAukaisu(Hakusana As String)

    Dim polku As String
    Dim wdApp As Object
    Dim wdDok As Object
    Dim TiedostonNimi As String

    TiedostonNimi = "IKE-hiilijalanjälkilaskurin käyttöopas.docx"

    ' Tehdään raporttipohjasta kopio ja nimeä se tuotteen nimen perusteella

    polku = Application.ActiveWorkbook.Path & "\" & TiedostonNimi

    'Jos raporttipohjaa ei löydy, anna virheilmoitus.

    If TarkistaLoytyykoPohja(polku) = False Then
        Exit Function
    End If

    ' Testataan, onko tiedosto jo auki ja avataan muokattavaksi.

    On Error Resume Next
    Set wdApp = GetObject(, "Word.Application")
    If Err.Number <> 0 Then
        ' Word ei ollut käynnissä, joten käynnistetään se
        Set wdApp = CreateObject("Word.Application")
    End If
    On Error GoTo 0

    For Each wdDok In wdApp.Documents
        ' Jos dokumentin nimi vastaa etsittävää nimeä, sulje dokumentti
        If wdDok.Name = TiedostonNimi Then
            wdDok.Close
            Set wdDok = Nothing
            Exit For
        End If
    Next wdDok

    Set wdDok = wdApp.Documents.Open(Filename:=polku, ReadOnly:=True)
    wdApp.Visible = True
    wdDok.Activate

    wdDok.Bookmarks(Hakusana).Range.Select
    wdApp.ActiveWindow.SmallScroll down:=20
    wdApp.ActiveWindow.WindowState = wdWindowStateMaximize

    On Error GoTo Virhe

    AppActivate TiedostonNimi
    Set wdDok = Nothing
    Set wdApp = Nothing

    On Error GoTo 0
    Exit Function

Virhe:

    AppActivate wdApp.Caption
    Set wdDok = Nothing
    Set wdApp = Nothing

    On Error GoTo 0

End Function
```

Kuva *57*. Kuvakaappaus OhjeenAukaisu-makrosta.

Tässä funktiossa erikoista aiempiin verrattuna on se, että siinä ohjataan myös Wordin toimintaa eikä pelkästään Exceliä. Jotta funktio ymmärtää, kumpaan applikaatioon viitataan, täytyy Wordiin kohdistettaviin komentoihin käyttää omia objektejaan. Näitä varten määritellään wdApp- ja wdDok-nimiset objektit. Näiden lisäksi määritellään polku- ja TiedostonNimi-nimiset String-muuttujat.

Muuttujien nimeämisen jälkeen määritellään TiedostonNimi-muuttujaan arvoksi avattavan ohjetiedoston nimi: "IKE-hiilijalanjälkilaskurin käyttöopas.docx”. Tämän jälkeen määritellään myös polku sillä oletuksella, että ohjetiedosto sijaitsee samassa kansiossa itse laskurin tiedoston kanssa ja TiedostoNimi vastaa haluttua tiedoston nimeä. Näiden määritysten jälkeen funktio testaa yksinkertaisella kutsuttavalla makrolla, onko määriteltyä tiedostoa kyseisellä polulla olemassa. Tämä kutsuttava makro: TarkistaLoytyykoPohja sijaitsee Raportin_Tulostus -moduulissa.

Ohjeen avauksen halutaan toimivan aina riippumatta siitä, onko ohjetiedosto jo valmiiksi auki vai ei. Tästä syystä funktio ensin yrittää määrittää wdApp-objektiin jo käynnissä olevan Word-applikaation. Mikäli näin ei kuitenkaan ole, normaalisti tällainen komento aiheuttaisi virheen. Kuitenkin ennen tätä on määritelty komento On Error Resume Next, jonka ansiosta ohjelman suoritus ei keskeydy. Seuraavaksi ehtorakenteen avulla käynnistetään Word, mikäli edellinen määritys aiheutti virheen. Tämän jälkeen wdApp-objektiin on määritetty joko aiemmin valmiiksi auki ollut Word tai se aktivoitiin itse.

Tämän jälkeen etsitään For Each-silmukkarakenteella, onko ohjetiedoston nimeä vastaava tiedosto jo auki. Mikäli näin on, tiedosto suljetaan. Näin tehdään, koska testien perusteella jo valmiina olleen tiedoston käyttö aiheutti herkästi ohjelman kaatumisen ja muita virheitä, joten todettiin turvallisemmaksi sulkea se ja avata uudestaan.

Tämän jälkeen ohjetiedosto avataan ReadOnly-tilassa, jotta käyttäjä ei voi tallentaa siihen muutoksia. Applikaatio asetetaan näkyväksi ja ohjetiedosto aktivoidaan. Ohjetiedostoon on määritetty ennalta kirjanmerkkejä, joihin hakusanan perusteella funktio voi navigoida. Seuraavaksi funktio valitseekin hakusanan mukaisen kirjanmerkin ja siirtyy siihen. Tämän jälkeen

ikkunaa hieman ”rullataan”, jotta haluttu kohta olisi keskemmällä ruutua. Lisäksi määritetään ikkunan mitat koko näytön alueelle.

Lopuksi ennen objektien nollaamista halutaan varmistaa, että ohjetiedosto avautui käyttäjälle päällimmäiseksi näytölle, eikä esimerkiksi jää laskurin alle piiloon. Tätä varten jouduttiin lisäämään kaksi vaihtoehtoista tapaa, sillä testien perusteella huomattiin kahden eri vaihtoehdon toimivan eri tietokoneilla, mutta vastaavasti aiheuttavan virheen käytettäessä toisin päin. Koska laskuri tulee julkiseen jakeluun, täytyy molemmat vaihtoehdot huomioida. Funktio yrittää ensin tehdä tämän komennolla AppActivate TiedostonNimi, mutta mikäli tämä aiheuttaa virheen, se kokeilee komentoa AppActivate wdApp.Caption. Teoriassa TiedostonNimi ja wdApp.Caption pitäisi tässä tapauksessa olla aivan sama asia, mutta jostain ohjelman koodanneelle tuntemattomasta syystä näin ei nähtävästi ole.

4.3.6 Raportin_tulostus

Raportin_tulostus on Ohjeen_avaus -moduulin kaltainen Wordia hyödyntävä moduuli, ja kyseiset moduulit hyödyntävätkin osittain samaa koodia. (Euroopan komissio, 2021, s. 99–100.) Raportin_tulostus on myös yhtä tarkoitusta varten hyödynnetty moduuli, eli nimensä mukaisesti sen avulla käyttäjä voi tulostaa tekemästään laskelmasta PEF-ohjeistuksen raportointivaatimuksia mukailevan raportin. Ohjelman perusidea on hyödyntää valmiiksi luotua raporttipohjaa, johon ohjelma sitten lisää tietoa laskurista.

Moduulin pääohjelma on kohtalaisen pitkä, joten tässä kappaleessa esitellään ensin aliohjelmat ja funktiot, joita se hyödyntää. Näitä pienempiä makroja ja funktioita on 11 kappaletta, ja ne liittyvät esimerkiksi datan siirtelyyn Excelin ja Wordin välillä, datan muotoiluun, tai virheenkäsittelyyn. Datan käsittelyyn laskurin Dataa-välilehdellä tarvittavat makrot on esitelty erikseen omassa kappaleessaan 4.3.7. Ensimmäiseksi käsitellään TarkistaTiedostoNimi-funktio, josta kuvakaappaus on esitetty kuvassa 58.

```
Function TarkistaTiedostoNimi(polku As String) As Boolean

    Dim Loytyyko As String

    Loytyyko = Dir(polku)

    If Loytyyko = "" Then
        TarkistaTiedostoNimi = False
    Else
        TarkistaTiedostoNimi = True
    End If

End Function
```

Kuva 58. TarkistaTiedostoNimi-funktio

Tämä funktio on todella yksinkertainen, ja sen tehtävä on nimensä mukaisesti varmistaa, onko määriteltyä tiedostoa olemassa. Funktio saa syötteenä tiedoston polun String-muuttujana, ja funktio saa itse arvon Boolean-muuttujana. Testaus toteutetaan VBA:n sisäisellä Dir-funktiolla, joka ottaa syötteenä tiedoston polun ja palauttaa tiedoston nimen, joka vasta annettua polkua. Mikäli polkua ei löydy, funktio palauttaa tyhjän arvon. Näin saadaan helposti ehtorakenteen avulla testattua, oliko annettua polkua vastaava tiedosto olemassa vai ei.

Seuraava funktio TarkistaLoytyykoPohja on edellisen kanssa hyvin samankaltainen, mutta sisältää hieman enemmän virheentarkistusta. Kuvakaappaus funktiosta on nähtävissä kuvassa 59.

```
Function TarkistaLoytyykoPohja(polku As String) As Boolean

    On Error GoTo Virhe

    Dim Loytyyko As String

    If Left(polku, 4) = "http" Then
        MsgBox "Virhe raporttipohjan etsimisessä. Huomaa, että tiedostot eivät" & _
        " voi olla tallennettuna verkkosijaintiin, jotta ne toimivat oikein.", vbOKOnly, "Virheilmoitus"
        On Error GoTo 0
        Exit Function
    End If

    Loytyyko = Dir(polku)

    If Loytyyko = "" Then
        MsgBox "Raporttipohjaa ei löydy", vbOKOnly, "Virheilmoitus"
        TarkistaLoytyykoPohja = False
    Else
        TarkistaLoytyykoPohja = True
    End If
    Exit Function

Virhe:

    MsgBox "Virhe raporttipohjan etsimisessä.", vbOKOnly, "Virheilmoitus"
    On Error GoTo 0

End Function
```

Kuva 59. TarkistaLoytyykoPohja-funktio

Seuraavat kolme funktiota liittyvät datan siirtämiseen Excelistä Wordiin. Nämä kolme funktiota ovat jaoteltu tekstin siirtämiseen, kuvien siirtämiseen ja taulukoiden siirtämiseen. Ensimmäisenä näistä esitellään EtsiJaKorvaaTekstia-funktio, joka on esitetty kuvassa 60.

```
Private Function EtsiJaKorvaaTekstia(wdDok As Word.Document, Hakusana As String, LisattavaTeksti As String)

    Dim i As Integer
    Dim PalaMaara As Integer
    Dim Pala As String

    If IsNumeric(LisattavaTeksti) = True Then
        LisattavaTeksti = CStr(Round(CSng(LisattavaTeksti), 2))
    End If

    PalaMaara = Round(Len(LisattavaTeksti) / 250, 0)

    If Len(LisattavaTeksti) Mod 250 > 0 Then
       PalaMaara = PalaMaara + 1
    End If

    With wdDok.Content.Find

        If PalaMaara = 1 Or LisattavaTeksti = "" Then
            .Text = Hakusana
            .Replacement.Text = LisattavaTeksti
            .Wrap = wdFindContinue
            .Execute Replace:=wdReplaceAll
        Else
           .Execute FindText:=Hakusana, ReplaceWith:="{1}", Replace:=wdReplaceAll

            For i = 1 To PalaMaara
                Pala = Mid(LisattavaTeksti, ((i - 1) * 250) + 1, 250)

                If i < PalaMaara Then
                    Pala = Pala & "{" & (i + 1) & "}"
                End If

                .Execute FindText:="{" & i & "}", ReplaceWith:=Pala, Replace:=wdReplaceAll
            Next i
        End If
    End With

    On Error GoTo 0

    Application.CutCopyMode = False

End Function
```

Kuva 60. EtsiJaKorvaaTekstia-funktio

EtsiJaKorvaaTekstia-funktion tarkoituksena on lisätä tekstiä valmiiseen Word-pohjaiseen raporttipohjaan. Lisääminen tapahtuu käytännössä korvaamalla ennalta määrätyt teksti-osat laskurista haetuilla tiedoilla. Esimerkiksi raporttipohjaan on kirjattu valmiiksi haluttuihin kohtiin teksti ”<TuotteenNimi>”, jotka funktio sitten oikeiden syötteiden avulla korvaa laskurin etusivulta löytyvän ”Tuotteen nimi:” viereisen solun sisältämällä tekstillä. Funktio tarvitsee syötteinä muokattavan Word-dokumentin objektin, Hakusana-tekstin, joka raportista korvataan sekä LisattavaTeksti, jolla edellinen Hakusana korvataan.

Funktio lähtee muuttujien määrittämisen jälkeen liikkeelle siitä, että se tarkistaa, onko LisattavaTeksti numeeriseksi muutettavissa oleva arvo. Mikäli näin on, funktio ensin muuttaa sen String-muuttujasta Single-muuttujaksi, pyöristää sen kahden desimaalin tarkkuuteen, ja lopuksi muuttaa sen takaisin String-muuttujaksi. Näin vältetään valtavan pitkien desimaalilukujen vieminen raporttiin.

Tämän jälkeen aloitetaan varsinainen tekstin korvaaminen. VBA:ssa on eräs tätä toimenpidettä monimutkaistava seikka, sillä funktiossa käytettävät Find- ja Replace-funktiot eivät kykene käsittelemään yli 255 merkin mittaisia tekstejä, vaikka itse String-muuttujaan sopiikin yli 30 000 merkkiä. Tämä ongelma on kierretty siten, että tekstin ollessa yli 250 merkkiä pitkä teksti pilkotaan maksimissaan 250 merkin mittaisiin pätkiin, jotka sitten lisätään raporttiin yksitellen peräkkäin. Ensin funktio selvittää pilkottujen osien lukumäärän, jonka jälkeen siirrytään varsinaiseen tekstin korvaamiseen.

Tekstin korvaaminen toteutetaan ehtorakenteella, jossa ensin funktio tutkii, onko pilkottuja osia vain yksi tai onko LisattavaTeksti-muuttuja tyhjä. Jos näin on, korvaaminen suoritetaan normaalisti etsimällä Hakusana-muuttujaa vastaava teksti, määrittelemällä korvaava teksti, määrittelemällä rivitysasetuksen ja suorittamalla korvauksen. Jos taas pilkottujen osien määrä on suurempi kuin yksi, lisätään teksti For-silmukkaa hyödyntäen pala kerrallaan. Tämä tapahtuu siten, että ennen silmukkaa Hakusana-muuttujaa vastaava teksti etsitään ja korvataan tekstillä {1}. Tämän jälkeen aloitetaan silmukka ykkösestä aina ”palojen” lukumäärää vastaavaan lukuun asti. Silmukan sisällä ensin määritellään Pala-nimiseen String-muuttujaan lisättävän tekstiosa Mid-funktiota hyödyntäen. Tämä funktio ottaa syötteenä halutun tekstin, kuinka monennen merkin kohdalta funktio ottaa tekstin, ja kuinka monta merkkiä sisällytetään. Tällä tavalla saadaan se pala tekstistä määritettyä muuttujaan, joka olisi seuraavaksi lisättävänä. Tämän jälkeen funktio tarkistaa ehtorakenteella, onko paloja vielä tulossa lisää. Mikäli on, lisätään Pala:n loppuun vielä {i+1}, jossa i vastaa silmukan menossa olevaa kierrosnumeroa. Tällä tavalla funktio osaa seuraavalla silmukan kierroksella lisätä seuraavan palan oikeaan kohtaan edellisen perään. Tämän jälkeen suoritetaan tekstin korvaus, ja jatketaan seuraavalle kierrokselle.

Seuraavassa LisaaTaulukko-funktiossa nimensä mukaisesti lisätään haluttu taulukko Excelistä Wordiin. Itse taulukon liittäminen on yksinkertainen toiminto, mutta tämän jälkeiset taulukon muotoiluun ja asetteluun liittyvät seikat tekevät funktiosta hieman monimutkaisemman. Kuvakaappaus funktiosta on nähtävissä kuvassa 61.

```
Private Function LisaaTaulukko(wdApp As Word.Application, wdDok As Word.Document, _
Kirjanmerkki As String, PohjaMerkki As String, AlkuMerkki As String, Alue As Range)

    Dim Taulu As Word.Table
    Dim SNumOtsikko As Integer
    Dim SNumPohja As Integer
    Dim vastaus As Integer

    On Error GoTo Virhe

    Alue.Copy

    wdDok.Activate

    With wdApp.Selection
        .GoTo What:=wdGoToBookmark, Name:=Kirjanmerkki
        .PasteExcelTable False, True, False
    End With

    On Error GoTo 0

    wdApp.ActiveDocument.Fields.Update
    wdApp.ActiveDocument.Repaginate

    Set Taulu = wdDok.Tables(wdDok.Tables.Count)

    With Taulu
        .Range.Font.Size = 10
        .AutoFitBehavior (wdAutoFitContent)
        .Columns.AutoFit
        .Columns(1).AutoFit
        .PreferredWidthType = wdPreferredWidthPercent
        .PreferredWidth = 100
        With .Range.Borders
            .Enable = True
            .InsideLineStyle = True
            .InsideLineStyle = wdLineStyleSingle
            .OutsideLineStyle = wdLineStyleSingle
            .OutsideLineWidth = wdLineWidth150pt
            .OutsideColor = RGB(0, 0, 0)
            .InsideColor = RGB(0, 0, 0)
        End With
    End With

    SNumOtsikko = wdApp.ActiveDocument.Bookmarks(Kirjanmerkki).Range.Information(wdActiveEndAdjustedPageNumber)
    SNumPohja = wdApp.ActiveDocument.Bookmarks(PohjaMerkki).Range.Information(wdActiveEndAdjustedPageNumber)

    If SNumOtsikko <> SNumPohja And Not Left(Kirjanmerkki, 5) = "Liite" Then
        wdApp.Selection.GoTo What:=wdGoToBookmark, Name:=AlkuMerkki
        wdApp.Selection.InsertBreak Type:=wdPageBreak
    End If

    Application.CutCopyMode = False

    On Error GoTo 0


    Exit Function

Virhe:

    vastaus = MsgBox("Odottamaton virhe taulukon viemisessä raporttiin. " & _
    "Haluatko silti jatkaa raportin tulostamista?", vbYesNo, "Virhe taulukon liittämisessä")

    If vastaus = vbYes Then

    Else
        Set wdApp = Nothing
        Set wdDok = Nothing
        ThisWorkbook.Sheets("Yhteenveto").Activate
        Application.ScreenUpdating = True
        On Error GoTo 0
        End
    End If

    On Error GoTo 0

End Function
```

Kuva 61. LisaaTaulukko-funktio.

Funktio tarvitsee useita muuttujia syötteenä toimiakseen. Funktiolle syötetään Word applikaatio ja dokumentti objekteina, Kirjanmerkki, johon taulukko halutaan lisätä, Pohjamerkki, joka kuvastaa taulukon alareunan sijaintia, Alkumerkki, joka kuvastaa taulukon yläreunan sijaintia sekä Alue, joka kuvastaa sijaintia Excelissä, josta taulukko haetaan.

Muuttujien määrittämisen jälkeen suoritetaan heti taulukon kopioiminen Excelistä ja sen liittäminen Wordiin halutulle paikalleen. Tämän kohdan ympärille on lisätty virheenkäsittelyn vuoksi On Error GoTo Virhe ja On Error GoTo 0, koska jostain syystä sekä .Copy että .PasteExcelTable -metodit ovat hyvin virheherkkiä, ja saattavat aiheuttaa satunnaisesti virheen ilman sen kummempaa syytä.

Taulukon liittämisen jälkeen Word-tiedoston kentät päivitetään (sisällysluettelo) ja korjataan sivunumeroinnit. Tämän jälkeen taulukkoa muotoillaan siistimmäksi muuttamalla muun muassa fonttia, sarakkeiden leveyksiä, taulukon leveyttä ja taulukon rajojen ominaisuuksia. Muotoilujen jälkeen testataan vielä taulukon yläpuolen ja alapuolen kirjanmerkkien avulla, onko taulukko jakaantunut useammalle sivulle. Mikäli näin on, lisätään sivunvaihto ennen taulukkoa.

Funktion lopussa on vielä Virhe: lohko, johon funktio ohjaa, mikäli alussa taulukon kopioiminen tai liittäminen aiheuttaa virheen. Funktio antaa tällöin ilmoituksen virheestä ja kysyy, halutaanko jatkaa siitä huolimatta, vaikka taulukko jäi nyt liittämättä. Mikäli halutaan, funktio loppuu normaalisti ja pääohjelma ei keskeydy, mutta jos käyttäjä haluaa keskeyttää, lopetetaan koko ohjelman suoritus eikä raporttia tulosteta loppuun asti.

Kolmas saman tapainen funktio tekstin ja taulukon lisäämisten kanssa on LisaaKuva-funktio. Tämä funktio on koodiltaan ja toimintaperiaatteeltaan lähes täysin vastaava, kuin LisaaTaulukko-funktio, joten tässä oppaassa ei käydä sitä uudestaan tarkemmin läpi. Oleellinen ero näiden kahden funktion välillä on se, että LisaaKuva-funktiossa ei ole vastaavia taulukon muotoiluun liittyviä koodirivejä.

Seuraava funktio liittyy jo liitettyjen taulujen ulkomuodon siistimiseen. TrimmaaTaulut-funktion tarkoituksena on tietyistä leveimmistä taulukoista poistaa sarakkeita, jotka eivät ole

raportin kannalta täysin välttämättömiä. Tämä tehdään, jotta suurimmilla taulukoilla olisi edes teoriassa mahdollisuus mahtua siististi Word tiedostoon. Funktio käy For Each-silmukalla kaikki Word-dokumentin taulukot läpi ja pesuvesien lämmityksen, ajoneuvojen ja varastojen taulukoista poistaa tietyt ennalta määrätyt sarakkeet. Kuvassa 62 on esitetty kuvakaappaus kyseisestä funktiosta.

```
Private Function TrimmaaTaulut(wdApp As Word.Application, wdDok As Word.Document)

    Dim Taulu As Table
    For Each Taulu In wdApp.ActiveDocument.Tables
        If Left(Taulu.cell(1, 1).Range.Text, 5) = "Pesuv" Then
            Taulu.Columns(6).Delete
            Taulu.Columns(6).Delete
        ElseIf Left(Taulu.cell(1, 1).Range.Text, 5) = "Ajone" Then
            Taulu.Columns(6).Delete
        ElseIf Left(Taulu.cell(1, 1).Range.Text, 5) = "Varas" Then
            Taulu.Columns(3).Delete
            Taulu.Columns(3).Delete
            Taulu.Columns(3).Delete
            Taulu.Columns(3).Delete
            Taulu.Columns(3).Delete
            Taulu.Columns(3).Delete
        End If
    Next Taulu

End Function
```

Kuva 62. TrimmaaTaulut-funktio.

Raportin_tulostus -moduulista löytyy neljä Excelin Dataa-välilehden muokkaamiseen liittyvää makroa. Näillä makroilla pyritään muodostamaan listat kaikista laskurin perusvirtauksista, jotka aiheuttavat päästöjä sekä laskea päästöt elinkaarivaihekohtaisesti ja päästökategoriakohtaisesti. Näistä listoista myös määritellään alueet, jotka sisältävät sellaisen määrän virtauksia, elinkaaren vaiheita tai päästökategorioita, jotka muodostavat vähintään 80 % kyseisen luokan yhteenlasketuista päästöistä. Nämä makrot esitellään selkeyden vuoksi omassa kappaleessaan 4.3.7.

Viimeisenä ”aliohjelmana” moduulissa on VirheTiedostokasittelyssa. Nimensä mukaisesti tämä toimii virheenkäsittelijänä, mikäli ohjelma syystä tai toisesta kaatuu. Virheenkäsittely tallentaa, sulkee ja lopettaa aktiiviset applikaatiot ja asettaa objektimuuttujat tyhjiksi sekä antaa käyttäjälle ilmoituksen epäonnistuneesta raportin tulostuksesta. Kuvakaappaus tästä makrosta on nähtävissä kuvassa 63.

```
Private Function VirheTiedostokasittelyssa(wdApp As Word.Application, wdDok As Word.Document)

    wdApp.ActiveDocument.Save
    wdApp.ActiveDocument.Close
    wdApp.Quit

    Set wdApp = Nothing
    Set wdDok = Nothing

    Unload OdotusIkkuna

    MsgBox "Raportin tulostus epäonnistui.", vbOKOnly, "Virhe"

End Function
```

Kuva 63. VirheTiedostokasittelyssa-funktio.

Raportin_tulostus -moduulin niin sanottu pääohjelma TulostettavaRaportti on niin pitkä, että se esitellään kuudessa osassa. Ensimmäinen osa on nähtävissä kuvassa 64 ja se sisältää muuttujien määrittelyt, välilehtien määrittelyt oikeisiin muuttujiin, odotusikkunan avauksen ja raporttipohjan kopioinnin.

```
Sub TulostettavaRaportti()

    ' Tässä ohjelmassa luodaan laskennan tuloksista raportti.

    Dim polku As String
    Dim kopio As String
    Dim RaportinNimi As String
    Dim FSO As Object
    Dim i As Integer
    Dim sh As Worksheet
    Dim shD As Worksheet
    Dim shYht As Worksheet
    Dim shRaa As Worksheet
    Dim shPak As Worksheet
    Dim shJat As Worksheet
    Dim shEner As Worksheet
    Dim shLog As Worksheet
    Dim wdApp As Word.Application
    Dim wdDok As Word.Document

    Set sh = ThisWorkbook.Sheets("IKE-Laskentamenetelmä")
    Set shD = ThisWorkbook.Sheets("Dataa")
    Set shYht = ThisWorkbook.Sheets("Yhteenveto")
    Set shRaa = ThisWorkbook.Sheets("Raaka-aineet")
    Set shPak = ThisWorkbook.Sheets("Pakkausmateriaalit")
    Set shJat = ThisWorkbook.Sheets("Jätteet ja sivuvirrat")
    Set shEner = ThisWorkbook.Sheets("Energia")
    Set shLog = ThisWorkbook.Sheets("Logistiikka")

    OdotusIkkuna.Show vbModeless

    Call VapautaKaikki
    ' Tehdään raporttipohjasta kopio ja nimeä se tuotteen nimen perusteella

    Application.ScreenUpdating = False
    RaportinNimi = sh.Range("TuoNimi").Value & " - hiilijalanjälkiraportti" & ".docx"
    polku = Application.ActiveWorkbook.Path & "\IKE-hiilijalanjälkilaskurin raportin pohja.docx"
    kopio = Application.ActiveWorkbook.Path & "\" & RaportinNimi
    'Jos raporttipohjaa ei löydy, anna virheilmoitus.
    If TarkistaLoytyykoPohja(polku) = False Then
        Application.ScreenUpdating = True
        Exit Sub
    End If
```

Kuva 64. TulostettavaRaportti-makro (1/6).

Makron alun koodeista voidaan mainita, että eri välilehdille tehtiin lyhyemmät muuttujat, jotta koodin pituus lyhenisi ja yksinkertaistuisi. Erikoisempana muuttujana on mukana FSO eli niin sanottu File System Object, jota hyödynnetään tiedostojen käsittelyssä VBA:n avulla. Määrittelyjen jälkeen kaikki työkirjan suojaukset vapautetaan ja aloitetaan raporttipohjan kopioimisella. Tämä tehdään siksi, että alkuperäistä raporttipohjaa voidaan edelleen hyödyntää uusien raporttien tekemiseen, eikä tyhjää pohjaa näin ollen tarvitse aina ladata uudestaan.

Raporttipohjan kopioiminen aloitetaan määrittämällä haluttu kopiotiedoston nimi RaportinNimi-muuttujaan. Tämä tapahtuu hakemalla etusivulta käyttäjän määrittelemä tuotteen nimi, ja yhdistämällä se tekstin ” – hiilijalanjälkiraportti” ja ”.docx” kanssa, jolloin nimi on

muotoa ”Tuotteen nimi – hiilijalanjälkiraportti.docx”. Tämän jälkeen määritellään raporttipohjan poluksi sama polku, jossa laskuri sijaitsee lisättynä tiedoston nimellä. Samalla tavalla määritellään myös kopion polku. Tähän perään on vielä lisätty virheentarkistus, mikäli raporttipohjaa ei määritellyllä polulla löydy.

Kuvassa 65 on esitetty toinen osa TulostettavaRaportti-makrosta, jossa varmistetaan, että kopioksi määritetty polku ei ole jo käytössä. Näin voi tapahtua esimerkiksi silloin, kun samannimisestä tuotteesta syystä tai toisesta halutaan tehdä useita raportteja. Makro on rakennettu siten, että samannimisen kopion ollessa olemassa lisätään tiedoston nimen loppuun järjestysnumero. Makro voi tehdä raportteja aina järjestysnumeroon 999 asti, jonka jälkeen makro ei enää osaa käsitellä tilannetta. Järjestysnumeron lisääminen tiedoston nimen loppuun toteutetaan Do Until -silmukalla, jonka sisällä ehtorakenteen avulla tiedoston nimestä korvataan loppuosa uudella järjestysnumerolla, mikäli saman niminen tiedosto jo kansiosta löytyy.

Kun pätevä tiedoston nimi ja polku kopiolle on varmistettu, voidaan luoda raporttipohjasta kopio FSO-objektia hyödyntäen. Tämän jälkeen vielä avataan juuri luotu kopiotiedosto muokkaamista varten.

```
' Jos tiedostonimi on jo käytössä, lisätään nimen loppuun suluissa järjestysnumero.
i = 1
Do Until TarkistaTiedostoNimi(kopio) = False
    If i = 1 Then
        kopio = Left(kopio, Len(kopio) - 5) & "(" & i & ").docx"
        RaportinNimi = Left(RaportinNimi, Len(RaportinNimi) - 5) & "(" & i & ").docx"
    ElseIf i < 10 And i > 1 Then
        kopio = Left(kopio, Len(kopio) - 8) & "(" & i & ").docx"
        RaportinNimi = Left(RaportinNimi, Len(RaportinNimi) - 8) & "(" & i & ").docx"
    ElseIf i < 100 And i > 10 Then
        kopio = Left(kopio, Len(kopio) - 9) & "(" & i & ").docx"
        RaportinNimi = Left(RaportinNimi, Len(RaportinNimi) - 9) & "(" & i & ").docx"
    End If
    i = i + 1
    If i > 1000 Then Exit Do
Loop

'Kopioidaan tiedosto ja nimetään se
Set FSO = CreateObject("Scripting.FileSystemObject")
FSO.CopyFile polku, kopio, True
Set FSO = Nothing

' Avataan muokattavaksi.

On Error GoTo Virhe

Set wdApp = CreateObject(Class:="Word.Application")

On Error GoTo 0

Set wdDok = wdApp.Documents.Open(kopio)
```

Kuva 65. TulostettavaRaportti-makro (2/6).

Kun kopiotiedosto on avattu muokattavaksi, voidaan aloittaa varsinainen raportin tietojen täyttäminen. Tämä on pääohjelmassa karkeasti jaettu siten, että ensin lisätään tekstit ja tämän jälkeen järjestyksessä taulukot ja kuvat, kuten ne raporttiin tulevat. Tällä tavalla vältetään virheitä muotoiluissa, mikäli tekstejä, taulukoita tai kuvia lisäiltäisiin satunnaisessa järjestyksessä. Kuvasta 66 voidaan nähdä TulostettavaRaportti-makron osio, jossa suurin osa tekstin korvaamisesta hoidetaan. Ensin lisätään suoraviivaisesti aina korvattavat tekstit, ja sen jälkeen ehtorakenteiden avulla korvataan tekstejä eritavoin riippuen sivuvirtojen olemassaolosta ja käytetystä allokointimenetelmästä. Jos sivuvirtoja löytyy, lisätään myös sivuvirroista taulukko.

```
' Täytä tiedot
Call EtsiJaKorvaaTekstia(wdDok, "<TuotteenNimi>", sh.Range("TuoNimi"))
Call EtsiJaKorvaaTekstia(wdDok, "<EräNumero>", sh.Range("TuoEraNmr"))
Call EtsiJaKorvaaTekstia(wdDok, "<YrityksenNimi>", sh.Range("YrityksenNimi"))
Call EtsiJaKorvaaTekstia(wdDok, "<Valmistuspaikka>", sh.Range("TuoValmistusPaikka"))
Call EtsiJaKorvaaTekstia(wdDok, "<Päivämäärä>", sh.Range("TuoLaskentaPvm"))
Call EtsiJaKorvaaTekstia(wdDok, "<Maat>", sh.Range("TuoMaat"))
Call EtsiJaKorvaaTekstia(wdDok, "<JärjestelmänRajat>", sh.Range("JarjestelmanKuvaus"))
Call EtsiJaKorvaaTekstia(wdDok, "<RajauksetJaOletukset>", sh.Range("Rajaukset"))
Call EtsiJaKorvaaTekstia(wdDok, "<TietojenLaatu>", sh.Range("TietojenLaatu"))
Call EtsiJaKorvaaTekstia(wdDok, "<Laskija>", sh.Range("Laskija"))
Call EtsiJaKorvaaTekstia(wdDok, "<Esiselvitys>", sh.Range("Esiselvitys"))

Call EtsiJaKorvaaTekstia(wdDok, "<Hiilijalanjälki>", shYht.Range("FosPaastot"))
Call EtsiJaKorvaaTekstia(wdDok, "<HiilijalanjälkiPerKg>", shYht.Range("FosPaastotPerKg"))
Call EtsiJaKorvaaTekstia(wdDok, "<BiogeenisetPäästöt>", shYht.Range("BioPaastot"))
Call EtsiJaKorvaaTekstia(wdDok, "<BiogeenisetPäästotPerKg>", shYht.Range("BioPaastotPerKg"))
Call EtsiJaKorvaaTekstia(wdDok, "<BiogeenistenPäästöjenOsuus>", shD.Range("BioOsuusRaportille"))

If shD.Range("SivMaara") > 0 Then
    Call EtsiJaKorvaaTekstia(wdDok, "<Sivuvirrat>", shD.Range("SivSelitys"))
    Call EtsiJaKorvaaTekstia(wdDok, "<Sivuvirrat2>", shD.Range("SivSelitys2"))

    If shYht.Shapes("MassaAllokointiNappi").Fill.ForeColor.RGB = RGB(251, 173, 24) Then
        Call EtsiJaKorvaaTekstia(wdDok, "<AllokointiMenetelmä>", shD.Range("MassaAllokointi"))
        Call EtsiJaKorvaaTekstia(wdDok, "<Allokointiperuste>", shD.Range("AllokointiperusteMassa"))
    Else
        Call EtsiJaKorvaaTekstia(wdDok, "<AllokointiMenetelmä>", shD.Range("TalousAllokointi"))
        Call EtsiJaKorvaaTekstia(wdDok, "<Allokointiperuste>", shD.Range("AllokointiperusteTalous"))
    End If

    Call EtsiJaKorvaaTekstia(wdDok, "<SivuvirtojenMäärä>", shD.Range("SivMaara"))
    Call LisaaTaulukko(wdApp, wdDok, "Sivuvirrat", "Sivuvirrat_Pohja", "Sivuvirrat_alku", _
    shYht.Range("SivutuotteetAlku").CurrentRegion)

Else
    Call EtsiJaKorvaaTekstia(wdDok, "<Sivuvirrat>", shD.Range("A1000000"))
    Call EtsiJaKorvaaTekstia(wdDok, "<Sivuvirrat2>", shD.Range("A1000000"))
    Call EtsiJaKorvaaTekstia(wdDok, "Taulukko 1. Laskennassa huomioidut sivutuotteet.", shD.Range("A1000000"))
    Call EtsiJaKorvaaTekstia(wdDok, "Taulukoissa 2, 3 ja 4", shD.Range("TaulVaihtoehtoinenTeksti"))
    Call EtsiJaKorvaaTekstia(wdDok, "Taulukko 2.", shD.Range("Taul1"))
    Call EtsiJaKorvaaTekstia(wdDok, "Taulukko 3.", shD.Range("Taul2"))
    Call EtsiJaKorvaaTekstia(wdDok, "Taulukko 4.", shD.Range("Taul3"))
    Call EtsiJaKorvaaTekstia(wdDok, "Taulukko 5.", shD.Range("Taul4"))
    Call EtsiJaKorvaaTekstia(wdDok, "Taulukossa 5", shD.Range("TaulVaihtoehtoinenTeksti2"))
End If
```

Kuva 66. TulostettavaRaportti-makro (3/6).

Tekstin lisäämisen jälkeen lisätään normaaleihin kappaleisiin lisättävät kuvat ja taulukot. Tässä otetaan huomioon vaihtoehtoiset tekstit, mikäli biogeenisiä päästöjä ei ollenkaan synny. Koodi esitetty kuvassa 67.

```
Call DatanPaivittaja

Call LisaaTaulukko(wdApp, wdDok, "Merkittävimmät_perusvirrat", "Merkittävimmät_päästökategoriat_alku", _
"Merkittävimmät_perusvirrat_alku", Alue80Paastoille("DataPerusvirrat", 4))
Call LisaaTaulukko(wdApp, wdDok, "Merkittävimmät_päästökategoriat", "Merkittävimmät_elinkaarenvaiheet_alku", _
"Merkittävimmät_päästökategoriat_alku", Alue80Paastoille("DataPaastokategoriat", 3))
Call LisaaTaulukko(wdApp, wdDok, "Merkittävimmät_elinkaarenvaiheet", "Merkittävimmät_elinkaarenvaiheet_pohja", _
"Merkittävimmät_elinkaarenvaiheet_alku", Alue80Paastoille("DataElinkaarenvaiheet", 3))
Call LisaaTaulukko(wdApp, wdDok, "Yhteenveto", "Kuva2", "Yhteenveto_alku", shYht.Range("YhteenvetoTaulu"))
Call LisaaKuva(wdApp, wdDok, "Kuva2", "Kuva3", "Kuva2", shYht.ChartObjects("YhteenvetoPalkkiKaavio"))
Call LisaaKuva(wdApp, wdDok, "Kuva3", "Kuva4", "Kuva3", shYht.ChartObjects("YhteenvetoPiirakkaKaavio"))

If shYht.Range("BioOsuus") > 0 Then
    Call EtsiJaKorvaaTekstia(wdDok, "<BioKaaviotSelitys>", shD.Range("BioKaaviotSelitys"))
    Call LisaaKuva(wdApp, wdDok, "Kuva4", "Kuva5", "Kuva4", shYht.ChartObjects("YhteenvetoBioPalkkiKaavio"))
    Call LisaaKuva(wdApp, wdDok, "Kuva5", "Kuva5_pohja", "Kuva5", shYht.ChartObjects("YhteenvetoBioPiirakkaKaavio"))
Else
    Call EtsiJaKorvaaTekstia(wdDok, "<BioKaaviotSelitys>", shD.Range("A1000000"))
    Call EtsiJaKorvaaTekstia(wdDok, "Kuva 4. Päästökategorioiden osuudet biogeenisistä " & _
    "päästöistä palkkikaaviolla esitettynä.", shD.Range("A1000000"))
    Call EtsiJaKorvaaTekstia(wdDok, "Kuva 5. Päästökategorioiden osuudet biogeenisistä " & _
    "päästöistä ympyräkaaviolla esitettynä.", shD.Range("A1000000"))
End If

Call EtsiJaKorvaaTekstia(wdDok, "<LisätiedotTuloksista>", sh.Range("Lisatiedot"))

Call VapautaKaikki
```

Kuva 67. TulostettavaRaportti-makro (4/6).

Seuraavaksi kuvan 68 mukaisesti TulostettavaRaportti-makrossa lisätään liitteisiin tarvittavat kuvat ja taulukot. Viimeisen taulukon lisäämisen jälkeen siistitään taulukoiden ulkoasu TrimmaaTaulut-funktiolla, tallennetaan tiedosto, avataan raportti koko näytön kokoiseksi, siirrytään raportin ensimmäiselle sivulle ja asetetaan raporttitiedosto näkyväksi.

```
Call LisaaKuva(wdApp, wdDok, "Liite1", "Liite2_alku", "Liite1", shRaa.ChartObjects("RaaYmpyra"))
Call LisaaTaulukko(wdApp, wdDok, "Liite1", "Liite2_alku", "Liite1_alku", shRaa.Range("RaaRaakaaineet").CurrentRegion)
Call LisaaKuva(wdApp, wdDok, "Liite2", "Liite3_alku", "Liite2", shPak.ChartObjects("PakYmpyra"))
Call LisaaTaulukko(wdApp, wdDok, "Liite2", "Liite3_alku", "Liite2_alku", shPak.Range("Pakkausmateriaalit").CurrentRegion)
Call LisaaKuva(wdApp, wdDok, "Liite3a", "Liite3b_alku", "Liite3a", shJat.ChartObjects("JatYmpyra"))
Call LisaaTaulukko(wdApp, wdDok, "Liite3a", "Liite3b_alku", "Liite3a_alku", shJat.Range("JatJatteetjaSivuvirrat").CurrentRegion)
Call LisaaKuva(wdApp, wdDok, "Liite3b", "Liite4a_alku", "Liite3b", shJat.ChartObjects("PesAYmpyra"))
Call LisaaTaulukko(wdApp, wdDok, "Liite3b", "Liite4a_alku", "Liite3b_alku", shJat.Range("PesuAineet").CurrentRegion)
Call LisaaKuva(wdApp, wdDok, "Liite4a", "Liite4b_alku", "Liite4a", shEner.ChartObjects("EJakYmpyra"))
Call LisaaTaulukko(wdApp, wdDok, "Liite4a", "Liite4b_alku", "Liite4a_alku", shEner.Range("Energianlahde").CurrentRegion)
Call LisaaKuva(wdApp, wdDok, "Liite4b", "Liite4c_alku", "Liite4b", shEner.ChartObjects("SahYmpyra"))
Call LisaaTaulukko(wdApp, wdDok, "Liite4b", "Liite4c_alku", "Liite4b_alku", shEner.Range("SahkoLaitteet").CurrentRegion. _
Resize(shEner.Range("SahkoLaitteet").CurrentRegion.Rows.Count, 8))
Call LisaaKuva(wdApp, wdDok, "Liite4c", "Liite4d_alku", "Liite4c", shEner.ChartObjects("MuutLYmpyra"))
Call LisaaTaulukko(wdApp, wdDok, "Liite4c", "Liite4d_alku", "Liite4c_alku", shEner.Range("MuutLaitteet").CurrentRegion)
Call LisaaKuva(wdApp, wdDok, "Liite4d", "Liite4e_alku", "Liite4d", shEner.ChartObjects("HoyryYmpyra"))
Call LisaaTaulukko(wdApp, wdDok, "Liite4d", "Liite4e_alku", "Liite4d_alku", shEner.Range("Höyry").CurrentRegion)
Call LisaaTaulukko(wdApp, wdDok, "Liite4e", "Liite5a_alku", "Liite4e_alku", shEner.Range("PesVesLammitysEner").CurrentRegion. _
Resize(shEner.Range("PesVesLammitysEner").CurrentRegion.Rows.Count, 8))
Call LisaaKuva(wdApp, wdDok, "Liite5a", "Liite5b_alku", "Liite5a", shLog.ChartObjects("JakYmpyra"))
Call LisaaTaulukko(wdApp, wdDok, "Liite5a", "Liite5b_alku", "Liite5a_alku", shLog.Range("AjoneuvonTyyppi").CurrentRegion. _
Resize(shLog.Range("AjoneuvonTyyppi").CurrentRegion.Rows.Count, 12))
Call LisaaKuva(wdApp, wdDok, "Liite5b", "Liite5c_alku", "Liite5b", shLog.ChartObjects("VarYmpyra"))
Call LisaaTaulukko(wdApp, wdDok, "Liite5b", "Liite5c_alku", "Liite5b_alku", shLog.Range("Varastointi").CurrentRegion. _
Resize(shLog.Range("Varastointi").CurrentRegion.Rows.Count, 14))
Call LisaaKuva(wdApp, wdDok, "Liite5c", "Liite5c_pohja", "Liite5c", shLog.ChartObjects("KylYmpyra"))
Call LisaaTaulukko(wdApp, wdDok, "Liite5c", "Liite5c_pohja", "Liite5c_alku", shLog.Range("KylmaKaasut").CurrentRegion)

Call TrimmaaTaulut(wdApp, wdDok)

wdApp.ActiveDocument.Save
wdDok.Activate
wdApp.ActiveWindow.WindowState = wdWindowStateMaximize
wdApp.Selection.GoTo What:=wdGoToLine, which:=wdGoToAbsolute, Count:=1
wdApp.Visible = True
```

Kuva 68. TulostettavaRaportti-makro (5/6).

TulostettavaRaportti-makron viimeisessä osassa käsitellään kaksi vaihtoehtoista tapaa avata Word tiedosto päällimmäiseksi näkyville. Käytännössä ensin siis kokeillaan komentoa AppActivate RaportinNimi ja mikäli tämä aiheuttaa virheen, yritetään seuraavaksi komentoa AppActivate wd.Caption. Kumpaa komentoa sitten käytetäänkään, ohjelma sulkee odotusikkunan, aktivoi yhteenvetosivun laskurista, nollaa wdApp- ja wdDok-muuttujat ja suojaa välilehdet. Aivan lopuksi makrossa on vielä virheenkäsittelijä, joka niin ikään lopettaa ohjelman toiminnan, nollaa muuttujat ja antaa käyttäjälle virheilmoituksen. Nämä toiminnot ovat nähtävissä kuvasta 69.

```
    On Error GoTo wordesittely2

    AppActivate RaportinNimi

    Unload OdotusIkkuna

    Call SuojaaKaikki

    ThisWorkbook.Sheets("Yhteenveto").Activate
    Application.ScreenUpdating = True

    Set wdApp = Nothing
    Set wdDok = Nothing
    On Error GoTo 0
    Exit Sub

wordesittely2:

    On Error GoTo Virhe

    AppActivate wdApp.Caption

    Unload OdotusIkkuna

    Call SuojaaKaikki

    ThisWorkbook.Sheets("Yhteenveto").Activate
    Application.ScreenUpdating = True

    Set wdApp = Nothing
    Set wdDok = Nothing
    On Error GoTo 0
    Exit Sub

Virhe:

    Call SuojaaKaikki

    ThisWorkbook.Sheets("Yhteenveto").Activate
    Application.ScreenUpdating = True
    Set wdApp = Nothing
    Set wdDok = Nothing
    MsgBox "Virhe Word tiedoston avaamisessa. Tämä virhe saattaa johtua siitä, " _
    & "että makrojen toiminta on estetty Wordissä.", vbOKOnly, "Virhe"

    On Error GoTo 0

End Sub
```

Kuva 69. TulostettavaRaportti-makro (6/6).

4.3.7 Dataa -välilehden päivitys

Raportin tulostusta varten käyttäjältä piilotetulle Dataa-välilehdelle on muodostettu kolme listaa laskennan merkittävimmistä yksittäisistä virtauksista, elinkaaren vaiheista sekä laskurissa käytetyistä päästökategorioista. (Euroopan komissio, 2021, s. 92.) PEF-ohjeistuksen mukaisesti virtaukset, päästökategoriat ja elinkaaren vaiheet järjestetään ensin suuruusjärjestykseen, jonka jälkeen näistä merkittäviksi luetaan ne, jotka suurimmasta pienimpään kumulatiivisesti yhteen laskien muodostavat vähintään 80 prosenttia tuotteen hiilijalanjäljestä. Nämä taulukot päivitetään aina raportin tulostuksen yhteydessä, sillä näiden jatkuva päivittäminen virtausten lisäämisen yhteydessä hidastaisi merkittävästi ohjelman toimintaa.

Näiden taulukoiden päivittämiseen käytetään neljää makroa: DatanTyhjennys, DatanLisays, DatanPaivittaja ja Alue80Paastoille. Näistä ensimmäinen pyyhkii taulukoista vanhat tiedot pois, toinen on yksittäisen laskentavälilehden taulukon datan lisäämiseen tarkoitettu funktio,

kolmas itse Dataa-välilehden taulukoiden päivittämisen suorittava makro ja viimeinen valmiin taulukon alueen palauttava funktio. Aloitetaan näiden makrojen läpikäynti tässä järjestyksessä ja kuvasta 70 onkin nähtävissä DatanTyhjennys-makro.

```
Private Function DatanTyhjennys()

    Call VapautaSivu("Dataa")

    Dim shDSivu As Worksheet
    Dim rngTaul As Range
    Dim VirtojenMaara As Integer
    Dim i As Integer

    Set shDSivu = ThisWorkbook.Sheets("Dataa")
    Set rngTaul = shDSivu.Range("DataPerusvirrat")

    'Poistetaan aiemmat rivit.
    If rngTaul.CurrentRegion.Rows.Count > 2 Then
        Range(rngTaul.Offset(2, 0), rngTaul.End(xlDown).End(xlToRight)).ClearContents
    End If

    Call SuojaaSivu("Dataa")

End Function
```

Kuva 70. DatanTyhjennys-makro.

DatanTyhjennys-makro ei ole toimintaperiaatteeltaan kovinkaan monimutkainen. Sivu vapautetaan muokattavaksi, määritetään muuttujat ja suoritetaan alueen tyhjennys .CurrentRegion-ominaisuutta hyödyntäen. Lopuksi sivu suojataan ja tyhjennys on valmis. Tyhjennysmakrossa huomioitavaa on, että se tyhjentää ainoastaan perusvirtausten taulukon tiedot, eikä koske päästökategorioiden ja elinkaaren vaiheiden taulukoihin ollenkaan. Tämä johtuu siitä, että näiden kahden taulukon sisältö on aina samanlainen laskuriin syötetyistä tiedoista huolimatta, joten niitä ei ole tarvetta poistaa ja lisätä aina uudestaan.

Kun vanha data on taulukoista tyhjennetty, voidaan lisätä päivitettyä tietoa. Tähän hyödynnetään DatanLisays-funktiota, joka ottaa syötteenä virtauksen välilehden nimen, taulukon otsikon, päästöjen otsikon, päästöjen osuuden sarakkeen otsikon, lukumäärän, yksikön sekä valinnaisena muuttujana montako tuotetta kyseisellä päästömäärällä käsiteltiin. DatanLisays-funktio on hyvin suuri johtuen siitä, että sen tulee osata ottaa huomioon paljon erilaisia poikkeavuuksia datan laadussa ja alkuperässä. Funktion koodi on pilkottu nähtäville kuviin 71, 72 ja 73.

```
Private Function DatanLisays(Sivu As Worksheet, Otsikko As String, Paastot As String, _
PaastoOsuus As String, Maara As String, Yks As String, Optional ByVal MontaTuotetta As String = "0")

    Call VapautaSivu("Dataa")

    Dim rngTaul As Range
    Dim rngOtsikko As Range
    Dim rngPaastot As Range
    Dim rngPaastoOsuus As Range
    Dim rngMaara As Range
    Dim shDSivu As Worksheet
    Dim shSivu As Worksheet
    Dim VirtojenMaara As Integer
    Dim i As Integer
    Dim rnglkm As Range
    Dim ohitus1 As Boolean
    Dim ohitus2 As Boolean
    Dim ohitus3 As Boolean


    Set shSivu = Sivu
    Set shDSivu = ThisWorkbook.Sheets("Dataa")
    Set rngTaul = shDSivu.Range("DataPerusvirrat")
    Set rngOtsikko = shSivu.Range(Otsikko)
    Set rngPaastot = shSivu.Range(Paastot)
    Set rngPaastoOsuus = shSivu.Range(PaastoOsuus)
    Set rngMaara = shSivu.Range(Maara)

    If MontaTuotetta = "0" Then
        Set rnglkm = Nothing
    Else
        Set rnglkm = shSivu.Range(MontaTuotetta)
    End If

    VirtojenMaara = Range(rngOtsikko, rngOtsikko.End(xlDown)).Rows.Count - 2

    ' Jos pesuvesien lämmityksellä on päästöjä, lisätään sillekin oma rivi laatikkoon.
    If shSivu.Name = "Energia" And ThisWorkbook.Sheets("Energia").Range("PesuvesienPaastotOsuus").Value > 0 _
    And Not Otsikko = "MuutLaitteet" And Not Otsikko = "Hoyry" Then
        rngTaul.End(xlDown).Offset(1, 0).Value = "Pesuvesien lämmitykseen kuluva energia"
        rngTaul.End(xlDown).Offset(0, 1).Value = _
        shSivu.Range("PesuvesienPaastotOsuus").Offset(0, -5).Value & " " & Yks
        rngTaul.End(xlDown).Offset(0, 2).Value = _
        shSivu.Range("PesuvesienPaastotOsuus").Offset(0, -2).Value
        rngTaul.End(xlDown).Offset(0, 3).Value = _
        shSivu.Range("PesuvesienPaastotOsuus").Value
    End If
```

Kuva 71. DatanLisays-funktio (1/3).

DatanLisays-funktio alkaa Dataa-välilehden vapauttamisella ja muuttujien määrittämisellä. Tämän jälkeen testataan ehtorakenteella, onko vapaavalintaiseen MontaTuotetta-muuttujaan määritetty haluttua saraketta. Jos on, määritetään tämän avulla alue rnglkm-muuttujaan, mutta muutoin jatketaan eteenpäin. Määritellään virtausten määrä VirtojenMaara-muuttujaan ja lisätään taulukkoon rivi pesuvesien lämmityksestä johtuville päästöille, mikäli sellaisia on.

```
    If Otsikko = "AjoneuvonTyyppi" Then
        If shSivu.Range("LogYriArvo1").Offset(0, 1) <> "" Then
            ohitus1 = True
            rngTaul.End(xlDown).Offset(1, 0).Value = _
            "Kuljetusyhtiön ilmoittamat päästöt raaka-aineiden kuljetukselle"
            rngTaul.End(xlDown).Offset(0, 1).Value = "-"
            rngTaul.End(xlDown).Offset(0, 2).Value = shDSivu.Range("Log1OsuusPaastoista").Offset(0, -1)
            rngTaul.End(xlDown).Offset(0, 3).Value = shDSivu.Range("Log1OsuusPaastoista")
        End If
        If shSivu.Range("LogYriArvo2").Offset(0, 1) <> "" Then
            ohitus2 = True
            rngTaul.End(xlDown).Offset(1, 0).Value = _
            "Kuljetusyhtiön ilmoittamat päästöt pakkausmateriaalien kuljetukselle"
            rngTaul.End(xlDown).Offset(0, 1).Value = "-"
            rngTaul.End(xlDown).Offset(0, 2).Value = shDSivu.Range("Log2OsuusPaastoista").Offset(0, -1)
            rngTaul.End(xlDown).Offset(0, 3).Value = shDSivu.Range("Log2OsuusPaastoista")
        End If
        If shSivu.Range("LogYriArvo3").Offset(0, 1) <> "" Then
            ohitus3 = True
            rngTaul.End(xlDown).Offset(1, 0).Value = "Kuljetusyhtiön ilmoittamat päästöt tuotteiden kuljetukselle"
            rngTaul.End(xlDown).Offset(0, 1).Value = "-"
            rngTaul.End(xlDown).Offset(0, 2).Value = shDSivu.Range("Log3OsuusPaastoista").Offset(0, -1)
            rngTaul.End(xlDown).Offset(0, 3).Value = shDSivu.Range("Log3OsuusPaastoista")
        End If
    End If

    If VirtojenMaara <= 1 And rngOtsikko.Offset(1, 0) = "" Then
        Call SuojaaSivu("Dataa")
        Exit Function
    End If
```

Kuva 72. DatanLisays-funktio (2/3).

Seuraavaksi DatanLisays-funktiossa tutkitaan, onko kyseessä ajoneuvojen taulukko ja jos on, niin onko yrityksen itse ilmoittamiin päästöihin merkitty jotakin johonkin kolmesta osamatkasta. Jos on, näistä merkitään Dataa-välilehden taulukkoon omat rivinsä ja lisätään ohitus muuttujaan arvo True, jotta ajoneuvolistasta ei myöhemmin lisätä kyseisestä osamatkasta virtoja. Tämän jälkeen testataan vielä ennen varsinaista tietojen täyttämistä, että onko käsiteltävässä taulukossa virtauksia ollenkaan. Jos ei ole, lopetetaan funktion toiminta tähän.

```
    ' Täytetään listaan uudet tiedot. Täytetään tietoja lisäämällä rivejä ja täyttämällä edelliseltä riviltä
    ' kaavat uudelle for-loopin sisällä niin monella kierroksella, kuin on raaka-aineitakin.
    For i = 1 To VirtojenMaara

        If ohitus1 = True And rngOtsikko.Offset(i, 1) = "Osamatka 1" Then
            'Ei tehdä mitään
        ElseIf ohitus2 = True And rngOtsikko.Offset(i, 1) = "Osamatka 2" Then
            'Ei tehdä mitään
        ElseIf ohitus3 = True And rngOtsikko.Offset(i, 1) = "Osamatka 3" Then
            'Ei tehdä mitään
        Else
            rngTaul.End(xlDown).Offset(1, 0).Value = rngOtsikko.Offset(i, 0)
            If rnglkm Is Nothing Then
                rngTaul.End(xlDown).Offset(0, 1).Value = Round(rngMaara.Offset(i, 0).Value, 2) & " " & Yks
            Else
                rngTaul.End(xlDown).Offset(0, 1).Value = _
                Round(rngMaara.Offset(i, 0).Value / rnglkm.Offset(i, 0).Value, 2) & " " & Yks
            End If
            rngTaul.End(xlDown).Offset(0, 2).Value = rngPaastot.Offset(i, 0)
            rngTaul.End(xlDown).Offset(0, 3).Value = rngPaastoOsuus.Offset(i, 0)
        End If
    Next i

    Call SuojaaSivu("Dataa")

End Function
```

Kuva 73. DatanLisays-funktio (3/3).

Funktion lopussa tehdään varsinainen tietojen lisääminen Dataa-välilehden perusvirtausten taulukkoon. Tämä toteutetaan For-silmukalla, jossa testataan ensin ehtorakenteen avulla, tarvitseeko ohittaa silmukan kierrosta yrityksen ilmoittamien päästötietojen takia ajoneuvojen tapauksessa. Jos ei tarvitse, lisätään Dataa-välilehden taulukkoon uusi rivi ja lisätään siihen tiedot laskentavälilehden taulukosta käsiteltävältä riviltä. Tässäkin tapauksessa testataan ehtorakenteen avulla, onko rnglkm-muuttujalla arvo, jolloin laskentakaava on hieman erilainen.

DatanPaivittaja-makro tekee varsinaisen Dataa-välilehden tietojen päivityksen kutsumalla DatanTyhjennys- ja DatanLisays-funktioita. Kuva DatanPaivittaja-makron koodista on nähtävissä kuvassa 74.

```
Sub DatanPaivittaja()

    Call VapautaSivu("Dataa")

    Dim rngTaulPv As Range
    Dim rngTaulPk As Range
    Dim rngTaulEv As Range

    Set rngTaulPv = ThisWorkbook.Sheets("Dataa").Range("DataPerusvirrat")
    Set rngTaulPk = ThisWorkbook.Sheets("Dataa").Range("DataPaastokategoriat")
    Set rngTaulEv = ThisWorkbook.Sheets("Dataa").Range("DataElinkaarenvaiheet")

    'Poistetaan vanhat tiedot ja päivitetään uudet
    Call DatanTyhjennys
    Call DatanLisays(ThisWorkbook.Sheets("Raaka-aineet"), _
    "RaaRaakaaineet", "RaaPaastot", "RaaOsuusPaastoista", "RaaMaara", "kg")
    Call DatanLisays(ThisWorkbook.Sheets("Pakkausmateriaalit"), _
    "Pakkausmateriaalit", "PakPaastot", "PakkauksetOsuusPaastoista", "PakMassaPerTuote", "kg")
    Call DatanLisays(ThisWorkbook.Sheets("Jätteet ja sivuvirrat"), _
    "JatJatteetjaSivuvirrat", "JatPaastot", "JatOsuusPaastoista", "JatMaara", "kg")
    Call DatanLisays(ThisWorkbook.Sheets("Jätteet ja sivuvirrat"), _
    "PesuAineet", "PesuAieidenPäästöt", "PesuaineidenOsuusPaastoista", _
    "PesuAineidenMassat", "kg", "PesuAineKäsittelyLKM")
    Call DatanLisays(ThisWorkbook.Sheets("Energia"), "SahkoLaitteet", _
    "SahkolaitteidenPaastotFos", "SahkoLaitteetOsuusPaastoista", "SahkolaitteetKulutus", "kWh")
    Call DatanLisays(ThisWorkbook.Sheets("Energia"), "MuutLaitteet", _
    "MuidenLaitteidenPaastotFos", "MuidenLaitteidenOsuusPaastoista", _
    "MuutLaitteetKulutus", "kg / l")
    Call DatanLisays(ThisWorkbook.Sheets("Energia"), "Höyry", "HöyryPäästötFos", _
    "HöyryOsuusPäästöistä", "HöyryMassa", "kg", "HöyryKäsiteltyLKM")
    Call DatanLisays(ThisWorkbook.Sheets("Logistiikka"), "AjoneuvonTyyppi", _
    "AjoneuvojenPaastot", "LogOsuusPaastoista", "MatkanPituus", "km")
    Call DatanLisays(ThisWorkbook.Sheets("Logistiikka"), "Varastointi", _
    "VasrastoinninPaastotFos", "VarastointiOsuusPaastoista", "VarastoinninEnergia", "kWh")
    Call DatanLisays(ThisWorkbook.Sheets("Logistiikka"), "KylmaKaasut", _
    "KylmaKaasutPaastot", "KylAinOsuusPaastoista", "KylAinKulutus", "kg")

    VapautaSivu ("Dataa")
    'Järjestetään suuruus järjestykseen
    If rngTaulPv.CurrentRegion.Rows.Count > 2 Then
        Range(rngTaulPv.Offset(2, 0), rngTaulPv.End(xlDown).End(xlToRight)). _
        Sort Key1:=rngTaulPv.Offset(2, 3), Order1:=xlDescending, Header:=xlNo
    End If
    If rngTaulPk.CurrentRegion.Rows.Count > 2 Then
        Range(rngTaulPk.Offset(2, 0), rngTaulPk.End(xlDown).End(xlToRight)). _
        Sort Key1:=rngTaulPk.Offset(2, 2), Order1:=xlDescending, Header:=xlNo
    End If
    If rngTaulEv.CurrentRegion.Rows.Count > 2 Then
        Range(rngTaulEv.Offset(2, 0), rngTaulEv.End(xlDown).End(xlToRight)). _
        Sort Key1:=rngTaulEv.Offset(2, 2), Order1:=xlDescending, Header:=xlNo
    End If

    Call SuojaaSivu("Dataa")

End Sub
```

Kuva 74. DatanPaivittaja-makro.

DatanPaivittaja-makro alkaa sivun vapauttamisella ja muuttujien määrittämisellä. Jokaiselle päivitettävälle taulukolle määritetään niille kuuluva alue omaan Range-muuttujaan, jonka jälkeen tyhjennetään vanhat tiedot taulukoista DatanTyhjennys-funktiolla ja aloitetaan perusvirtausten päivittäminen laskentataulukko kerrallaan DatanLisays-funktion avulla. Lopuksi vielä järjestetään taulukoiden päästölähteet suuruusjärjestykseen.

Lopuksi käsitellään vielä Alue80Paastoille-funktio. Tämän funktion tarkoituksena on palauttaa Range-arvo, joka vastaa halutun taulukon aluetta, jossa yhteenlaskettu päästöjen osuus on vähintään 80 prosenttia. Tätä aluetta hyödynnetään raportin tulostustoiminnossa, kun halutaan lisätä raporttiin merkityksellisimpien perusvirtausten, päästökategorioiden ja elinkaaren vaiheiden taulukoita. Funktio tarvitsee syötteenä taulukon otsikon ja päästöosuussarakkeen sijainnin kokonaislukuna (monesko sarake otsikkosarakkeesta). Alue80Paastoille-funktion koodi on nähtävissä kuvasta 75.

```
Private Function Alue80Paastoille(Taul As String, PaastoOsuusSarake As Integer) As Range

    VapautaSivu ("Dataa")
    Dim rngTaul As Range
    Dim irow As Integer
    Dim Paastot As Single

    Set rngTaul = ThisWorkbook.Sheets("Dataa").Range(Taul)
    Paastot = 0
    irow = 1

    If rngTaul.CurrentRegion.Rows.Count <= 2 Then
        Set Alue80Paastoille = rngTaul.CurrentRegion
        Exit Function
    End If

    Do Until Paastot >= 0.8 Or irow > rngTaul.CurrentRegion.Rows.Count
        Paastot = Paastot + rngTaul.Offset(irow + 1, PaastoOsuusSarake - 1).Value
        irow = irow + 1
    Loop

    Set Alue80Paastoille = Range(rngTaul, rngTaul.Offset(irow, PaastoOsuusSarake - 1))

    SuojaaSivu ("Dataa")

End Function
```

Kuva 75. Alue80Paastoille-makro.

Alue80Paastoille-funktion toiminto aloitetaan normaaliin tapaan Dataa-välilehden vapauttamisella ja muuttujien määrittämisellä. Tämän jälkeen asetetaan rngTaul-muuttujaan koko halutun taulukon alue ja annetaan Paastot-muuttujalle arvoksi nolla ja irow-muuttujalle arvo 1. Tämän jälkeen vielä testataan ehtorakenteen avulla, onko taulukossa 2 riviä tai vähemmän, jolloin tästä rngTaul-muuttujan sisältämästä taulukosta tehdään funktion palauttava arvo ja lopetetaan funktion toiminta. Näin tehdään siksi, että mikäli rivejä on kaksi, tällöin koko taulukossa on otsikkorivin lisäksi vain yksi rivi, jolloin seuraava vaihe on funktion toiminnan kannalta merkityksetön.

Jos kuitenkin rivejä oli enemmän kuin kaksi, selvitetään missä kohtaa taulukon rivien sisältämät päästölähteet muodostavat yhteensä vähintään 80 prosenttia tuotteen kokonaispäästöistä. Tämä toteutetaan Do Until -tyyppisellä silmukkarakenteella, joka toistuu niin kauan, kunnes Paastot-muuttujan arvo on saavuttanut arvon 0,8 tai irow-muuttujan arvo on kasvanut yhtä suureksi kuin taulukon rivien lukumäärä. Jokaisella silmukan toistolla Paastot-muuttujan arvoon lisätään käsiteltävän rivin päästöjen osuuden arvo ja lisätään irow-muuttujaan yksi.

Kun silmukkarakenne on lopettanut, voidaan olettaa, että taulukon rivejä on käyty niin pitkälle, että päästöjen yhteenlaskettu osuus ylitti 80 prosenttia tai koko lista käytiin läpi. Tällöin voidaan asettaa irow-muuttujan arvon perusteella alue funktion arvoksi, joka käsittää ne rivit taulukosta, jossa rivien päästöosuuksien yhteenlaskettu lukumäärä on vähintään 80 prosenttia. Lopuksi vielä palautetaan sivun suojaus.

4.3.8 Rivien_hallinnointi

Rivien hallinnointiin liittyvä moduuli sisältää laskurin sisältämien taulukoiden rivien lisäämiseen tai poistamiseen liittyvät makrot. Rivin poistaminen on niin yksinkertainen operaatio, että se pystytään tekemään kaikille taulukoille hyödyntäen samaa RivinPoistaja(ListanYla As String, ListanPohja As String)-funktiota. Kuitenkin uuden rivin lisääminen on monimutkaisempi toimenpide, joten jokaiselle taulukolle on tehty oma rivin lisäämiseen tarkoitettu makro. Rivin lisäämiseen liittyvät makrot ovat keskenään hyvin samankaltaisia ja niiden perusperiaatteet ovat aivan samat, mutta taulukoiden kokojen, kaavojen, väritysten, yms. asioiden vaihdellessa samaa funktiota ei ole mahdollista järkevästi tai helposti käyttää.

Seuraavaksi esitellään esimerkkinä rivin lisäämiseen tarkoitetusta makrosta cmdRivinLisaysRaa, joka lisää uuden rivin raaka-aineiden taulukkoon. Kyseinen makro on esitelty kuvassa 76.

```
Sub cmdRivinLisaysRaa()

    ' Puretaan välilehden suojaus
    Call VapautaSivu("Raaka-aineet")
    Call VapautaSivu("Dataa")

    Application.EnableEvents = False

    Call NapinKlikkaus1(ActiveSheet.Shapes("RaaLisaysNappi"))

    ' Lisätään rivi
    Range("RaakaaineetYhteensa").EntireRow.Select
    Selection.Insert

    ' Palataan kiintopisteen kautta viimeisimmälle olemassa olevalle riville.
    ' Täytetään viimeisimmän rivin sisältö mm. kaavat ja väritäytöt uudelle riville
    With Range("RaakaaineetYhteensa").Offset(-1, 0)
        Range(.Offset(-1, 0), .Offset(-1, 6)).AutoFill Destination:=Range(.Offset(-1, 0), _
        .Offset(0, 6)), Type:=xlFillDefault
        ' Poistetaan edellisen rivin kopioimisesta jääneet sisällöt.
        .ClearContents
        .Offset(0, 1).ClearContents
        .Offset(0, 3).ClearContents
    End With

    Dim alarivi As Integer
    Dim ylärivi As Integer
    Dim VSar As Integer
    Dim datasivu As Range

    ylärivi = Range("RaaRaakaaineet").row
    alarivi = Range("RaakaaineetYhteensa").row
    VSar = Range("RaaRaakaaineet").Column

    With Range("RaakaaineetYhteensa")
        .Offset(0, 1).Formula = "=SUM(" & SarakeKirjain(VSar + 1) & ylärivi + 1 & ":" & _
        SarakeKirjain(VSar + 1) & alarivi - 1 & ")"
        .Offset(0, 2).Formula = "=SUM(" & SarakeKirjain(VSar + 2) & ylärivi + 1 & ":" & _
        SarakeKirjain(VSar + 2) & alarivi - 1 & ")"
        .Offset(0, 4).Formula = "=SUM(" & SarakeKirjain(VSar + 4) & ylärivi + 1 & ":" & _
        SarakeKirjain(VSar + 4) & alarivi - 1 & ")"
        .Offset(0, 5).Formula = "=SUM(" & SarakeKirjain(VSar + 5) & ylärivi + 1 & ":" & _
        SarakeKirjain(VSar + 5) & alarivi - 1 & ")"
    End With

    Set datasivu = ThisWorkbook.Sheets("Dataa").Range("Kokonaismassa")
    datasivu.Formula = "='Raaka-aineet'!" & SarakeKirjain(VSar + 1) & alarivi
    datasivu.Offset(0, -1).Formula = "=COUNTA('Raaka-aineet'!" & SarakeKirjain(VSar) & _
    ylärivi + 1 & ":" & SarakeKirjain(VSar) & alarivi - 1 & ")"

    Call PaivitaKaavio("RaaYmpyra", Range("RaaRaakaaineet"), Range("RaakaaineetYhteensa"), _
    Range("RaaOsuusPaastoista"))

    Range("RaakaaineetYhteensa").Select
    Application.EnableEvents = True

    Call NapinKlikkaus2(ActiveSheet.Shapes("RaaLisaysNappi"))

    'Suojataan välilehti
    SuojaaSivu ("Raaka-aineet")
    SuojaaSivu ("Dataa")

End Sub
```

Kuva 76. cmdRivinLisaysRaa-makro.

Makron alussa vapautetaan muokattavat Raaka-aineet- ja Dataa-välilehdet, estetään tapahtumiin reagoivien makrojen toiminta ja aktivoidaan painikkeen efektin makro. Tämän jälkeen suoritetaan heti varsinainen rivin lisääminen valitsemalla taulukon koko alarivi, ja lisäämällä tämän yläpuolelle uusi rivi. Tämän jälkeen aloitetaan rivin täyttäminen ja muotoilu.

Ensin täytetään uudelle riville tiedot ja muotoilut niin sanotulla automaattisen täytön toiminnolla. Tällöin kuitenkin kaikki tiedot ja kaavat eivät välttämättä ole vielä oikein, joten näitä täytyy edelleen muokata. Tärkeimpänä näistä poistetaan edelliselle riville käyttäjän täyttämät tiedot, jotta uusi rivi on tyhjä.

Uuden rivin lisäämisen myötä alimmalla yhteenvetorivillä sijaitsevien summakaavojen alueet eivät enää pidä paikkaansa, joten nämä kaavat tulee määritellä uudelleen. Näiden kaavojen määrittämisessä käytetään VSar-, ylärivi- ja alarivi-nimisiä muuttujia sekä SarakeKirjain-funktiota summakaavan määrittämiseen siten, että kaavaan tulee oikea alue eikä taulukon sijainnilla Excel-taulukossa ole merkitystä.

Tämän jälkeen muokataan kaksi kaavaa Dataa-välilehdelle. Nämä kaavat laskevat kaikkien raaka-aineiden massojen summaa sekä eri raaka-aineiden lukumäärää. Kaavat päivitetään samalla periaatteella samoja muuttujia ja funktioita hyödyntäen, kuten raaka-aineiden taulukkoon lisättävien kaavojen tapauksessa.

Makron lopuksi välilehden ympyräkaavio päivitetään omaa PaivitaKaavio-funktiota hyödyntäen, johon tässä kappaleessa tutustutaan tarkemmin myöhemmin. Tämän lisäksi kutsutaan napin klikkaamisen efektistä vastaavaa makroa ja suojataan välilehdet.

Muut rivin lisäämiseen liittyvät makrot ovat sisällöltään ja yksityiskohdiltaan hieman erilaisia, mutta periaatteeltaan ja toiminnoiltaan vastaavat kuten cmdRivinLisaysRaa-makrossa, joten muita rivin lisäys makroja ei käydä yksityiskohtaisesti läpi. Seuraavaksi esitellään rivin poistamisesta vastaava RivinPoistaja(ListanYla As String, ListanPohja As String) -funktio, jonka koodi on esitelty kuvakaappauksessa kuvassa 77.

```
Private Function RivinPoistaja(ListanYla As String, ListanPohja As String)

    Application.EnableEvents = False
    Dim YNAnswer As Integer
    Dim Maara As Integer
    Maara = Range(ListanPohja).row - Range(ListanYla).row - 1

    'Varmistetaan, että valittu alue on listalla.
    If Selection.Column = Range(ListanYla).Column And _
    Selection.row > Range(ListanYla).row _
    And (Selection.row + Selection.Rows.Count - 1) < Range(ListanPohja).row Then
       GoTo Jatketaan
    Else:
        MsgBox "Valittu alue ei ole listalla. Valitse vain poistettavat kohteet listasta.", _
        vbOKOnly, "Virheilmoitus"
        GoTo Loppu
    End If

Jatketaan:

    ' Varmistetaan, ettei poisteta ainoaa riviä. Jos rivi on viimeinen, ei poistoa suoriteta.
    If Selection.Rows.Count >= Maara Then
        MsgBox "Kaikkia rivejä ei voida poistaa. Voit halutessasi jättää solut tyhjiksi.", _
        vbOKOnly, "Virheilmoitus"
        GoTo Loppu
    End If

    ' Poistetaan rivin suojaus
    VapautaSivu (ActiveSheet.Name)
    ' Poistetaan rivi.
    Selection.EntireRow.Select
    Selection.Delete
    Range(ListanPohja).Select

Loppu:

    Application.EnableEvents = True
    SuojaaSivu (ActiveSheet.Name)

End Function
```

Kuva 77. RivinPoistaja-funktio.

RivinPoistaja-funktio tarvitsee toimiakseen kaksi muuttujaa syötteenä: ListanYla- ja Listan-Pohja-nimiset String-muuttujat. Näiden perusteella funktio määrittelee taulukon sijainnin Excelissä, eikä näin ollen poista vahingossa taulukon ulkopuolisia tietoja, otsikkorivejä tai yhteenvetorivejä.

Ensimmäiseksi funktiossa muuttujien määrittämisen jälkeen varmistetaan, että käyttäjän valitsema alue on taulukossa. Tämä suoritetaan ehtorakenteella, joka vertaa valitun alueen sarakkeiden ja rivien numeroita toisiinsa. Mikäli alue on sopiva, siirrytään seuraavaan rivin poistavaan osioon. Jos taas alue ei ole poistettavissa, siirrytään suoraan funktion loppuun ja annetaan käyttäjälle virheilmoitus.

Rivien poistamisen suorittavassa osioissa varmistetaan aluksi, että ei poisteta ainoaa olemassa olevaa riviä. Tätä ei sallita, koska rivien lisäämiset perustuvat olemassa olevan rivin kopiointiin, eikä kaikkia tarvittavia kaavoja ja muotoiluja ole välttämättä määritetty VBA:ssa. Jos rivin poistamiseen vaadittavat ehdot ovat kunnossa suoritetaan varsinainen rivien poistaminen valitsemalla alueen rivit ja poistamalla ne .Delete-metodia käyttäen.

Rivien lisäämisen ja poistamisen lisäksi Rivien_hallinnointi-moduulista löytyy makro PaivitaKaavio(Kaavio As String, OtsikkoAlue As Range, ListanPohja As Range, ArvoAlue As Range), jonka tarkoituksena on päivittää välilehdillä sijaitsevien ympyräkaavioiden tietoalueita. Kaavioiden tietoalueet tulee päivittää rivien lisäämisen yhteydessä, sillä kaaviot eivät itse osaa automaattisesti muuttaa tietoalueitaan niiden kasvaessa. Kuvassa 78 on nähtävissä kuvakaappaus kyseisestä makrosta.

```
Sub PaivitaKaavio(Kaavio As String, OtsikkoAlue As Range, ListanPohja As Range, ArvoAlue As Range)

    ActiveSheet.ChartObjects(Kaavio).Activate
    Application.CutCopyMode = False

    ActiveChart.FullSeriesCollection(1).XValues = "='" & OtsikkoAlue.Worksheet.Name & _
    "'!" & OtsikkoAlue.Offset(1, 0).Address & ":" & ListanPohja.Offset(-1, 0).Address

    ActiveChart.FullSeriesCollection(1).Values = "='" & OtsikkoAlue.Worksheet.Name & _
    "'!" & ArvoAlue.Offset(1, 0).Address & ":" & ArvoAlue.End(xlDown).Offset(-1, 0).Address

End Sub
```

Kuva 78. PaivitaKaavio-makro

PaivitaKaavio-makro tarvitsee syötteenä kaavion nimen Kaavio As String, sarakkeen, josta haetaan kaavioon otsikot OtsikkoAlue as Range, sarakkeen, josta haetaan kaavioon arvot ArvoAlue as Range sekä kyseisen taulukon viimeisen rivin sijainti ListanPohja As Range. Makro on kohtalaisen yksinkertainen, sillä tietoalueet voidaan päivittää otsikoille helposti komennolla ActiveChart.FullSeriesCollection(1).XValues =… ja vastaavasti arvoille komennolla ActiveChart.FullSeriesCollection(1).Values =… Näiden perään määritellään kaava, joka viittaa haluttuun alueeseen.

4.3.9 Sekalaista

Sekalaista-moduuli sisältää sellaisia makroja, jotka eivät järkevästi sopineet muihin moduuleihin tai joiden takia ei myöskään ollut järkevää tehdä omia moduulejaan. Nämä makrot voivat

olla hyvin spesifeihin käyttötarkoituksiin suunniteltuja, tai niitä voidaan kutsua monista eri moduuleista, Microsoft Excel Objekteista tai käyttöliittymistä.

Moduuli sisältää yksitoista makroa: TalousAllokointi, MassaAllokointi, SarakeKirjain(SarakeNumero As Integer) As String, zoom, AakkostaPK(ListanNimi As String), NapinKlikkaus1(klikki As Shape), NapinKlikkaus2(klikki As Shape), OletusAsetukset, KorjaaPudotusvalikko(strKohde As String, strKohdeSivu As String, strLahde As String), KehittajaAsetukset. Näistä ensimmäiset kaksi liittyvät Yhteenveto-sivulta löytyvään sivuvirtojen allokointimenetelmän valintaan. TalousAllokointi-makron koodi on esitelty kuvassa 79.

```
Sub TalousAllokointi()

    If ThisWorkbook.Sheets("IKE-Laskentamenetelmä").Shapes("SensuuriNappula").Fill.ForeColor.RGB = RGB(251, 173, 24) Then
        Exit Sub
    End If

    Call VapautaSivu("Yhteenveto")

    Application.EnableEvents = False

    Dim rngLista As Range
    Dim VSar As Integer
    Dim alarivi As Integer
    Dim ylarivi As Integer
    Dim i As Integer

    Set rngLista = ThisWorkbook.Sheets("Yhteenveto").Range("SivutuotteetAlku").CurrentRegion
    VSar = Range("SivutuotteetAlku").Column
    ylarivi = Range("SivutuotteetAlku").row
    alarivi = Range("SivutuotteetAlku").row + rngLista.Rows.Count - 1

    For i = 2 To rngLista.Rows.Count
        rngLista(i, 4).Interior.Color = RGB(251, 173, 24)
        Call LisaaMuokkaus(rngLista(i, 4), rngLista(i, 1), "Yhteenveto")
        Call VapautaSivu("Yhteenveto")
        rngLista(i, 4).ClearContents
        rngLista(i, 4).NumberFormat = "0.00%"
    Next i

    rngLista.Borders.LineStyle = xlContinuous

    ActiveSheet.Shapes("TalousAllokointiNappi").Fill.ForeColor.RGB = RGB(251, 173, 24)
    ActiveSheet.Shapes("MassaAllokointiNappi").Fill.ForeColor.RGB = RGB(218, 217, 215)

    Application.EnableEvents = True

    Call SuojaaSivu("Yhteenveto")

End Sub
```

Kuva 79. TalousAllokointi-makro

TalousAllokointi-makron tarkoituksena on muuttaa sivuvirtojen taulukon allokointiperiaatetta siten, että kohdentamiskerroin sarakkeeseen käyttäjä voisi määritellä kertoimen itse. Ensimmäiseksi makro varmistaa, ettei Piilota prosessitiedot-asetus ole kytkettynä päälle. Mikäli näin on, lopetetaan ohjelman toiminta, jottei taulukko mene rikki. Muussa tapauksessa jatketaan eteenpäin.

Asetusten muuttamisen ja muuttujien määrittämisen jälkeen sivuvirtojen taulukkoa käydään rivi kerrallaan läpi For-silmukkaa hyödyntäen. Jokaisella silmukan kierroksella muutetaan kohdennuskertoimen sarakkeen taustaväri oranssiksi, lisätään muokkausalue (Allow Edit Range), puhdistetaan vanhat tiedot ja muutetaan solun formaatti prosenteiksi kahden desimaalin tarkkuudella. Tämän jälkeen enää vaihdetaan Massaperusteinen allokointi- ja Talousperusteinen allokointi -painikkeiden väritykset, palautetaan asetukset ja suojataan välilehti.

MassaAllokointi-makro toimii täysin samalla periaatteella, kuin TalousAllokointi-makro. Oleellisena erona lisäämisen sijaan poistetaan muokkausalue ja lisätään kohdennuskertoimen soluun kaava, joka laskee automaattisesti kohdennuskertoimen sivuvirran ja tuotteen kokonaismassan suhteesta: "=IFERROR(" & SarakeKirjain(VSar + 2) & ylarivi + i - 1 & "/('IKE-Laskentamenetelmä'!TuoMassa),0)".

Seuraava Sekalaista-moduulin funktio on hyvin yksinkertainen, mutta sitä käytetään laajasti lähes kaikkialla ohjelmassa erityisesti kaavojen määrittämisen yhteydessä. Funktio SarakeKirjain(SarakeNumero As Integer) As String ottaa syötteenä kokonaisluvun ja palauttaa tätä kokonaislukua vastaavan kirjaimen. Tämä on hyvin hyödyllinen toiminto, sillä Excelin kaavoihin tulee määritellä sarakkeet kirjaimina, mutta VBA:ssa sarakkeiden sijainnit ovat määritelty numeroina. Funktio sisältää vain yhden rivin, joka on nähtävissä kuvassa 80.

```
Function SarakeKirjain(SarakeNumero As Integer) As String

  SarakeKirjain = Split(Cells(1, SarakeNumero).Address, "$")(1)

End Function
```

Kuva 80. SarakeKirjain-makro.

Sarakekirjain-makro hyödyntää VBA:n Split-funktiota ja solun Address-ominaisuutta. Makro luo Excelistä ensimmäiseltä riviltä solun osoitteen siitä sarakkeesta, joka SarakeNumero-muuttujassa määriteltiin Cells(1, SarakeNumero).Address -komennon avulla. Esimerkiksi SarakeNumeron ollessa 3, tämä komento tuottaa tekstin: ”C1”, joka on siis määritellyn solun ”ankkuroitu” osoite. Tämä termi paloitellaan Split-funktion avulla ”$” merkkien kohdilta ja valitaan ensimmäinen osa, jolloin saadaan saraketta vastaava kirjain ”C”.

Seuraava makro Sekalaista-moduulissa on nimeltään zoom. Tämän makron tarkoitus on säätää työkalun zoomaus siten, että laskurin ylävalikko on juuri ja juuri kokonaan näkyvissä käytetyllä näytöllä. Kuvassa 81 on esitetty kuvakaappaus zoom-makron koodista.

```
Sub zoom()

    Dim zoom As Single

    ActiveWindow.zoom = Application.UsableWidth / 1500 * 0.97 * 100

End Sub
```

Kuva 81. zoom-makro.

Makro on toiminnaltaan hyvin yksinkertainen. Se määrittelee laskurin zoomauksen (ActiveWindow.zoom) käytetyn näytön leveyteen perusteella (Application.UsableWidth). Käytetyn näytön leveys jaetaan 1500:lla, joka on laskurissa määritetty valikon leveydeksi kuvapisteinä. Näin saadaan suhde saatavilla olevan leveyden ja valikon leveyden välillä. Tämä suhde kerrotaan 0,97, jotta zoomatessa valikko olisi kuitenkin mieluummin hieman näyttöä kapeampi eikä jäisi osittain reunan alle. Tämä luku vielä kerrotaan sadalla, jolloin saadaan prosenttiyksikköinä tarvittava zoomaus.

Kuvassa 82 on esitetty kuvakaappaus AakkostaPK(ListanNimi As String)-funktiosta. Tämän funktion tarkoitus on yksinkertaisesti järjestää Päästökertoimet-välilehdellä raaka-aineiden päästökertoimet aakkosjärjestykseen. Excelissä on olemassa omakin Lajittele ja suodata -toiminto, joka osaisi järjestää rivit haluttuun järjestykseen huomattavasti yksinkertaisemmin. Tässä Excelin sisäisessä toiminnossa on kuitenkin yksi hyvin merkittävä heikkous, sillä se käytännössä tuhoaa solujen sisältämät hyperlinkit. Koska raaka-aineiden päästökertoimien listassa on yhdessä sarakkeessa lähdeviitteet linkkeineen, ei Excelin omaa työkalua voida käyttää. Tästä syystä oli siis tarpeellista luoda oma työkalu laskurin kehittäjien tarpeisiin, kun päästökertoimia lisätään paljon kerralla.

AakkostaPK-funktion toiminta perustuu siihen, että solujen kopioiminen ja liittäminen eivät tuhoa hyperlinkkejä. (Eric Bentzen, i.a.) Lyhyesti ideana on, että ensin koko järjestettävä lista kopioidaan eri paikkaan alkuperäisen listan kanssa (tässä tapauksessa listan viereen), lisätään kopioon uusi sarake, johon lisätään juokseva järjestysnumero, jonka

jälkeen kopioitu alue järjestetään VBA:n omalla Sort-funktiolla aakkosjärjestykseen. Tällöin kopiossa hyperlinkit vioittuvat, mutta järjestysnumerot kertovat alkuperäiset rivien paikat. Nyt alkuperäisestä järjestämättömästä listasta voidaan kopioida yksi kerrallaan ehjät hyperlinkit vioittuneiden tilalle kopioon järjestysnumeroiden perusteella. Kun kaikki hyperlinkit on korvattu, voidaan alkuperäinen lista korvata kopiolla, joka on nyt aakkosjärjestyksessä ja josta hyperlinkit ovat vaihdettu. Koska kyseessä ei ole laskurin toiminnan kannalta kaikista tärkeimmästä makrosta, ei koodia käydä rivi riviltä läpi. Funktion toimintaan voi kuitenkin tutustua lisää Eric Bentzenin kirjoituksesta, jonka ajatuksiin tämä makro pohjautuu.

```
Function AakkostaPK(ListanNimi As String)

    Call SuojaaSivu("Päästökertoimet")

    ' Tässä functiossa hyvin vahvoja vaikutteita Eric Bentzenin kirjoituksesta:
    ' Sort hyperlinks in Excel with VBA macros
    ' https://sitestory.dk/excel_vba/sorting-hyperlinks.htm

    Dim rngAlkuperainen As Range
    Dim rngKopio As Range
    Dim rngKSolu As Range
    Dim i As Long

    Application.ScreenUpdating = False

    Set rngAlkuperainen = Range(ListanNimi).CurrentRegion

    rngAlkuperainen.Copy Destination:=Range(ListanNimi).Offset(0, 6)

    Set rngKopio = Range(ListanNimi).Offset(0, 6).CurrentRegion

    For i = 1 To rngKopio.Rows.Count
        Range(ListanNimi).Offset(0, 6).Offset(i - 1, 5) = i
    Next

    Range(ListanNimi).Offset(0, 6).CurrentRegion.Select

    Selection.Sort Key1:=Range(ListanNimi).Offset(0, 6).Offset(0, 1), Order1:=xlAscending, _
    Header:=xlGuess, _
    OrderCustom:=1, MatchCase:=False, _
    Orientation:=xlTopToBottom

    Set rngKopio = Range(Range(ListanNimi).Offset(0, 6), Range(ListanNimi).Offset(0, 6).End(xlDown))

    For Each rngKSolu In rngKopio.Cells
        Range(ListanNimi).Offset(rngKSolu.Offset(0, 5).Value - 1, -1).Copy Destination:=rngKSolu
        With Range(rngKSolu, rngKSolu.Offset(0, 4))
            .Select
            .Font.Name = "Arial"
            .Font.Size = 12
            .Font.Color = RGB(0, 0, 0)
            .Font.Underline = False
            If rngKSolu.Offset(0, 5).Value = 1 Then
                .Font.Bold = True
            Else
                .Font.Bold = False
            End If

        End With
    Next

    rngKopio.Offset(0, 5).Clear

    Range(ListanNimi).Offset(0, 6).CurrentRegion.Cut Destination:=Range(ListanNimi).CurrentRegion


    ActiveWorkbook.Names.Add Name:=ListanNimi, RefersToR1C1:=Range("RaakaaineetPK").Offset(2, 0)
    Range(ListanNimi).CurrentRegion.Borders.LineStyle = xlContinuous
    Set rngKopio = Nothing
    Set rngAlkuperainen = Nothing
    Set rngKSolu = Nothing
    Range(ListanNimi).Select

    Application.ScreenUpdating = True

    Call SuojaaSivu("Päästökertoimet")

End Function
```

Kuva 82. AakkostaPK-makro.

Sekalaista-moduuli sisältää makrot painikkeiden klikkauksien efektejä varten. Näillä muutetaan hetkellisesti halutun painikkeen väriä ja varjostustehostetta, jolloin syntyy illuusio painikkeen painamisesta. Tämä toteutetaan kahdella erillisellä makrolla, joista toista (NapinKlikkaus1(klikki As Shape)) kutsutaan ennen painikkeen painamisesta aktivoitavan toiminnon aloittamista ja toista (NapinKlikkaus2(klikki As Shape)) tämän jälkeen. Kuvassa 83 on nähtävissä kuvakaappaus molemmista makroista.

```
Function NapinKlikkaus1(klikki As Shape)

    Application.ScreenUpdating = True

    klikki.Fill.ForeColor.RGB = RGB(251, 173, 24)
    klikki.Shadow.Type = msoShadow34

End Function
```

```
Function NapinKlikkaus2(klikki As Shape)

    Application.ScreenUpdating = True

    klikki.Fill.ForeColor.RGB = RGB(218, 217, 215)
    klikki.Shadow.Type = msoShadow22

End Function
```

Kuva 83. NapinKlikkaus1 ja NapinKlikkaus2 -funktiot.

Seuraavaksi Sekalaista-moduulissa on OletusAsetukset-makro, jonka tarkoituksena on tehdä tarvittavat muutokset oletusasetusten palauttamiseksi, kun käyttäjä painaa laskurin etusivulla Palauta oletusasetukset -painiketta. Tämä voi tulla tarpeeseen, mikäli käyttäjä ei ole varma siitä, ovatko suositellut asetukset päällä tai tietyt Excelin omat sisäiset aseukset eivät ole palautuneet oikein virhetilanteen yhteydessä. Kuvassa 84 on nähtävissä kuvakaappaus OletusAsetukset -makrosta.

```
Sub OletusAsetukset()

    Application.EnableEvents = True
    Application.ScreenUpdating = True

    If ActiveSheet.Shapes("SensuuriNappula").Fill.ForeColor.RGB = RGB(251, 173, 24) Then
        Call Naytto_Esittely_Click
    End If

    Call zoom

    If Not ActiveSheet.Shapes("VapautusNappula").Fill.ForeColor.RGB = RGB(122, 154, 1) Then
        Call Suojaa_Vapauta_Click
    End If

End Sub
```

Kuva 84. OletusAsetukset-makro.

Makrossa palautetaan Application.EnableEvents- sekä Application.ScreenUpdating-asetukset True-asentoon. Nämä asetukset saattavat jäädä esimerkiksi virhetilanteissa False-asetukselle, mikäli jonkin toisen makron toiminta on keskeytynyt. Esimerkiksi Application.EnableEvents = False -asetuksen ollessa päällä tapahtumiin reagoivat (Microsoft Excel Objects) makrot eivät toimi ollenkaan, joihin kuitenkin monet laskurin toiminnot perustuvat.

Tämän jälkeen tarkistetaan, onko Piilota prosessitiedot -asetus päällä ja palautetaan se Näytä prosessitiedot -asentoon, mikäli näin on. Tämän jälkeen kutsutaan zoom-makroa ja palautetaan laskuri laskentatilaan, mikäli se oli kehittäjätilassa.

Seuraava Sekalaista-moduulin makro on KorjaaPudotusvalikko (strKohde As String, strKohdeSivu As String, strLahde As String). Nimensä mukaisesti makron tarkoitus on korjata pudotusvalikko, mikäli valikon tiedot esimerkiksi muuttuvat. Näin voi tapahtua esimerkiksi silloin, kun pudotusvalikon vaihtoehdot ovat Päästökerroin-välilehden tietokannan mukaisia, ja tietokantaan lisätään uusia päästökertoimia. Kuvakaappaus makrosta on nähtävissä kuvassa 85.

```
Sub KorjaaPudotusvalikko(strKohde As String, strKohdeSivu As String, strLahde As String)

    Dim rngKohde As Range
    Dim rngLahde As Range
    Dim cell As Range
    Dim Sivu As Worksheet

    Set Sivu = ThisWorkbook.Sheets(strKohdeSivu)
    If strKohde = "MuutLaitteetPolttoaineet" Then
        Set rngKohde = Range(Sivu.Range(strKohde).Offset(1, 0), Sivu.Range(strKohde). _
        Offset(0, -1).End(xlDown).Offset(-1, 1))
    ElseIf strKohde = "AjoneuvotPolttoaineet" Then
        Set rngKohde = Range(Sivu.Range(strKohde).Offset(1, 0), Sivu.Range(strKohde). _
        Offset(0, 2).End(xlDown).Offset(-1, -2))
    Else
        Set rngKohde = Range(Sivu.Range(strKohde).Offset(1, 0), Sivu.Range(strKohde). _
        End(xlDown).Offset(-1, 0))
    End If

    Set rngLahde = Range(ThisWorkbook.Sheets("Päästökertoimet").Range(strLahde). _
    Offset(1, 0), ThisWorkbook.Sheets("Päästökertoimet").Range(strLahde).End(xlDown))

    For Each cell In rngKohde.Cells
        cell.Validation.Modify Type:=xlValidateList, AlertStyle:=xlValidAlertStop, _
        Formula1:="='" & ThisWorkbook.Sheets("Päästökertoimet").Name & "'!" & rngLahde.Address
    Next cell

End Sub
```

Kuva 85. KorjaaPudotusvalikko-makro.

Makro ottaa syötteenä kohteen, jonne pudotusvalikko päivitetään (strKohde As String), välilehden nimen, jossa pudotusvalikko sijaitsee (strkohdeSivu As String) ja kohteen, jossa pudotusvalikon lähtötiedot sijaitsevat Päästökertoimet-sivulla (strLahde As String).

Makrossa on aluksi ehtorakenne, jossa testataan, onko pudotusvalikko muiden laitteiden polttoaineita tai ajoneuvojen polttoaineita koskevissa sarakkeissa. Tällöin pudotusvalikoiden alue määritellään hieman eri tavoin, kuin muissa yleisissä tapauksissa. Tämä johtuu siitä, ettei näissä tapauksissa alimmalla rivillä ollut tietoja kyseisessä sarakkeessa, jolloin .End(xlDown)-ominaisuus ei toimi tarkoitetulla tavalla.

Tämän jälkeen määritellään Päästökertoimet-välilehdeltä alue pudotusvalikon tietoja varten. Kun pudotusvalikon alue ja tietoalue ovat määritelty, voidaan lisätä itse pudotusvalikot. Tämä toteutetaan For Each -silmukkarakenteella, jossa käydään halutun kohde alueen jokainen solu läpi, ja lisätään sinne halutusta sijainnista määritelmä uudesta listasta, jonka perusteella pudotusvalikko muodostetaan.

Viimeisenä makrona Sekalaista-moduulissa on KehittajaAsetukset, jonka tarkoituksena muuttaa Excelistä tietyt ulkoasuun liittyvät asetukset kehittäjätilaan ja laskentatilaan eri näköisiksi. Tämän ideana on tehdä laskentatilasta siistimpi ja yksinkertaisempi, ja kehittäjätilassa palauttaa niin sanotut normaalit ulkoasuasetukset. Kuvassa 86 on nähtävissä kuvakaappaus KehittajaAsetukset-makrosta.

```
Sub KehittajaAsetukset()

    If ThisWorkbook.Sheets("IKE-Laskentamenetelmä").Shapes("VapautusNappula"). _
    Fill.ForeColor.RGB = RGB(255, 0, 0) Then
        ActiveWindow.DisplayHeadings = True
        ActiveWindow.DisplayGridlines = True
        ActiveWindow.DisplayWorkbookTabs = True
        ThisWorkbook.Sheets("Päästökertoimet").Shapes("Suorakulmio 1").Visible = True
    Else
        ActiveWindow.DisplayHeadings = False
        ActiveWindow.DisplayGridlines = False
        ActiveWindow.DisplayWorkbookTabs = False
        ThisWorkbook.Sheets("Päästökertoimet").Shapes("Suorakulmio 1").Visible = False
    End If

End Sub
```

Kuva 86. KehittajaAsetukset-makro.

Makro yksinkertaisesti testaa ensin ehtorakenteen avulla, onko laskuri laskenta- vai kehittäjätilassa. Mikäli kehittäjätila on aktiivinen, makro asettaa näkyville rivien ja sarakkeiden otsikot, taustaruudukon, välilehtivalikon sekä Aakkosta raaka-aineet -painikkeen Päästökertoimet -välilehdeltä. Jos taas laskentatila on aktiivinen, vastaavasti kyseiset asiat asetetaan näkymättömiksi tai pois päältä.

4.3.10 Suojaus

Suojaus-moduuli sisältää makroja, joilla ohjaillaan työkirjan, välilehtien ja yksittäisten alueiden suojauksia ja muokkausoikeuksia. Moduuli sisältää seitsemän makroa, jotka ovat: Suojaa_Vapauta_Click, SuojaaKaikki, VapautaKaikki, SuojaaSivu(Valilehti As String), VapautaSivu(Valilehti As String), PoistaMuokkaus(Alue As Range, Valilehti As String), LisaaMuokkaus(Alue As Range, OtsikkoAlue As Range, Valilehti As String). Nämä makrot ovat monimutkaisuudeltaan hyvin eri tasoisia ja osa niistä kutsuu toisia saman moduulin makroja, joten aloitetaan näiden avaaminen yksinkertaisemmista ja edetään monimutkaisempiin.

Ensimmäisenä käsitellään SuojaaSivu(Valilehti As String) ja VapautaSivu(Valilehti As String) -makrot. Näiden makrojen tehtävä on suojata tai vapauttaa halutun yksittäisen välilehden suojaukset antamalla välilehden nimi funktioille syötteenä. Kuvassa 87 on nähtävissä kuvakaappaus näistä kahdesta funktiosta.

```
Function SuojaaSivu(Valilehti As String)

    If ThisWorkbook.Sheets("IKE-Laskentamenetelmä").Shapes("VapautusNappula"). _
    Fill.ForeColor.RGB = RGB(255, 0, 0) Then
        ThisWorkbook.Sheets(Valilehti).Unprotect Password:="IKE"
    Else
        ThisWorkbook.Sheets(Valilehti).Protect Password:="IKE"
    End If

End Function
Function VapautaSivu(Valilehti As String)

    ThisWorkbook.Sheets(Valilehti).Unprotect Password:="IKE"

End Function
```

Kuva 87. SuojaaSivu ja VapautaSivu -funktiot.

Makrot suorittavat suojaamisen tai vapauttamisen hyvin suoraviivaisesti yhden rivin mittaisella komennolla, mutta SuojaaSivu-funktion tapauksessa täytyy ensin selvittää, onko kehittäjätila päällä. Jos kehittäjätila on päällä, välilehteä ei haluta suojata. Suojauksissa käytetään yksinkertaista salasanaa ”IKE”, jotta suojauksia ei vahingossa saa purettua liian helposti.

Yksittäisten välilehtien lisäksi välillä saatetaan haluta suojata tai vapauttaa kaikki työkirjan suojaukset. Tällöin voidaan hyödyntää SuojaaKaikki- ja VapautaKaikki-makroja. Ne ovat toiminnaltaan hyvin yksinkertaisia ja niiden koodi on nähtävissä kuvasta 88.

```
Sub SuojaaKaikki()

    ThisWorkbook.Sheets("IKE-Laskentamenetelmä").Protect Password:="IKE"
    ThisWorkbook.Sheets("Työkalut").Protect Password:="IKE"
    ThisWorkbook.Sheets("Raaka-aineet").Protect Password:="IKE"
    ThisWorkbook.Sheets("Pakkausmateriaalit").Protect Password:="IKE"
    ThisWorkbook.Sheets("Jätteet ja sivuvirrat").Protect Password:="IKE"
    ThisWorkbook.Sheets("Energia").Protect Password:="IKE"
    ThisWorkbook.Sheets("Logistiikka").Protect Password:="IKE"
    ThisWorkbook.Sheets("Yhteenveto").Protect Password:="IKE"
    ThisWorkbook.Sheets("Tasetarkastelu").Protect Password:="IKE"
    ThisWorkbook.Sheets("Päästökertoimet").Protect Password:="IKE"
    ThisWorkbook.Sheets("Dataa").Protect Password:="IKE"

    ThisWorkbook.Sheets("IKE-Laskentamenetelmä").Activate

End Sub
```

```
Sub VapautaKaikki()

    On Error Resume Next

    Application.ScreenUpdating = False

    On Error GoTo 0

    ThisWorkbook.Sheets("Raaka-aineet").Unprotect Password:="IKE"
    ThisWorkbook.Sheets("Työkalut").Unprotect Password:="IKE"
    ThisWorkbook.Sheets("Pakkausmateriaalit").Unprotect Password:="IKE"
    ThisWorkbook.Sheets("Jätteet ja sivuvirrat").Unprotect Password:="IKE"
    ThisWorkbook.Sheets("Energia").Unprotect Password:="IKE"
    ThisWorkbook.Sheets("Logistiikka").Unprotect Password:="IKE"
    ThisWorkbook.Sheets("Yhteenveto").Unprotect Password:="IKE"
    ThisWorkbook.Sheets("Tasetarkastelu").Unprotect Password:="IKE"
    ThisWorkbook.Sheets("Päästökertoimet").Unprotect Password:="IKE"
    ThisWorkbook.Sheets("Dataa").Unprotect Password:="IKE"

    ThisWorkbook.Sheets("IKE-Laskentamenetelmä").Activate

    ThisWorkbook.Sheets("IKE-Laskentamenetelmä").Unprotect Password:="IKE"

    On Error Resume Next

    Application.ScreenUpdating = True

    On Error GoTo 0

End Sub
```

Kuva 88. SuojaaKaikki- ja VapautaKaikki-makrot.

Seuraava makro Suojaa_Vapauta_Click hyödyntää edellisiä SuojaaKaikki- ja Vapauta-Kaikki -makroja. Sen tarkoitus on tehdä tarvittavat muutokset, kun Kehittäjätila / Laskenta-tila kytkimen asentoa vaihdetaan. Makron koodi on nähtävissä kuvasta 89.

```
Sub Suojaa_Vapauta_Click()

    Dim vastaus As Integer

    Call VapautaKaikki
    If ThisWorkbook.Sheets("IKE-Laskentamenetelmä").Shapes("VapautusNappula"). _
    Fill.ForeColor.RGB = RGB(122, 154, 1) Then

        vastaus = MsgBox("Olet siirtymässä kehittäjätilaan. Kehittäjätila on " & _
        "tarkoitettu nimensä mukaisesti laskurin kehittäjien käyttöön, eikä sitä " & _
        "suositella käytettäväksi laskentaa tehdessä. Kehittäjätilan ollessa " & _
        "käytössä riski laskurin rikkoutumisesta kasvaa." & Chr(13) & Chr(13) & _
        "Oletko varma, että haluat jatkaa kehittäjätilassa?", vbYesNo, _
        "Siirtyminen kehittäjätilaan")

        If vastaus = vbYes Then
            ThisWorkbook.Sheets("IKE-Laskentamenetelmä").Shapes("VapautusNappula"). _
            Fill.ForeColor.RGB = RGB(255, 0, 0)
            ThisWorkbook.Sheets("IKE-Laskentamenetelmä").Shapes("VapautusNappula"). _
            TextFrame.Characters.Text = "Kehittäjätila"
            Application.DisplayFormulaBar = True
            Application.DisplayStatusBar = True
            Call VapautaKaikki
        Else
            Exit Sub
        End If

    ElseIf ThisWorkbook.Sheets("IKE-Laskentamenetelmä").Shapes("VapautusNappula"). _
    Fill.ForeColor.RGB = RGB(255, 0, 0) Then
        ThisWorkbook.Sheets("IKE-Laskentamenetelmä").Shapes("VapautusNappula"). _
        Fill.ForeColor.RGB = RGB(122, 154, 1)
        ThisWorkbook.Sheets("IKE-Laskentamenetelmä").Shapes("VapautusNappula"). _
        TextFrame.Characters.Text = "Laskentatila"
        Application.DisplayFormulaBar = False
        Application.DisplayStatusBar = False
        Call SuojaaKaikki
    End If

    If Application.CommandBars("Ribbon").Height < 100 And ActiveSheet. _
    Shapes("VapautusNappula").Fill.ForeColor.RGB = RGB(255, 0, 0) Then
        CommandBars.ExecuteMso "MinimizeRibbon"
    ElseIf Application.CommandBars("Ribbon").Height > 100 And ActiveSheet. _
    Shapes("VapautusNappula").Fill.ForeColor.RGB = RGB(122, 154, 1) Then
        CommandBars.ExecuteMso "MinimizeRibbon"
    End If

    Call KehittajaAsetukset

End Sub
```

Kuva 89. Suojaa_Vapauta_Click-makro.

Makron aluksi vapautetaan kaikkien välilehtien suojaus VapautaKaikki-makron avulla, jonka jälkeen tutkitaan ehtorakenteella, onko laskuri kehittäjätilassa vai laskentatilassa. Käsitellään ensin vaihtoehto, jossa laskuri oli ennen makron aktivoimista laskentatilassa ja ollaan siirtymässä kehittäjätilaan.

Kehittäjätilaan siirryttäessä käyttäjältä varmistetaan, haluaako hän todella siirtyä poislaskentatilasta. Tällä tavoin pyritään varmistamaan, ettei käyttäjä vahingossa poista laskurin suojauksia ja riko sitä. Mikäli käyttäjä on edelleen sitä mieltä, että haluaa siirtyä kehittäjätilaan, muutetaan painikkeen väri ja teksti, tuodaan kaavarivi sekä Excelin alapalkki näkyviin ja vapautetaan kaikki suojaukset.

Jos taas kehittäjätila oli päällä ja ollaan siirtymässä laskentatilaan, ei käyttäjältä kysytä varmistusviestillä mitään, vaan tehdään muutokset suoraan. Tässä tapauksessa tehdään samat toiminnot vain toisin päin, eli vaihdetaan eri värit ja tekstit painikkeeseen, piilotetaan kaavarivi ja Excelin alapalkki ja suojataan kaikki välilehdet. Makron lopussa vielä kutistetaan työkalurivi, mikäli tarpeellista. Työkalurivin kutistuksessa tai laajentamisessa tulee laskurin tilan lisäksi tutkia myös työkalurivin tila siltä varalta, jos käyttäjä on itse muuttanut sitä. Komento CommandBars.ExecuteMso "MinimizeRibbon" toimii niin sanotusti Toggle kytkimenä, eli samalla komennolla asetus kytketään päälle tai pois.

Seuraavat kaksi makroa: PoistaMuokkaus(Alue As Range, Valilehti As String) ja LisaaMuokkaus(Alue As Range, OtsikkoAlue As Range, Valilehti As String). Poistavat tai lisäävät muokkausalueita laskuriin. Näitä niin sanottuja Allow Edit Rangeja tarvitaan siihen, että käyttäjä voi muokata tarpeellisia soluja silloin kun suojaukset ovat päällä laskentatilassa. Näitä ei ole tässä laskurissa mahdollista määritellä kiinteäksi, koska taulukoiden koot muuttuvat rivien lisäämisen ja poistamisen yhteydessä ja tiettyjen toimintojen ollessa päällä (esim. Talousperusteinen allokointi) muokkausalueita halutaan muuttaa. Näin siis varmistetaan, että käyttäjän on aina laskentatilan ollessa päällä mahdollista muokata vain ja ainoastaan tarpeellisia soluja.

Funktio PoistaMuokkaus(Alue As Range, Valilehti As String) ottaa syötteenä alueen, josta muokkausoikeudet halutaan poistaa (Alue As Range) ja välilehden nimen, jossa alue

sijaitsee (Valilehti As String). Jälkiviisaana voidaan huomauttaa, että välilehden nimi olisi voitu selvittää alueen perusteella, jolloin funktiolle olisi riittänyt vain yksi syöte. Funktion koodi on nähtävissä kuvassa 90.

```
Function PoistaMuokkaus(Alue As Range, Valilehti As String)

    Dim sh As Worksheet, rng As Range, aer As AllowEditRange
    Set sh = ThisWorkbook.Sheets(Valilehti)
    VapautaSivu (Valilehti)

    On Error Resume Next

    For Each aer In sh.Protection.AllowEditRanges
        If Not Application.Intersect(Alue, aer.Range) Is Nothing Then
            aer.Delete
            Exit For
        End If
    Next

    On Error GoTo 0

    SuojaaSivu (Valilehti)
End Function
```

Kuva 90. PoistaMuokkaus-makro.

Makron alussa määritellään tarvittavat muuttujat ja vapautetaan käsiteltävä välilehti. Tämän jälkeen määritellään virheen käsittelyä varten On Error Resume Next, jotta funktio ei kaadu tarpeettomasti. Tämän jälkeen käydään For Each -silmukkarakenteen avulla läpi kaikki kyseisen välilehden sisältämät erilliset muokkausalueet. Silmukka toimii sillä periaatteella, että se tarkistaa Application.Intersect -komennon avulla, leikkaavatko syötteenä saatu alue ja silmukan kierroksella käsiteltävä muokkausalue toisensa. Jos näin tapahtuu, muokkausalue poistetaan ja silmukka katkaistaan. Lopuksi vielä palautetaan virheenkäsittely On Error GoTo 0-tilaan ja suojataan välilehti.

Funktio LisaaMuokkaus vastaavasti lisää halutulle välilehdelle haluttuun kohtaan muokkausalueen. Funktio ottaa kolme muuttujaa syötteenä, joista kaksi ovat samoja, kuten PoistaMuokkaus-funktion tapauksessa. Näiden lisäksi tarvitaan OtsikkoAlue-niminen Range-tyyppinen muuttuja, jota käytetään alueen nimeämiseen. LisaaMuokkaus-funktion koodi on nähtävissä kuvasta 91.

```
Function LisaaMuokkaus(Alue As Range, OtsikkoAlue As Range, Valilehti As String)

    Dim aer As AllowEditRange
    Dim aerTitle As String
    Dim i As Integer
    Dim sh As Worksheet

    Call PoistaMuokkaus(Alue, Valilehti)

    VapautaSivu (Valilehti)

    i = 0

UudestaanUudestaan:

    If i > 100 Then
        MsgBox "Virhe ohjelman toiminnassa.", vbOKOnly, "Virhe"
        Exit Function
    End If
    i = i + 1
    aerTitle = "rng" & Replace(Replace(OtsikkoAlue.Value, " ", ""), ".", "") & i

    On Error GoTo UudestaanUudestaan

    Set aer = Application.ActiveSheet.Protection.AllowEditRanges.Add(title:=aerTitle, Range:=Alue)
    On Error GoTo 0

    SuojaaSivu (Valilehti)

End Function
```

Kuva 91. LisaaMuokkaus-funktio.

Funktio aloitetaan tavalliseen tapaan tarvittavien muuttujien määrittelyllä, jonka jälkeen hyödynnetään PoistaMuokkaus-funktiota. Tämä tehdään siksi, että voidaan olla varmoja, ettei Exceliin synny päällekkäisiä muokkausalueita. Tämän jälkeen välilehti vapautetaan ja annetaan muuttujalle i arvo 1. Tässä vaiheessa myös määritellään koodiin kohta UudestaanUudestaan, minne ohjelma myöhemmin tarvittaessa palaa.

Seuraavaksi on suoritettu ehtorakenteen avulla varmistus, että ohjelma ei joudu päättymättömään silmukkaan. Ehtorakenne lopettaa ohjelman toiminnan, mikäli i saavuttaa arvon 100. Tämän jälkeen i:n arvoa kasvatetaan yhdellä ja nimetään alueen otsikko perustuen syötteenä saadun otsikkoalueen arvoon sekä i:n arvoon. Nimestä karsitaan Replace-funktion avulla mahdolliset välilyönnit ja pisteet pois.

Tämän jälkeen määritellään virheenkäsittely On Error GoTo UudestaanUudestaan, jolloin ohjelman suoritus palaa takaisin muutaman rivin taaksepäin. Tämä aiheuttaa siis toisin sanoen hyvin paljon silmukkarakenteita muistuttavan efektin, joka vain perustuu Excelin antamiin virheilmoituksiin. Täysin saman asian olisi kyllä voinut tehdä vähemmällä kikkailulla

silmukka- ja ehtorakenteita hyödyntäen. Joka tapauksessa seuraavaksi ohjelma yrittää määritellä uuden muokkausalueen. Mikäli alueen nimi on kuitenkin jo käytössä, ohjelma normaalisti kaatuisi virheeseen. Nyt kuitenkin ohjelma palaa virheen sattuessa funktion puoleen väliin, ja yrittää alueen muodostamista uudelleen eri nimellä.

4.3.11 Tasetarkastelu

Tasetarkastelu-välilehdellä olevan kaavion päivittäminen tapahtuu välilehden Worksheet_Activate-makrossa, ja Tasetarkastelu-moduuli sisältää sivulla olevien painikkeiden toimintoihin liittyviä asioita. Sivulta löytyvät Raaka-aine-, Prosessointi- ja Valmis tuote -painikkeiden toiminnot on koodattu tähän moduuliin, sekä nuolien piilottamisesta huolehtiva makro. Nämä makrot eivät ole kuitenkaan kovinkaan oleellisia laskurin toiminnan kannalta, joten niihin ei syvennytä tässä oppaassa tarkemmin.

4.3.12 Tyhjennys

Tyhjennys-moduuli sisältää välilehtien ja koko työkirjan käyttäjän täyttämien tietojen poistamiseen tarkoitettuja makroja. Jokaisen välilehden (joita käyttäjä pystyy laskentatilassa muokkaamaan) tyhjentämiseen on koodattu oma makro, ja kaikkien tietojen tyhjentämiseen tarkoitettu makro sitten hyödyntää toiminnassaan näitä makroja tyhjentäessään tiedot välilehti kerrallaan.

Välilehtien tyhjentämiseen tarkoitetut makrot ovat hyvin samanlaisia toisiinsa nähden, joten tässä oppaassa ei käydä näitä kaikkia läpi. Välilehden tyhjentämiseen tarvitaan käytännössä kahta makroa, joista ensimmäinen reagoi käyttäjän suorittamaan painikkeen klikkaamiseen ja toinen suorittaa varsinaisen tietojen tyhjentämisen.

Kuvassa 92 on nähtävissä kuvakaappaus TyhjennaRaa_Click-makrosta, joka aktivoituu käyttäjän painaessa Raaka-aineet-välilehdellä painiketta Tyhjennä sivun tiedot. Makrossa varmistetaan MsgBox-funktion avulla käyttäjältä, halutaanko sivun tiedot varmasti tyhjentää, jonka jälkeen käyttäjän vastauksen perusteella joko keskeytetään makron toiminta tai kutsutaan varsinaista tyhjennyksen suorittavaa makroa TyhjennaRaa. Syy siihen, ettei käyttäjältä

kysytä varmistusta vain suoraan TyhjennaRaa-makrossa piilee siinä, että kun tyhjennetään koko työkirjan kaikki tiedot, ei haluta ohjelman kysyvän varmistusta erikseen jokaisen välilehden tapauksessa.

```
Sub TyhjennaRaa_Click()

    Dim YNAnswer As Integer

    'Varmistetaan käyttäjältä, että haluaako hän varmasti tiedot rivin pysyvästi.
    YNAnswer = MsgBox("Kaikki tälle sivulle täytetyt tiedot poistetaan pysyvästi. Oletko aivan varma?", _
    vbYesNo, "Varmistus viesti")

    If YNAnswer = vbNo Then
        Exit Sub
    End If

    Call TyhjennaRaa

End Sub
```

Kuva 92. TyhjennaRaa_Click-makro.

Varsinainen Raaka-aine-välilehden tyhjennys tapahtuu kuvassa 93 esitellyssä TyhjennaRaa-makrossa. Normaalien muuttujien määritysten, sivun vapautuksen, napin klikkausefektien, ja asetusten muuttamisen jälkeen aloitetaan tyhjentäminen poistamalla rivejä. Rivit poistetaan hyödyntämällä cmdRivinPoistoRaa-funktiota ehtorakenteen sisällä, joka testaa onko poistettavia rivejä ylipäätään olemassa. Mikäli rivejä raaka-aineiden taulukosta löytyy, valitaan poistettava alue ja suoritetaan poistaminen. Tämän jälkeen tyhjennetään viimeisen rivin tiedot, koska cmdRivinPoistoRaa ei poista kaikkia rivejä. Lopuksi vielä palautetaan asetukset, tehdään klikkausefektit ja suojataan sivu.

```
Sub TyhjennaRaa()

    Dim shRaa As Worksheet
    Set shRaa = ThisWorkbook.Sheets("Raaka-aineet")

    shRaa.Activate

    Call VapautaSivu("Raaka-aineet")

    Call NapinKlikkaus1(ThisWorkbook.Sheets("Raaka-aineet").Shapes("TyhjRaaNappi"))

    Application.ScreenUpdating = False
    Application.EnableEvents = False

    If shRaa.Range("RaaRaakaaineet").CurrentRegion.Rows.Count > 3 Then
        shRaa.Range(shRaa.Range("RaaRaakaaineet").Offset(2, 0), _
        shRaa.Range("RaakaaineetYhteensa").Offset(-1, 0)).Select
        Call cmdRivinPoistoRaa
    End If

    With shRaa.Range("RaaRaakaaineet")
        .Offset(1, 0).ClearContents
        .Offset(1, 1).ClearContents
        .Offset(1, 3).ClearContents
    End With

    Application.ScreenUpdating = True
    Application.EnableEvents = True

    Call NapinKlikkaus2(ActiveSheet.Shapes("TyhjRaaNappi"))
    Call SuojaaSivu("Raaka-aineet")

End Sub
```

Kuva 93. TyhjennaRaa-makro.

Muiden välilehtien tyhjentämiseen käytetyt makrot toimivat täysin samalla periaatteella, kuten Raaka-aineet-välilehdelläkin. Eroja on vain niiden välilehtien tapauksessa, joissa on enemmän kuin yksi taulukko. Tällöin tiedot kuitenkin poistetaan täysin samalla tavalla, mutta erikseen jokaisesta taulukosta.

Kuvassa 94 on esitetty kuvakaappaus TyhjennaTiedot-makrosta, joka tyhjentää kaikki käyttäjän laskuriin syöttämät tiedot. Makro hyödyntää välilehtien tyhjentämiseen suunniteltuja makroja, joiden avulla välilehdet tyhjennetään yksi kerrallaan.

```
Sub TyhjennaTiedot()

    Dim iWindowState As Integer
    Dim YNAnswer As Integer

    Call VapautaKaikki

    Call NapinKlikkaus1(ActiveSheet.Shapes("TyhjKaikkiNappi"))

    'Varmistetaan käyttäjältä, että haluaako hän varmasti tiedot rivin pysyvästi.
    YNAnswer = MsgBox("Kaikki tähän tiedostoon täytetyt tiedot poistetaan pysyvästi." _
    & " Oletko aivan varma?", vbYesNo, "Varmistusviesti")

    If YNAnswer = vbNo Then
        Call NapinKlikkaus2(ActiveSheet.Shapes("TyhjKaikkiNappi"))
        Exit Sub
    End If

    With Application
        iWindowState = .WindowState
        .WindowState = xlMinimized
        .ScreenUpdating = True
        .DisplayAlerts = False
    End With

    Application.EnableEvents = False
    ThisWorkbook.Sheets("Energia").Activate
    ThisWorkbook.Sheets("Logistiikka").Activate
    ThisWorkbook.Sheets("Tasetarkastelu").Activate
    Application.EnableEvents = True

    Call TyhjennaEtus 'Etusivu
    Call TyhjennaRaa 'Raaka-aineet
    Call TyhjennaPak 'Pakkausmateriaalit
    Call TyhjennaJat 'Jätteet ja sivuvirrat
    Call TyhjennaEner 'Energia
    Call TyhjennaLog 'Logistiikka

    ThisWorkbook.Sheets("Dataa").Range("Muistiinpanot").CurrentRegion.Columns(2).ClearContents

    ThisWorkbook.Sheets("Tasetarkastelu").Activate
    ThisWorkbook.Sheets("IKE-Laskentamenetelmä").Activate

    With Application
        .ScreenUpdating = True
        .DisplayAlerts = True
        .WindowState = iWindowState
    End With

    Call NapinKlikkaus2(ActiveSheet.Shapes("TyhjKaikkiNappi"))

    Call SuojaaKaikki

End Sub
```

Kuva 94. TyhjennaTiedot-makro.

Makron alussa kysytään käyttäjältä varmistusviesti tietojen tyhjentämisestä samaan tapaan, kuten yksittäisten välilehtien tapauksessa. Tämän jälkeen muutetaan asetukset, jonka yhteydessä laskurin ikkuna kutistetaan, jotta käyttäjä ei näe välilehtien vilkkumista. Varsinainen tietojen tyhjennys välilehdiltä suoritetaan kutsumalla niihin tarkoitettuja makroja, jonka jälkeen tyhjennetään vielä Muistiinpanot-työkalun sisältö poistamalla tiedot Dataa-välilehdeltä.

Lopuksi vielä aktivoidaan tasetarkastelukaavion korjaamiseksi ja palautetaan fokus etusivulle, palautetaan asetukset, suoritetaan klikkausefekti ja suojataan kaikki sivut.

5 KEHITYSEHDOTUKSET

Tässä luvussa esitellään mahdollisia tulevaisuuden kehityskohteita IKE-hiilijalanjälkilaskurille. Ensimmäiseksi käsitellään todennäköisintä päivittämisen tarvetta eli laskurin sisältämää päästökerrointietokantaa. Tämän jälkeen esitellään laskurissa olemassa olevien toimintojen kehityskohteita, mikä niissä on pielessä ja miten niitä voisi mahdollisesti parantaa. Lopuksi pohditaan mahdollisia kokonaan uusia ominaisuuksia, jotka voisivat tuoda laskurille lisäarvoa laadukkaamman laskennan tai parannetun käyttäjäystävällisyyden muodossa.

5.1 Tietokantojen päivittäminen

Päästökerroin-välilehdellä on kaikki laskurin käyttämät päästökertoimet lajiteltuna sektoreittain, jonka sisällä kertoimet on lajiteltu aakkosjärjestykseen päästökertoimen mukaan. Muistiinpanot-, Lisää päästökerroin- ja Hae päästökerrointa-painikkeet löytyvät välilehden yläreunasta. Raaka-aineiden yhteydestä löytyy Aakkosta päästökertoimet -painike, josta enemmän kappaleessa 4.3.9 Sekalaista.

Tietokannat eivät ole kaikenkattavia, sillä etenkin pakkausmateriaaleja ja raaka-aineita on satoja, ellei tuhansia. Hiilijalanjälkilaskentaa tehdään vuosi vuodelta enemmän eli uusia ja tarkempia päästökertoimia tulee koko ajan lisää. Myös kotimaisista tuotteista on paremmin tietoa saatavilla. EU:n lainsäädäntö kehittyy jatkuvasti siihen suuntaan, että ympäristövaikutusten arviointi tulee osaksi jokaisen yrityksen toimintaa, jolloin laskentatietokantaan on hyvä kerätä omassa tuotantoprosessissa tarvittavia päästökertoimia. Lisättävät päästökertoimet kannattaa muuntaa suoraan yksikköön kg CO_2-ekv./kg Työkalut-välilehdeltä löytyvää muunninta apuna käyttäen, jotta laskuri toimisi oikein. Pakkausmateriaaleista puuttuu muun muassa uudelleen käytettäviä materiaaleja kuten meijeriteollisuuden käyttämät maitokorit ja tarjottimet sekä panimoteollisuuden dollyt ja juomapaletit sekä kertakäyttöisiä annospakkauksia, esimerkiksi jogurttipikareita pahvivyötteellä ja ilman sekä pahvikannelliset foliovuoat.

Kiilto Oy on tekemässä hiilijalanjälkilaskentaa ja näillä näkymin työryhmäkohtaisia tuloksia olisi tulossa syksyllä 2023 (T. Peltonen, henkilökohtainen tiedonanto, 9.3.2023). Tilastokeskuksen polttoaineluokitus julkaistaan vuosittain, joten polttoaineiden päästökertoimet olisi

hyvä tarkistaa säännöllisesti. Luonnonvarakeskuksella (LUKE) (i.a.) on Ilmastovaikutusdatasetti ruokapalvelusektorille -hanke, jossa on tarkoituksena tuottaa tuotekohtainen hiilijalanjälkiaineisto ruokapalvelu- ja ravintolatoimialalle, eli sieltä voi saada hyödyllistä tietoa laskuriin.

Raaka-aineita tietokannassa oli tämän oppaan kirjoittamisen aikaan 1160, joten olisi kätevää, jos päästökertoimia voisi lajitella muutenkin kuin aakkosjärjestykseen. Käyttäjän kannalta toimivia ratkaisuja voisi olla lajittelu tuoreryhmän mukaan, esimerkiksi hedelmät ja vihannekset, maitotuotteet, lisäaineet jne. Mielenkiintoista olisi tutkailla kertoimia myös suuruusjärjestyksessä ja lähteen mukaan.

Tämänhetkinen hakutyökalu hakee päästökertoimia tekstin alusta eikä esimerkiksi löydä lisäaineita e-koodin perusteella, koska tietokannassa on ensin aineen nimi ja sitten vasta e-koodi. Näin ollen myös esim. jauhe- tai viinimarja -sanat eivät toimi hakusanoina.

5.2 Kehitettävät asiat laskentamenetelmässä

IKE-hiilijalanjälkilaskurissa on paljon makroja, käyttöliittymiä ja muita ominaisuuksia, jotka on mahdollista tehdä paljon tehokkaammin ja yksinkertaisemmin. Laskurin VBA-ohjelman rakennetta ei suunniteltu etukäteen kovinkaan tarkasti, vaan laskuria kehitettiin sitä mukaan, kun uusia ideoita toiminnallisuuksista keksittiin ja koodaajan taidot kehittyivät uusien toiminnallisuuksien mahdollistamiseksi. Laskuri sisältää näitä tehottomia, tarpeettoman monimutkaisia ja puutteellisia makroja, koska ohjelmaa on pikkuhiljaa rakennettu näiden makrojen varaan, eikä näitä ole nähty siten järkeväksi muokata silloisten aikataulujen puitteissa.

5.2.1 Päästökertoimien hakutyökalu

Kuten aiemmassa kappaleessa 5.1, mainittiin, päästökertoimien hakutyökalulla on tällä hetkellä mahdollista hakea päästökertoimia ainoastaan valitusta sektorista aakkosjärjestyksessä. Tämä johtuu siitä, että hakutyökalun käyttöliittymän toiminta perustuu nykyisellä toteutuksella siihen, että kategorian valinnan jälkeen päästökertoimet lajitellaan siinä järjestyksessä toiseen pudotusvalikkoon, jossa ne ovat päästökerrointietokannassa. Käyttöliittymän

pudotusvalikosta ei ole myöskään mahdollista hakea päästökerrointa muutoin kuin sanan alusta.

Näiden monipuolisempien hakutoimintojen lisääminen on teknisesti mahdollista, mutta voivat olla jokseenkin monimutkaisia ja haastavia koodattavia. Erityisesti raaka-aineiden päästökertoimien tutkimiseksi monipuolisemmat hakutoiminnot olisivat hyödyllisiä. Seuraavaksi esitellään ideoita, millä tavoin nämä muutokset voisivat onnistua.

Kaikista helpoin muutos olisi lisätä raaka-aineiden listaan tuotekategoria-sarake, jonka perusteella päästökertoimia voitaisiin jaotella. Toimiakseen tätä varten tulisi lisätä yksi uusi pudotusvalikko hakutyökaluun, josta tuotekategoria valittaisiin. Tätä toimintoa varten tulisi siis lisätä tietokantaan raaka-aineille uusi sarake, tehdä tarvittavat muokkaukset raaka-aineiden taulukkoa hyödyntäviin makroihin, muokata hakutyökaluun uusi pudotusvalikko ja luoda uusi makro uuden pudotusvalikon tietojen päivittämiseen.

Päästökertoimien järjestyksen muuttaminen esimerkiksi päästökertoimien suuruusjärjestykseen tai lähteiden perusteella onkin sitten astetta haastavampi muutos, mutta kuitenkin mahdollinen. Koska pudotusvalikon tiedot ovat kaikista helpointa laittaa taulukosta silmukkarakenteen avulla suoraan listaan, helpoin tapa saada tiedot pudotusvalikkoon halutussa järjestyksessä on järjestää lista itsessään uudelleen. Tätä ei kuitenkaan haluta välttämättä tehdä, sillä Excelin oman lajittelutoiminnon käyttäminen rikkoo lähteistä hyperlinkit ja oman AakkostaPk-funktion käyttäminen on todella hidasta, vaikka olisikin helppoa muokata se järjestämään rivit muidenkin periaatteiden mukaisesti. Kuitenkin hakutyökalussa hyperlinkit ovat tarpeettomia, joten tämä olisi luultavasti helpoin toteuttaa siten, että tehtäisiin erilliselle piilotetulle välilehdelle tai jonnekin kopio ”oikeasta” raaka-aineiden päästökertoimien listasta, jota voitaisiin sitten lajitella Excelin lajittelutoiminnoilla välittämättä hyperlinkkien rikkoutumisesta. Tästä ”kopiolistasta” voitaisiin sitten alustaa käyttöliittymän päästökertoimien pudotusvalikkoon päästökertoimet valitussa järjestyksessä. Tätä varten käyttöliittymään tulisi lisätä joko pudotusvalikko tai valintapainike päästökertoimien järjestysperiaatteen valintaan.

Kaikista haastavin parannus hakutyökaluun olisi todennäköisesti sanahaku, joka löytää hakusanan myös sanan keskeltä tai lopusta, eikä ainoastaan alusta. Excelin VBA:n

käyttöliittymien pudotusvalikot eivät automaattisesti osaa tätä tehdä, joten tällainen työkalu olisi luotava manuaalisesti. Ensin käyttöliittymään täytyisi lisätä tekstikenttä hakusanan kirjoittamiseen, jonka syötteen perusteella sitten päivitettäisiin pudotusvalikko, joka sisältäisi kaikki ne päästökertoimet, jotka sisältävät nimessään haetun tekstin. Tämä olisi mahdollista InStr-funktion avulla, joka etsii annetusta String-muuttujasta toista String-muuttujaa ja palauttaa tämän sijainnin kokonaislukuna (monennestako merkistä alkaa). Kaikki päästökertoimet käytäisiin läpi yksi kerrallaan silmukassa ja niille suoritettaisiin testi InStr-funktion avulla. Mikäli funktio palauttaa muun arvon kuin nollan, lisättäisiin rivin päästökerroin pudotusvalikkoon. Tähän olisi myös mahdollista yhdistää laskuri, jolla käyttäjälle voitaisiin käyttöliittymässä ilmoittaa haun osumien lukumäärä. Ongelmana tässä ratkaisussa on, että on hyvin vaikeaa sanoa etukäteen, kuinka paljon InStr-funktio hidastaisi käyttöliittymän pudotusvalikon päivittämistä. Mikäli hidastuminen olisi merkittävää, hakutoiminnon lisääminen tällä tavoin ei olisi järkevää.

5.2.2 Rivien hallinnointi

Rivien_hallinnointi on yksi aivan ensimmäisistä laskuriin koodatuista moduuleista. Kun rivien lisäystoimintoja tehtiin, laskurin taulukot olivat huomattavasti yksinkertaisempia lopulliseen versioon verrattuna. Koodi perustui myös hyvin pitkälti alueiden valitsemiseen VBA:ssa .Select-metodin avulla ja näiden valintojen hyödyntämiseen .Selection-ominaisuuden avulla. Tällainen alueiden käyttäminen on tietenkin tehotonta ja epäkäytännöllistä, ja näitä on yritetty jälkikäteen koodista karsia. Koodin rakenne perustuu kuitenkin edelleen tähän lähtökohtaan.

Jälkikäteen Rivien_hallinnointi-moduulin koodia onnistuttiin parantamaan poistamalla yksittäisten taulukoiden omat rivien poistamiseen tarkoitetut makrot, sillä ymmärrettiin tämän tapahtuvan paljon helpommin yhdellä omalla funktiollaan. Vastaava yksinkertaistus ei kuitenkaan onnistu yhtä helposti rivien lisäämisen tapauksessa, sillä kaikki taulukot ovat hyvin erilaisia kooltaan ja sisällöltään. Jotta rivien lisääminen olisi onnistunut yhdellä funktiolla helpommin, olisi taulukoiden rakenne täytynyt alusta asti suunnitella yhdenmukaisemmaksi.

Yksi mahdollinen keino myös olisi siirtää lisättävien kaavojen valinta ja määritys omaan funktioonsa, jolloin tämä voitaisiin niin sanotusti ulkoistaa rivin lisäämisen suorittavasta makrosta.

Tämä kuitenkin vaatisi paljon vaivaa ja aikaa, ja nykyisen ratkaisun toimiessa tätä muutosta ei nähty tarpeelliseksi toteuttaa.

Mitä Rivien_hallinnointi-moduulille sitten kannattaisi tehdä, olisi yleinen siistiminen ja koodin yksinkertaistaminen. Makrot sisältävät edelleen aivan turhia .Select -metodeja ja ovat keskenään osittain erilaisia, vaikka tekevät lähes samoja asioita.

5.2.3 InputBox funktiot

Ohjelman kehityksen alkuvaiheilla käytettiin paljon VBA:n omaa InputBox-funktiota tiedon kysymiseen käyttäjältä. Tässä funktiossa on kuitenkin muutamia ongelmia, kuten ulkoasun puutteellisuus sekä huono ja sekava virheenkäsittely. Kun ohjelmaa kehittäessä harjaannuttiin omien käyttöliittymien luomisessa, suurimmasta osasta InputBox-funktioista luovuttiin, mutta näitä on edelleen cmdSyotakWh, imgKäytäOmaaPäästökerrointa_Napsauta ja cmdSyotaPolttoaineMuutLaitteet -makroissa. Näiden InputBox-funktioiden korvaaminen omilla käyttöliittymillä voisi parantaa ja yhtenäistää ohjelman ulkoasua sekä monipuolistaa ja yksinkertaistaa virhekäsittelyrutiineja käyttäjän syötteistä.

5.2.4 Kuljetusten syöttäminen

IKE-hiilijalanjälkilaskurin Logistiikka-välilehdellä sijaitseva ajoneuvojen lista on monella tapaa hankala käyttäjälle, ja sen taustalla käytetään enemmän koodia, kuin missään muussa yksittäisessä taulukossa. Taulukosta hankalan tekee se, että siinä käytetään paljon toistensa kanssa hyvin samankaltaisia termejä (esim. kuljetettavan tuotteen massa, kuorman massa ja Max kuorman massa), jotka tarkoittavat kuitenkin eri asioita ja vaikuttavat hyvin voimakkaasti laskennan lopputuloksiin. Taulukossa muutetaan myös laskentaperiaatteita automaattisesti eri kuljetusvälineiden tapauksissa, mikä voi hämmentää käyttäjää entisestään. Myös taulukon yläpuolella oleva kuljetusyrityksen itse ilmoittamat päästöt saattavat hämmentää käyttäjää, joka ei ole tarkoin laskentaohjeistuksiin perehtynyt.

Ratkaisuna taulukko tulisi tavalla tai toisella selkeyttää ja jollain tavalla selventää käyttäjälle, että mikäli kuljetusyrityksen ilmoittamia päästöjä on lisätty, listaan on turha tämän osamatkan

päästöjä lisätä. Tämä on myös merkittävä rajoite laskurissa, eli jos esimerkiksi kuljetusyritys ilmoittaa päästöjä osamatkalle 1, mutta tuotteen valmistaja hoitaa osan osamatka 1 kuljetuksista itse, ei tämän syöttäminen laskuriin onnistu. Tämän korjaaminen vaatisi ohjelmaan muutoksia kaikkialle, missä näiden osamatkojen välillä päästöjä luokitellaan. Alun perin nykyinen toteutus tehtiin kaksoislaskennan välttämiseksi, mutta tähän voisi olla hyvä kehittää muita keinoja.

5.2.5 Varastoinnin päästöjen laskenta

Myös Logistiikka-välilehdeltä löytyvä varastoinnin päästöjen laskentaan tarkoitettu taulukko on laskurin yksi suurimmista, monimutkaisimmista ja hankalimmista osista käyttäjän näkökulmasta. PEF-ohjeistusten mukaista laskentaa varten tarvitaan paljon esitietoja käyttäjältä, minkä vuoksi taulukko ei tahdo kunnolla edes mahtua näkyville valikon levyisenä. Varastoinnin päästöjen laskentaperiaatteessa on myös tehty paljon oletuksia ja yksinkertaistuksia, koska PEF-ohjeistuksessa ei tarkkaan määritellä tiettyjä laskennan kannalta välttämättömiä lähtöarvoja.

Varastointi on IKE-hiilijalanjälkilaskurin ainut osio, jossa arvioidaan lämmitysenergian ilmastovaikutuksia tuotteeseen. Muita lämmitysenergian kulutuksen aiheuttamia vaikutuksia esimerkiksi toimisto- tai tuotantotiloista ei huomioida, koska näiden kohdentaminen yksittäiselle tuotteelle on todella vaikeaa toteuttaa luotettavasti. Näiden ilmastovaikutusten merkitys tuotteen hiilijalanjälkeen on myös suurella todennäköisyydellä melko merkityksetön. Tämä rajaus on todennäköisesti järkevä myös tulevissa mahdollisissa laskurin versioissa.

Varastoinnin päästöjen laskenta on siltä osin ristiriitainen osa, että laskurin käyttäjäystävällisyyden parantamiseksi sitä tulisi yksinkertaistaa, mutta laskennan tarkkuuden parantamiseksi sitä pitäisi päinvastoin monimutkaistaa. Tämä asia pätee tietenkin lähes kaikkien osioiden kanssa, mutta tässä tapauksessa sopivan tasapainon löytäminen näiden kahden tekijän välillä on erityisen haastavaa. Yksinkertaisemman lähestymistavan kehittämiseksi varastoinnille on mahdollista lisätä tietoja valmiiksi määriteltyjen oletusskenaarioiden avulla, mutta nämä tekevät hyvin paljon isoja sekä pieniä oletuksia käyttäjän puolesta.

Yksi mahdollinen ratkaisu voisi olla huomattavasti nykyistä yksinkertaistetumpi taulukko, jossa käytettyihin oletusarvoihin olisi mahdollista vaikuttaa esimerkiksi käyttöliittymän avulla. Tällöin taulukkoa saataisiin pienemmäksi ja käyttäjälle yksinkertaisemmaksi, mutta samalla säilytettäisiin ja jopa parannettaisiin mahdollisuuksia tarkentaa laskentaa käyttäjän näin halutessa.

5.2.6 Painikkeiden klikkausefekti

Painikkeiden klikkausefekti on jokseenkin hankalalla tavalla toteutettu, koska painikkeiden ulkomuodon muuttamiselle klikatessa ja alkuperäisen ulkoasun palauttamiselle on laadittu omat makronsa, joista toinen ajetaan ennen varsinaisen klikkauksen laukaisevan makron toimintaa ja toinen tämän jälkeen. Tämä toteutus on ensinnäkin hankala käyttää kehittäjän näkökulmasta, ja toisekseen voi aiheuttaa ongelmia, mikäli painikkeelle määritellyn makron toiminnassa ilmenee virheitä, eikä painikkeen ulkomuodon palauttavaa makroa koskaan ajeta.

Vaihtoehtoinen toteutustapa painikkeiden klikkausefektille olisi käyttää ajastettua makroa, jolloin muutettaisiin painikkeen ulkoasua samalla tavalla, kuten NapinKlikkaus1-makrossa. Tämän jälkeen pysäytettäisiin ohjelman toiminta esimerkiksi 0,25 sekunniksi, jonka jälkeen palautettaisiin painikkeen ulkoasu makron NapinKlikkaus2 tapaan, ja tämän jälkeen aloitettaisiin varsinainen ohjelman toiminta. Tätä toteutusta yritettiin aluksi, mutta jostain syystä sitä ei tällöin saatu toimimaan halutulla tavalla.

5.2.7 Yhteenvetosivun kehittäminen

Yhteenvetosivua voisi kehittää selkeämmäksi siten, että fossiiliset ja biogeeniset päästöt eroteltaisiin selkeämmin toisistaan. (Euroopan komissio, 2021, s. 68.) Lisäksi PEF-ohjeistuksen mukaisesti biogeeniset päästöt voisi sisällyttää kokonaispäästöihin silloin, kun niiden osuus kokonaispäästöistä on alle 5 prosenttia. Toisaalta voisi olla myös hyödyllistä käyttäjälle antaa mahdollisuus itse päättää biogeenisten päästöjen eriyttämisestä tai liittämisestä kaikkiin päästöihin esimerkiksi valintakytkimen avulla.

Toinen parannus yhteenvetosivulle olisi merkittävimpien perusvirtojen, elinkaaren vaiheiden ja päästökategorioiden taulukoiden lisääminen. Tämä olisi kohtalaisen helppo muutos, sillä näiden taulukoiden muodostamiseen tarvittavat makrot ja funktiot on jo koodattu valmiiksi raportin tulostustoimintoa varten. Nämä taulukot voisivat osittain myös korvata olemassa olevaa yhteenvetotaulukkoa.

5.2.8 Allokointimenetelmät

Laskurissa on nykyisellä toteutuksella kaksi vaihtoehtoista menetelmää sivuvirtojen päästöjen allokoimiseksi: massaperusteinen ja talousperusteinen allokointi. Laskuriin voisi kuitenkin olla hyödyllistä monipuolistaa ja selkeyttää allokointia ottamalla mukaan allokointimenetelmien lisäksi allokoinnin välttämismenetelmät, jolloin käyttäjää ohjattaisiin mieluummin välttämään allokointia kuin kohdentamaan päästöjä sivuvirroille.

Allokoinnin välttämismenetelmien lisäämisen lisäksi varsinaisia allokointimenetelmiä voisi tarkentaa. Laskuri laskee kohdentamiskertoimet hyvin yksinkertaistetusti ilmoitettujen massojen perusteella tai käyttää suoraan käyttäjän itse ilmoittamia kertoimia taloudellisen allokoinnin tapauksessa. Massaperusteiseen laskentaan tuo menetelmä toimii hyvin, mutta taloudellisessa menetelmässä laskentaa kohdentamiskertoimien määrittelystä jätetään käyttäjän itse laskettavaksi.

Kaiken kaikkiaan sivuvirtojen päästöjen allokoiminen on monelle elinkaarimallintamiseen vihkiytyneellekin haastava konsepti, joten siihen liittyviä asioita tulisi pyrkiä tekemään käyttäjälle mahdollisimman yksinkertaiseksi ja ohjata käyttäjää tekemään allokointiin liittyviä valintoja oikean päätöshierarkian mukaisesti. Tästä voi lukea tarkemmin Elintarvikealan Pk-yritysten ympäristövaikutusten hallinnan oppaasta kappaleesta 3.1.4.

5.2.9 Ohjetiedoston avaamiset ja raportin tulostus

Laskurin tiettyihin toimintoihin tarvittava Excelin ja Wordin yhteistoiminta VBA:n kautta osoittautui laskinta tehdessä yllättävän haastavaksi. Raportin_tulostus-moduulin toiminnot vaikuttavat aiheuttavan silloin tällöin virheitä ilman sen kummempaa syytä. Erityisesti kuvien ja

taulukoiden kopioinnissa Excel-laskurista ja niiden liittäminen Word-raporttiin vaikuttavat aiheuttavan satunnaisesti virheitä.

Yksi suuri heikkous Word-tiedostojen käyttämisessä ohjeiden avaamiseen ja raportin tulostamiseen on myös laskurin riippuvuus näistä tiedostoista ja siitä, että ne ovat oikein nimetty eikä niiden sisältöä ole oleellisella tavalla muutettu. Laskurin sisältämät niin sanotut ”infopainikkeet” ja ohjeiden avaaminen olisi helposti toteutettavissa myös täysin Excelin sisäisesti, jolloin ohjeet tallennettaisiin esimerkiksi erilliselle välilehdelle soluihin ja infopainiketta klikkaamalla makro avaisi ponnahdusikkunan, johon ohjelma hakisi oikean tekstin toiselle välilehdelle kirjoitetuista ja määritellyistä soluista. Erilliseen ohjetiedostoon kuitenkin päädyttiin, koska se mahdollistaa laajempien ja kattavimpien ohjeiden laatimisen, sekä näiden ohjeiden päivittämisen jälkikäteen huomattavasti helpommin.

Raportin tulostamisen toiminto olisi myös teoriassa mahdollista toteuttaa ilman erillistä raporttipohjaa, mutta tämä edellyttäisi koko raportin laatimista VBA-koodissa tyhjästä Word-työkirjasta asti. Tämä vaatisi paljon aikaa ja vaivaa, sekä mahdollisesti raportin ulkoasun yksinkertaistamista. Tämä ei ole välttämättä optimaalisin tapa, mutta raportin tulostus toimintoa olisi mahdollisesti hyvä parantaa muutoin. Raporttipohjaa olisi hyvä yksinkertaistaa erityisesti vähentämällä kuvien ja taulukoiden määrää. Lisäksi koodia voi olla mahdollista toteuttaa tehokkaamminkin.

5.2.10 Tasetarkastelu

Tasetarkastelu-välilehdellä sijaitsevan kaavion päivittäminen on myös yksi IKE-hiilijalanjälkilaskurin vanhimmista ominaisuuksista. Suhteessa toiminnon yksinkertaisuuteen ja sitä toteuttavan koodin monimutkaisuuteen tämä on varmasti laskurin kehnoimmin toteutettu kokonaisuus. Tasetarkastelukaaviota toteuttava koodi on myös tehty uudelleen kolmesti laskurin kehityksen aikana. Uusien ominaisuuksien ja uusien laskettavien kohteiden myötä alkuperäinen kaavio oli huomattavasti nykyistä yksinkertaisempi. Nykyinen toteutus toimii hyvin hitaasti, päivittyy kerralla välilehden aukaisemisen yhteydessä ja kokemuksien perusteella on hyvin herkkä menemään rikki. Kun koodiin on tehty korjauksia, jotta ohjelma osaisi ottaa

erilaiset poikkeustilanteet huomioon, on ohjelma niin sanotusti alkanut rönsyilemään hyvin laajasti.

Nykyisen toteutuksen sijaan tasetarkastelukaavio olisi mahdollisesti järkevämpää toteuttaa Shape-objektien avulla, jolloin eri taulukoille olisi omat objektinsa, ja nämä täyttyisivät taulukoiden täyttämisen yhteydessä. Tällöin soluja ei tarvitsisi täyttää, lisätä tai poistaa kaavion muodostamiseksi, jolloin ohjelman suoritus nopeutuisi ja virheherkkyys vähenisi. Tätä yhdessä vaiheessa laskurin kehityksen aikana kokeiltiin, mutta tuolla hetkellä ongelmia tuotti Shape-objektien sijainnit ja niiden yhdistäminen nuolilla virtauksien suuntien mukaisesti. Kaavion päivittyessä objektien sijainnit eivät pysyneet oikeissa kohdissa. Tämä ongelma voisi olla kuitenkin kierrettävissä muuttamalla objektien sijainteja toistensa suhteessa VBA-koodin avulla niin sanottuja tasaustoimintoja käyttäen.

5.3 Uudet ominaisuudet

Laskuriin voisi olla hyödyllistä myös kehittää kokonaan uusia toiminnallisuuksia ja laajentaa laskentaa käsittämään paremmin PEF-ohjeistusten mukaisesti tuotteen elinkaaren aikaiset päästöt. Laskentamenetelmän nykyisessä toteutuksessa on muutamia puutteita, joiden korjaaminen voisi lisätä laskimen tarkkuutta ja parantaa tulosten luotettavuutta sekä laskurin käytettävyyttä.

5.3.1 Yhteensopivuus Mac-tietokoneiden kanssa

IKE-hiilijalanjälkilaskurin viimeisissä kehitysvaiheissa havaittiin, että laskuri ei ole kaikkien toimintojen osalta yhteensopiva Mac-tietokoneiden käyttöjärjestelmissä. Pintapuolisten testien perusteella kuitenkin kaikkein tärkeimmät ominaisuudet toimivat myös Macilla, joten niin sanotusti Mac-yhteensopivan laskurin version tekeminen on luultavasti mahdollista.

Tätä varten monia laskurin toimintoja olisi kuitenkin muokattava tai jopa poistettava kokonaan. Esimerkiksi Wordin kanssa toimiminen Excelin VBA:n kautta tuntuu hyvin haastavalta Macilla, muistiinpanot työkalussa on ongelmia ja kehittäjätilaan siirtyminen aiheuttaa virheen.

Nämä eivät kuitenkaan ole laskurin kannalta täysin välttämättömiä toimintoja, sillä tuotteen hiilijalanjäljen saa laskettua ilman näitäkin.

Jos laskurista halutaan tehdä Mac-yhteensopiva, kaikista helpoin tapa on tehdä kokonaan uusi Excel-tiedosto, josta on muokattu tai poistettu toimimattomat osat. Toinen vaihtoehto olisi koodata laskuriin makro, joka havaitsisi mikä käyttöjärjestelmä on käytössä ja muuttaisi kehittäjä- / laskentatilan tapaisesti asetuksia siten, että tietyt toiminnot poistettaisiin Macilla käytöstä. Ensimmäinen vaihtoehto on kuitenkin ehdottomasti helpompi toteuttaa ja olisi toimintavarmempi.

5.3.2 CFF-kaavojen hyödyntäminen

Kiertojalanjäljen laskentakaava (Circular Footprint Formula eli CFF-kaava) on kolmen kaavan kokonaisuus, joiden avulla tulisi mallintaa kierrätysmateriaalien käyttöä ja käytöstä poistamista (Euroopan komissio, 2021). Tämä on jätetty laskurista pois, mutta se voisi olla hyödyllinen lisä sekä jätteiden ja sivuvirtojen mallinnuksessa, että mahdollisessa elinkaaren loppuvaiheiden mallinnuksessa, joka toki liittyy hyvin merkittävällä tavalla jätteet ja sivuvirrat -kategoriaan.

CFF-kaavojen sisältö, käyttötarkoitus ja käytännön hyödyntäminen saattavat PEF-ohjeistusten perusteella jäädä hieman hämäriksi, mutta saksalainen konsulttiyritys GreenDelta (Rickert. J, 2020) on esitellyt kotisivuiltaan löytyvässä artikkelissa CFF-kaavat ymmärrettävämmällä tavalla. Sivustolta löytyy myös Excel-laskuri CFF-kaavojen hyödyntämiseen.

5.3.3 ”Portilta-hautaan” mallinnus

Tällä hetkellä IKE-hiilijalanjälkilaskuri ottaa rajaa tarkastelun niin sanottuun ”kehdosta-portille” -tarkasteluun. Tällöin elinkaaren loppuvaiheen päästöjä esimerkiksi tuotteen käyttövaiheesta ja loppusijoituksesta ei huomioida. Laskenta haluttiin rajata nimenomaan yrityksen tuottamiin päästöihin, jolloin ns. kuluttajavaihetta ei huomioida elintarvikeprosessissa syntyneeksi päästöksi.

Mikäli elinkaaren myöhemmät vaiheet halutaan laskurissa huomioida, tulisi laskuriin tehdä muutoksia Jätteet ja sivuvirrat, Energia ja Logistiikka -välilehdille tai vaihtoehtoisesti luoda tuotteen käyttövaiheelle ja loppusijoitukselle. Jätteissä ja sivuvirroissa tulisi huomioida mahdollinen hävikki ja tuotteen pakkausmateriaalien käsittely, energiassa esimerkiksi tuotteen valmistamiseen tai lämmittämiseen vaadittu energia ja logistiikassa tarkemmin kuljetukset ja varastointi käyttö- ja loppusijoitusvaiheista. Tällä hetkellä laskuri kykenee mallintamaan varastointia ja kuljetuksia myös tehtaalta jälleenmyyjän kautta kuluttajalle, mutta kuljetus ja varastointi loppusijoitukseen päätyvästä materiaalista puuttuu.

5.3.4 Alkutuotannon huomioiminen laskennassa

Nykyisellä toteutuksella IKE-hiilijalanjälkilaskuri huomioi raaka-aineiden ja pakkausmateriaalien päästöjä ainoastaan niille määriteltyjen päästökertoimien avulla. Laskurilla ei ole mahdollista mallintaa alkutuotannon päästöjä tarkemmin muutoin, kuin tekemällä jokaiselle raaka-aineelle oman mallinnuksen ja lisäämällä näiden mallinnusten tulokset päästökerrointietokantaan ja käyttämällä näitä tuloksia tietokannan kautta varsinaisen lopputuotteen hiilijalanjäljen laskennassa. Tällöinkin laskurista puuttuu tiettyjä alkutuotannolle tyypillisiä päästölähteitä. Alkutuotannon mallintaminen suoraan laskurissa ilman useita erillisiä laskentoja voisi olla hyödyllinen lisä, mikäli se pystytään tekemään ilman laskurin kohtuutonta monimutkaistamista.

Teoriassa alkutuotannossa on useita samoja päästölähteitä, kuin varsinaisen tuotteen valmistuksessa, koska jokainen yksittäinen raaka-ainekin on käytännössä tuote. Laskentavälilehdillä tulisi kuitenkin huomioida paremmin perusvirtausten luokittelu eri tuotannon vaiheiden kesken, jotta alkutuotannon päästöt olisivat erotettavissa prosessointivaiheen päästöistä. Joitakin tiettyjä yksityiskohtia tulisi myös lisätä laskentaan, kuten käytettyjen lannoitteiden ja torjunta-aineiden huomioiminen ja maankäytön vaikutukset.

Alkutuotannon monipuolisempi huomioiminen laskennassa olisi mahdollista toteuttaa joko omalla laskentavälilehdellään, jossa käsiteltäisiin alkutuotannon kannalta oleelliset tekijät erikseen, tai sisällyttämällä nämä asiat olemassa oleville välilehdille. Kehittäjän kannalta kaikista helpoin ratkaisu olisi toteuttaa näistä jälkimmäinen ja lisätä esimerkiksi yhteenvetosivulle painike, jolla laskennan tulokset voi liittää suoraan päästökerrointietokantaan. Tässä

huonona puolena on se, että mikäli mallinnettavia alkutuotantoja eri raaka-aineille on useita, menee laskenta hyvin hankalaksi tällä periaatteella.

5.3.5 Muiden vaikutusluokkien sisällyttäminen laskentaan

IKE-hiilijalanjälkilaskuri huomioi nimensä mukaisesti vain yhden elinkaarimallintamisen vaikutusluokista eli ilmastonmuutos-vaikutusluokan laskemalla tuotteen hiilijalanjäljen. PEF-ohjeistuksen mukaiseen elinkaarimallinnukseen tulisi huomioida kaikki 16 olemassa olevaa vaikutusluokkaa, joten näiden tai edes osan niistä huomioiminen laskentamenetelmässä olisi merkittävä parannus.

Teoriassa tämä ei edes vaatisi kovin suuria muutoksia laskurin rakenteeseen, sillä laskurin kerätessä aktiviteettidatan sen tulisi enää osata hyödyntää erilaisia päästökertoimia ja karakterisointikertoimia, jotta voitaisiin laskea muiden vaikutusluokkien vaikutuksia. Kaikista haastavinta tässä muutoksessa olisi löytää riittävän kattavasti näitä kertoimia, jotta laskuria olisi mielekästä käyttää. Ottaen huomioon, kuinka haastava tehtävä kertoimien koostamien pelkästään hiilijalanjäljen laskentaa varten oli, tämä voi osoittautua toistaiseksi hyvin hankalaksi tehtäväksi.

Jos kuvitellaan, että tarvittavia kertoimia olisi saatavilla ja laskuria voitaisiin lähteä laajentamaan, tulisi laskuriin tehdä muutamia rakenteellisia muutoksia. Välilehdillä voitaisiin edelleen kerätä aktiviteettidataa prosessista, mutta kaikkien erilaisten kertoimien sisällyttäminen laskentavälilehdille ei välttämättä enää olisi järkevää ainakaan siten, että nämä kertoimet ovat käyttäjälle näkyvillä. Mikäli kaikki kertoimet jätettäisiin näkyville, tulisi useista taulukoista niin leveitä, etteivät ne mahtuisi kunnolla enää näytölle. Jotta laskurin ulkoasu ei kärsisi, kertoimet tulisi olla joko laskentavälilehdillä piilotettuna käyttäjältä tai vaihtoehtoisesti kokonaan omalla välilehdellään.

Kahdesta yllä esitetystä vaihtoehdosta erillinen välilehti eri vaikutusluokkien kertoimia ja karakterisointimalleja varten voisi olla järkevämpi vaihtoehto, sillä tällöin nämä tiedot olisivat kootusti nähtävillä käyttäjälle, joka pääsisi niitä halutessaan muokkaamaan. Piilotettujen

kertoimien tapauksessa käyttäjän mahdollisuudet muokata laskurin käyttämää dataa vähenee, mikä ei ole tarkoituksenmukaista.

5.3.6 Laskurin siirtäminen selainpohjaiseksi

VBA:n merkittävä heikkous on, että se toimii kunnolla ainoastaan Officen ohjelmien työpöytäversioilla. Tarvittavat tiedostot tulee olla aina ladattuna, oikein nimettyinä ja oikeassa kansiossa muiden tiedostojen kanssa, jotta laskuri toimii oikein. Excel-pohjainen laskuri toimii myös ainoastaan, mikäli makrot ovat toiminnassa ja käyttäjän tietoturva-asetukset sallivat niiden toiminnan. Excel-pohjaisen laskurin ongelmana on myös käyttäjien tottumukset, sillä joillekin käyttäjille hyvin voimakkaasti automatisoitu ja rajoitettu Excel-tiedosto voi aiheuttaa hämmennystä, mikäli on totuttu käyttämään Exceliä vapaammin ja yksinkertaisempiin tehtäviin. Luonnollisesti laskurin käyttäminen ei myöskään onnistu, mikäli käyttäjältä ei löydy Microsoft Officen lisenssiä.

Näihin ja moneen muuhun ongelmaan ratkaisu olisi siirtää IKE-hiilijalanjälkilaskuri pois Excelistä. Vaikkakin VBA:n kaltaisesti selaimessa toimivilla Office Scripteillä olisi mahdollista tehdä lähes samat asiat kuin nykyisessä laskurissa VBA:lla, näissä on tiettyjä heikkouksia, joiden takia Office Scriptit eivät välttämättä ole ratkaisu. Esimerkiksi nämä eivät mahdollista vastaavia tapahtumiin reagoivia ohjelmia, vaan kaikki Office Scriptit vaativat käyttäjän aktivoimaan niiden toiminnan, vaikkakin tätä ongelmaa voisi olla mahdollista kiertää Power Automate:n avulla. Vaikka emme ole perehtyneet Office Scripteihin ja Power Automate -toimintoihin kovinkaan syvällisesti, voisi kuitenkin laskurin irrottaminen Excelistä ja Officen työkaluista kokonaan olla viisain ratkaisu.

Helppoa tällainen siirtymä ei varmasti ole, sillä laskurin toiminnot ovat suunniteltu ja tehty hyvin pitkälti VBA:n ja Excelin ominaisuuksien ja rakenteen puitteissa. Osa toiminnoista jouduttaisiin todennäköisesti muuttamaan tai jopa poistamaan, eikä makrojen ja funktioiden koodeja voida hyödyntää kovin paljon täysin eri koodikielillä tehtävässä selainpohjaisessa laskentamenetelmässä. Kuitenkin laskurin muuttaminen selaimessa käytettäväksi olisi niin merkittävä parannus sen käytettävyyden kannalta, että se olisi ehdottomasti kannattavaa, mikäli tällaiseen muutokseen vaadittavat resurssit ovat olemassa.

LÄHTEET

Bentzen, E. (i.a) Sort hyperlinks in Excel with VBA macros. https://sitestory.dk/excel_vba/sorting-hyperlinks.htm

Euroopan Komissio. (2021). Komission suositus (EU) 2021/2279, ympäristöjalanjälkeä koskevien menetelmien käyttämisestä tuotteiden ja organisaatioiden elinkaaren ympäristöte-hokkuuden mittaamiseen ja siitä tiedottamiseen. s.48-51, 56, 68, 92, 99-100 https://eur-lex.europa.eu/legal-con-tent/FI/TXT/PDF/?uri=CELEX:32021H2279&from=FI

Luonnonvarakeskus. (i.a.). Ilmastovaikutusdatasetti ruokapalvelusektorille: FOODGWP. https://www.luke.fi/fi/projektit/foodgwp

My Excel Genious (i.a) VBE Colors: How to code VBA on the ‘Dark Side’. https://myexcelgenius.com/coding-vba-on-the-dark-side/

Rickert, J. (2020) Understanding and applying the Circular Footprint Formula (CFF) in Product Environmental Footprints (PED) https://www.greendelta.com/understanding-and-applying-the-circular-footprint-formula-cff-in-product-environmental-footprints-pef/