Tieteellisen tekstin tuottaminen LaTeXilla

Fysiikan typografiset menetelmät

Perttu Luukko

JYFL, kesä 2015

Tämä on vuoden 2015 versio Jyväskylän yliopiston fysiikan laitoksella noin vuoden välein järjestetyn kurssin luentomateriaalista. Uusin versio tästä dokumentista löytyy osoitteesta `http://iki.fi/perttu.luukko/latexkurssi-handout.pdf`. (Tulostusvalmis versio: `latexkurssi-printable.pdf`)

Mikäli löydät virheitä tai puutteita, tai jos sinulla on parannusehdotuksia tai jotain ihan muuta sanottavaa, lähetä sähköpostia osoitteeseen `perttu.luukko@iki.fi`.

- Opettaja: Perttu Luukko (`perttu.luukko@iki.fi`), huone FL353.
- Kurssin nettisivut:
  `http://users.jyu.fi/~akjujoki/fi/opetus/latex/`
- Nämä luentokalvot ovat saatavilla kurssin nettisivuilta.
- Kurssi sisältää 6 kahden tunnin luentoa sekä 6 kahden tunnin harjoitussessiota.
- Kurssista on saatavilla 2 op. Kurssin suorituksen ehtona on harjoitustehtävien hyväksytty suorittaminen. Kurssi arvostellaan asteikolla hyväksytty/hylätty.
- Harjoituksista vastaa Aku Jokinen (`aku.jokinen@iki.fi`).
- Harjoitustehtävät on palautettava viimeistään 16.8.2015.

- Kukin luennoista käsittelee yhden LaTeXin käytön osa-alueen, alkaen perusteista ja siirtyen tavallisen tekstin, kuvien ja matematiikan kautta esitysgrafiikkaan.
- Viimeinen luento esittelee sekalaisia vinkkejä ja yksityiskohtaisempia temppuja, jotka on ohitettu aiemmilla luennoilla.
- Luentojen otsikot ovat:

Perinteinen luento ei ole erityisen hyvä tapa välittää käytännön taitoja kuten LaTeXin käyttö.

- Harjoitustehtäviin kannattaa panostaa – niiden kautta asia oikeasti opitaan.
- Kysymyksiä kannattaa esittää – niin luentojen aikana kuin muinakin aikoina.
- Itsenäinen tiedonhaku kannattaa opetella – se on paras ja ainoa tie todelliseen osaamiseen.

Tämä luentomateriaali ei ole esimerkki hyvistä luentokalvoista eikä hyvistä luentomuistiinpanoista. Tämä dokumentti yrittää kelvata välttävästi molempiin tarkoituksiin.

- Perustasoisen, suomenkielisen oppaan LaTeXin käytöstä tarjoaa Antti-Juhani Kaijanahon mainio, joskin hieman vanhentunut, *LaTeX ja AMS-LaTeX. Opus asiatekstin ladonnasta*. Kirja on saatavilla Jyväskylän yliopiston yliopistokauppa Sopista opiskelijaystävälliseen hintaan.
- Ilmaista dokumentaatiota LaTeXista on netti väärällään. Mainittavan arvoisia osoitteita ovat ainakin:
  - *Wikibooks: LaTeX*:
    `http://en.wikibooks.org/wiki/LaTeX`
  - *The Not So Short Introduction to LaTeX 2ε*:
    `http://mirrors.ctan.org/info/lshort/english/lshort.pdf`
  - *Pitkänpuoleinen johdanto LaTeX 2ε:n käyttöön*:
    `http://mirrors.ctan.org/info/lshort/finnish/lyhyt2e.pdf`
  - *The Comprehensive TeX Archive Network*:
    `http://www.ctan.org/`

Osa I

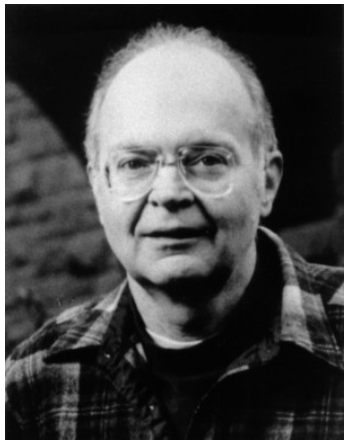
Alkeet

Sisällys

Pelin henki ja homman nimi

Esipuhe

Ennen teknisempää johdatusta LaTeXin maailmaan on syytä perehtyä hieman LaTeXin historiaan, ja erityisesti siihen, mikä erottaa LaTeXin useimmille tutuista tekstinkäsittelyohjelmista. Tämä siksi, että kurssin alku tapahtuu mahdollisimman pehmeästi, eikä mitään elintärkeää nippelitietoa anneta kurssin ensimmäisillä minuuteilla. Toinen hyvä syy olla aloittamatta johdatusta aivan *ad rem* on se, että joidenkin tiukan pragmaattisten LaTeX-johdatusten seurauksena maailmassa on ihmisiä, jotka ehkä saavat selkkarinsa kirjoitettua LaTeXilla, mutta *eivät tiedä mikä se on*.

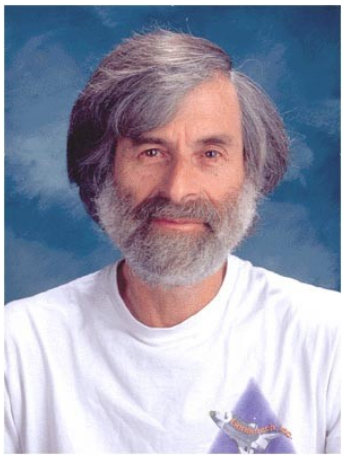


Donald Ervin Knuth

TEXin synty

Vuonna 1977 yhdysvaltalainen tietojenkäsittelytieteilijä Donald E. Knuth suunnitteli maailmankuulun pääteoksensa *"The Art of Computer Programming"* toisen painoksen julkaisemista. Knuth ei kuitenkaan ollut tyytyväinen tuon ajan ladontajärjestelmien laatuun, joten hän päätti suunnitella itse paremman. Näin syntyi TEX, jonka ensimmäinen versio julkaistiin vuonna 1978.

- TEX toi ammattilaistason tekstinladonnan kaikkien saataville, maksutta.
- TEX saavutti ominaisuuksien puolesta lopullisen muotonsa vuonna 1989, jolloin Knuth julkaisi TEX 3.0:n. Pienten yksityiskohtien hienosäätöä kuitenkin edelleen tapahtuu, ja sen mukana TEXin versionumero lähestyy asymptoottisesti lukua π.
  - Nykyinen TEX 3.14159265 julkaistiin vuonna 2014.
  - Knuthin kuollessa julkaistaan lopullinen TEX π, joka on määritelmän mukaan virheetön.
- TEX on pitkän elämänsä aikana tuottanut lukuisia jälkeläisiä, laajennoksia ja johdannaisia, jotka tulevat jatkamaan kehitystään hamaan tulevaisuuteen saakka.



Leslie Lamport

LATEXin alku

Helpottaakseen TEXin käyttöä toinen yhdysvaltalainen tietojenkäsittelytieteilijä Leslie Lamport kirjoitti 1980-luvun alkupuolella oman laajennoksensa LATEXin, tarkoituksenaan antaa käyttäjän keskittyä itse tekstin sisältöön, jättäen ladontatekniikan yksityiskohtaisemmat puolet tietokoneen huoleksi.

- TEXin toiminta tapahtuu melko matalalla tasolla, mutta LATEXin avulla käyttäjä suunnittelee *sisällön* ja kuvailee LATEXin komentoja käyttämällä tekstin *rakenteen* (”Tässä vaihtuu kappale, tässä alkaa uusi luku nimeltään Johdanto”). Tämän jälkeen LATEX kertoo TEXille mitä kirjoitetaan ja minne, ja TEX suorittaa lopullisen ladonnan 30-vuotisella kokemuksellaan.
  - Mielikuva: Kirjoittaja toimittaa kirjapainoon käsin kirjoitetun nivaskan tekstiä, jossa on seassa huomautuksia ja toiveita koskien lopullista ulkoasua. Painotalo huolehtii yksityiskohdista.
- TEXin käyttö tapahtuu nykyään yleensä juuri LATEXin kautta.
- LATEXia kehitetään edelleen – monia uusia ominaisuuksia sisältävä LATEX3 on työn alla (ja on ollut jo 20 vuotta…).

- Alkuperäisen TEXin nimen takana on kreikan kielen taitoa, taidetta ja tekniikkaa tarkoittava sana τέχνη.
- Itse Knuth on varsin tarkka siitä, että TEX tulisi lausua alkuperäiskieltään jäljitellen kuten *tech*, missä *ch* lausutaan kuten gaelin sanassa *loch*.
- Suomalaisittain TEX vääntyy usein muotoon *teh*.
- LATEXin luoja Leslie Lamport on puolestaan sanonut ettei suosi tai karsasta mitään lausumisasua, joten LATEXin voi kuulla lausuttavan puhujasta riippuen esimerkiksi *latech*, *leitech*, *lateh*, *leiteh*, *latek*, *leitek*, *lateks* tai *leiteks*.

- LaTeX (oikeastaan TeX) tarjoaa erinomaisen ladontajäljen, erityisesti matematiikan osalta.
- Mahdollisimman suuri osa ”tekstinkäsittelyn” yksityiskohtien nyhertämisestä jää tietokoneen vastuulle (rivitys, sijoittelu, sivunvaihdot, sisäiset viittaukset, sisällysluettelot, hakemistot…), ja itse kirjoittaja voi keskittyä sisältöön.
- LaTeX-koodi on pelkkää tekstiä. Dokumentin sisältö ja muotoilu näkyvät käyttäjälle, eikä niitä ole piilotettu jonkin sekavan graafisen käyttöliittymän taakse.
- LaTeX taipuu hyvin monenlaisten dokumenttien ladontaan, kunhan dokumentilla on selvä rakenne. Sopivilla laajennoksilla LaTeX latoo niin kirjoja, esityskalvoja, shakkiotteluita kuin sudoku-ristikoitakin. Kaikkeen se ei tietenkään sovellu.
- LaTeX, kuten TeX, on ilmainen ja vapaasti käyttäjiensä muokattavissa ja kehitettävissä.

- Käyttäjä kertoo: *Hakemistoon viittaus tähän kohtaan tekstiä tällä nimellä.*
  - → Tietokone latoo dokumentin loppuun hakemiston, ja lisää hakemistoon käyttäjän ilmoittaman hakusanan sekä sivunumeron, jossa se mainitaan.
- Käyttäjä kertoo: *Haluaisin tästä luentoesityksestä tulostettavan version, eli siis ilman väriefektejä ja krumeluureja, ja neljä kalvoa per sivu.*
  - → Tietokone latoo tulostettavan version, ilman että käyttäjän tarvitsee alkaa nyhertämään yksitellen jokaista kalvoa yksiväriseksi, poistelemaan navigointielementtejä jne.

☹☹ LaTeX ja TeX ovat *muinaisia*, ja monissa asioissa se kyllä näkyy. Toistaiseksi kukaan ei ole kuitenkaan riistänyt planeetan herruutta näiltä dinosauruksilta.

☹ Siinä missä lähes kuka hyvänsä ymmärtää tavallisen tekstinkäsittelyohjelman nähdessään kuinka se suunnilleen toimii, LaTeXin käyttöönotto tarvitsee pientä perehtymistä käyttäjän osalta.

☹ Ajan kuluessa yhä suurempi osa LaTeXin toiminnallisuudesta on toteutettu kolmansien osapuolien tekemien laajennuspakettien avulla. Kun eri vuosikymmenillä eri ihmisten kirjoittamat koodit yrittävät tulla toimeen keskenään, syntyy väistämättä kummallisia yhteensopimattomuusongelmia.

- Lyhyt vastaus: LaTeX on *kieli*, jolla esitetään dokumentin ulkoasu ja rakenne.
- Ohjelmointia harrastaneille idea on tuttu:
  - Käyttäjä kirjoittaa *koodin*, joka kuvailee ihmisen ymmärtämässä muodossa mitä halutaan.
  - Jokin tietokoneohjelma, *kääntäjä*, tulkkaa tämän koodin tietokoneen ymmärtämään, lopulliseen muotoon (LaTeXin tapauksessa valmis PDF-tiedosto).
- LaTeX *ei* siis varsinaisesti ole tekstinkäsittelyohjelma kuten Microsoft Word, LibreOffice tai vastaavat.

Edellisen sivun tuotti seuraavanlainen pätkä LaTeX-koodia

```
\begin{frame} % JOS MUUTAT TÄTÄ, muista muuttaa myös seuraava esimerkki! :)
  \frametitle{Kauniita sanoja, mutta mikä se \LaTeX\ siis on?}
  \begin{itemize}
    \item Lyhyt vastaus: \LaTeX\ on \emph{kieli}, jolla esitetään dokumentin
      ulkoasu ja rakenne.
    \pause
    \item Ohjelmointia harrastaneille idea on tuttu:
      \begin{itemize}
        \item Käyttäjä kirjoittaa \emph{koodin}, joka kuvailee ihmisen ymmärtämässä
          muodossa mitä halutaan.
        \item Jokin tietokoneohjelma, \emph{kääntäjä}, tulkkaa tämän koodin
          tietokoneen ymmärtämään, lopulliseen muotoon\\(\LaTeX in tapauksessa
          valmis PDF-tiedosto).
      \end{itemize}
    \pause
    \item \LaTeX\ \emph{ei} siis varsinaisesti ole tekstinkäsittelyohjelma kuten
      Microsoft Word, LibreOffice tai vastaavat.
  \end{itemize}
\end{frame}
```

- Kirjoittaessasi LaTeXia kirjoitat tavallista, muotoilematonta tekstiä, jonka sekaan kirjoitat toiveesi dokumentin ulkoasusta LaTeXin ymmärtämässä muodossa (edellisen esimerkin kenoviivalla alkavat sanat).
- Kun haluat muuntaa koodisi esimerkiksi PDF-dokumentiksi, annat koodin erityisen kääntäjäohjelman käsiteltäväksi.
- Itse LaTeX-koodin kirjoittamiseen ei ole olemassa mitään tiettyä virallista ohjelmaa – voit käyttää mitä hyvänsä tekstieditoria.
- LaTeXin käyttämiseen on kuitenkin tehty erityisesti sille tarkoitettuja tekstieditoreja, joista löytyy esimerkiksi valikot ja nappulat yleisimmille komennoille sekä koodin kääntämiselle.

LaTeX-koodin perussyntaksi

Esipuhe

Seuraavassa kappaleessa perehdytään LaTeX-kielen perusrakenteeseen. Kappale voi tuntua hieman tekniseltä, mutta sen ei kannata antaa häiritä. Tarkoituksena on vain esitellä yleistä termistöä, eli esimerkiksi mitä tarkoitetaan *komennolla*, *ympäristöllä* tai *paketilla*. Kun termistö on tuttua, voidaan mennä itse asiaan eli tekstin kirjoittamiseen, jonka myötä syntaksinkin yksityiskohdat oppii kokemuksen kautta.

- LaTeX-koodi on tavallista tekstiä, jonka seassa on muotoiluun vaikuttavia *komentoja*.
- LaTeX-komennot alkavat kenoviivalla (\), ja päättyvät seuraavaan merkkiin, joka ei ole kirjain. Esimerkiksi `\LaTeX` on LaTeXin ymmärtämä komento, jolla saa aikaan LaTeX-logon.
- Lisäksi joillain erikoismerkeillä (%, &, $…) on oma merkityksensä LaTeXissa.
- Jatkon esimerkeissä käytetään selkeyden vuoksi merkkiä ␣ kuvaamaan välilyönnin paikkaa, silloin kun välilyöntiä on syytä korostaa.

Huom!

Komennoissa isoilla ja pienillä kirjaimilla on eroa!

- Mikäli välittömästi komennon jälkeen tuleva merkki on välilyönti, ei välilyöntiä tulosteta. Tämä siksi, että voitaisiin latoa jotain kirjaimia heti komennon tulosteen jälkeen.
- Mikäli komennon jälkeen nimenomaisesti halutaan välilyönti, voidaan käyttää normaalin välilyönnin pakottavaa komentoa `\␣` (kenoviiva välilyönti).

Esimerkki

Jos halutaan sanoa ”Rakastan LaTeXia valtavasti”, ei voida kirjoittaa

```
Rakastan \LaTeXia valtavasti!,
```

sillä `\LaTeXia` ei ole mikään komento. Tämän sijaan tulee kirjoittaa

```
Rakastan \LaTeX␣ia valtavasti!,
```

sillä komentoa `\LaTeX` seuraava välilyönti jätetään huomiotta.

- Jotkin komennot ottavat yhden tai useamman *argumentin*, jotka kirjoitetaan komennon jälkeen aaltosulkuihin (`{` ja `}`).
- Mikäli argumentteja on useampia, kirjoitetaan kukin omiin sulkuihinsa, eli muotoon `\komento{argumentti1}{argumentti2}`.
- Oikeastaan aaltosulut vain ryhmittelevät sisältönsä yhdeksi ”yksiköksi”, ja komennot vaikuttavat seuraavaan koodista löytyvään yksikköön. Ilman sulkuja tämä olisi seuraava kirjain.

Esimerkki

Tekstiä korostava komento `\emph` vaati argumentikseen korostettavan tekstin: Esimerkiksi koodinpätkä

```
Rakastan \LaTeX ia \emph{valtavasti!}
```

Tuottaa seuraavaa:

Rakastan LaTeXia *valtavasti!*

- Komennot voivat ottaa myös *valinnaisia* argumentteja, joka tavallisesti kirjoitetaan hakasulkuihin (`[` ja `]`).
- Jos komento sallii useamman valinnaisen argumentin, ne kirjoitetaan yleensä pilkuilla erotettuina samoihin sulkuihin, eli muotoon `\komento[varg1,varg2]`.
- Mikäli sama komento ottaa sekä valinnaisen että pakollisen argumentin, tulee valinnainen argumentti yleensä pakollista ennen.
- ☹ Kuten edellisistä näkyy (”joskus” näin, ja ”yleensä” noin), eri komennot saattavat noudattaa hieman eri käytäntöjä argumenttien vastaanottamisen suhteen.

Varo vaaraa!

Joillakin tekstin seassa käytettävillä erikoismerkeillä on oma merkityksensä LaTeXissa. Erikoismerkkien kanssa kannattaa olla varovainen, sillä esimerkiksi kenoviivan käyttäminen kesken tekstiä saa LaTeXin tulkitsemaan sitä seuraavan sanan komennoksi.

Esimerkkejä erikoismerkeistä

Ainakin seuraavien kanssa kannattaa olla tarkkana:

```
\  {  }  $  %  &  ~  ^  _
```

- Prosenttimerkki (%) tulkitaan LaTeXissa siten, että kaikki koodi prosenttimerkistä seuraavan rivin alkuun jätetään huomiotta. Prosenttimerkkiä käyttäen voi siis jättää koodiin kommentteja, tai ”deaktivoida” rivejä väliaikaisesti.
- Mikäli LaTeXin tulosteeseen haluaa prosenttimerkin, esimerkiksi kirjoittaakseen lauseen *Panostan LaTeX-kurssiin 100% ajastani*, tulee käyttää komentoa \%.

Esimerkki

Voit kirjoittaa LaTeX-koodiin esimerkiksi rivit

```
% Muistahan vielä siivota sinä-passiivit pois seuraavasta
% tekstikappaleesta ettei assari revi taas peliverkkareitaan.
```

eikä lopullisessa dokumentissa näy tästä jälkeäkään!

- Usein tarvitaan komentoja, joiden vaikutusalueena on kokonainen pala tekstiä. Tällöin käytetään *ympäristöjä*.
- Ympäristö aloitetaan komennolla `\begin{ympäristön nimi}` ja päätetään komennolla `\end{ympäristön nimi}`.
- Ympäristölle annettavat pakolliset tai valinnaiset argumentit kirjoitetaan `\begin`-komennon argumenteiksi.

Esimerkki

Tekstissä olevat lainaukset voi sisällyttää `quote`-ympäristöihin, jolloin LaTeX osaa latoa ne esimerkiksi sisennettynä ja kursiivilla:

Koodi

```
\begin{quote}
There are no limits. There are
plateaus, but you must not stay
there, you must go beyond
them.\\
--- Bruce Lee
\end{quote}
```

Tulos

> *There are no limits. There are plateaus, but you must not stay there, you must go beyond them.*
> *— Bruce Lee*

Kohti ensimmäistä LaTeX-dokumenttia

Esipuhe

Koodin perussyntaksin lisäksi on syytä tietää jotain LaTeX-dokumenttien yleisestä rakenteesta. Tässä kappaleessa päästään vihdoin esittelemään lyhyitä esimerkkejä kokonaisista LaTeX-dokumenteista, sekä esittelemään LaTeX-koodin saattaminen PDF-tiedostoksi. Kunnollista sisältöä ensimmäiseen LaTeX-dokumenttiimme saamme kuitenkin vasta seuraavassa osassa, jossa esitellään tavallisen leipätekstin kirjoittaminen.

- Jokainen LaTeX-dokumentti alkaa komennolla `\documentclass`, joka kertoo LaTeXille dokumentin tyylin. Komento ottaa pakollisena argumenttinaan tyylimäärittelyn nimen.
- Valmiita tyylimäärittelyjä ovat esimerkiksi `article` ja `book`, jotka antavat pohjan artikkeli- ja kirjamuotoisille dokumenteille.
- `\documentclass` ottaa myös valinnaisia argumentteja, joilla voidaan määrätä joitain yleisiä tekstin asetuksia.

Esimerkki dokumenttityylin määrittelystä

Rivi

```
\documentclass[a4paper,12pt]{article}
```

dokumentin alussa julistaa dokumentin käyttävän valmista `article`-pohjaa siten, että leipätekstin fonttikoko on 12, ja lopullinen dokumentti tulee A4-kokoiselle paperille.

- Sisältöä ei päästä kirjoittamaan heti komennon `\documentclass` jälkeen – varsinaiseen dokumenttiin päästään vasta kun on aloitettu ympäristö `document`.
- Koodia `\documentclass`:in ja dokumentin aloittavan `\begin{document}`:in välissä kutsutaan *alustukseksi* (engl. *preamble*).
- Alustuksessa tyypillisesti tehdään dokumentin yleiseen ulkoasuun vaikuttavat määritykset ja ladataan iso joukko LaTeXia laajentavia *paketteja*, joista lisää pian.

Seuraavassa koodipätkässä on kaikki LaTeX-dokumentille oleellinen materiaali.

Esimerkki validista LaTeX-dokumentista

```
\documentclass[a4paper,12pt]{article}          ⟵ Tyylimäärittely
                                               ⟵ Tässä välissä on alustus
\begin{document}
  Hello World!                                 ⟵ Itse sisältö
\end{document}
```

Eihän näytä monimutkaiselta?

- Perus-LaTeX ei sellaisenaan useinkaan riitä. Tyypillisesti LaTeXia laajennetaan joukolla *paketteja*, jotka muuttavat LaTeXin käyttäytymistä ja määrittelevät uusia komentoja.
- Paketti ladataan komennolla `\usepackage`, joka lisätään dokumentin alustusosaan. Komento ottaa pakollisena argumenttinaan paketin nimen.
- Usein pakettien käyttäytymiseen voidaan vaikuttaa antamalla `\usepackage`-komennolle valinnaisia argumentteja.

Esimerkki

Oletetaan, että edellisen minimalistisen esimerkin latoneen täyttää yhtäkkinen palava halu saada dokumenttiinsa aiemmin konduktanssin yksikkönä käytetyn mhon symboli ℧. Koska väärinpäin olevaa isoa omegaa ei perus-LaTeXista löydy, löytyy se jostain paketista. Oikea paketti tässä tapauksessa on ison joukon erikoisia symboleja sisältävä `textcomp`:

Hello mho!

```
\documentclass[a4paper,12pt]{article}
\usepackage{textcomp}
\begin{document}
Hello W\textmho rld!
\end{document}
```

Tulos

Hello W℧rld!

- Kuten aiemmin mainittua, LaTeX-koodi on vain tekstiä, ja siten voimme käyttää koodin kirjoittamiseen mieleistämme tekstieditoria. Mitään erillistä ”LaTeX-ohjelmaa” ei koodin kirjoittamiseen tarvita.
- Huomaa kuitenkin, että tekstinkäsittelyohjelmat kuten Word tallentavat yleensä tiedostot omissa tiedostomuodoissaan (esim. `.doc`), eivät tekstinä.
- LaTeX-kooditiedostot tallennetaan yleensä tiedostopäätteellä `.tex`.
- Vaikka muotoilematon teksti on muuten varsin universaali tiedostomuoto, on syytä panna merkille käytetty *merkistö*, sillä se on LaTeXille kerrottava. Mikäli et tiedä mikä merkistö on, älä huolestu – asiaan paneudutaan myöhemmin tarkemmin.

Linux, Windows & Mac

Vim, Emacs

Linux

- gedit
- Kate
- Nano/pico
- TEXstudio
- Texmaker

Windows

- TEXNicCenter
- Texmaker
- WinShell
- (Notepad eli Muistio)

Mac

- iTeXMac
- TeXShop
- Texmaker

- Tämän kurssin harjoituksissa esimerkkieditorina käytetään TEXNicCenteriä, joka on saatavissa Windows-koneille ilmaiseksi osoitteesta `http://www.texniccenter.org/`.
- TEXNicCenter on erityisesti LATEX-koodille tarkoitettu editori, tai tarkemmin sanottuna IDE eli *Integrated Development Environment*.
- TEXNicCenteristä löytyy monia toimintoja, jotka helpottavat LATEX-koodin käsittelyä, esimerkiksi syntaksinkorostus, rakennenäkymät, valmiit nappulat koodin kääntämiselle jne.
- TEXNicCenterin käyttöä käsitellään tarkemmin harjoituksissa.
- Linux-käyttäjät voivat kokonaisen LATEX-IDEn halutessaan käyttää esimerkiksi Texmakeria.

- Vaikka LaTeX-koodin kirjoittamiseen tai muokkaamiseen ei tarvita mitään erityistä ohjelmaa, kun koodista halutaan valmis dokumentti, on koneesta löydyttävä erityinen *kääntäjä*.
- Kääntäjä, joka tulkkaa LaTeX-koodin PDF-tiedostoksi, on nimeltään `pdflatex`.
- Tämän lisäksi on olemassa perinteinen kääntäjä nimeltään `latex`, joka tuottaa DVI-muodossa olevia tiedostoja. DVI on kuitenkin muinainen ja levityksen kannalta hankala formaatti, jota ei ole enää syytä käyttää.
  - Toinen vanhentunut tapa on kääntää koodi ensin DVI-tiedostoksi, sitten DVI:stä PDF:ksi. Tätäkään tapaa ei kannata enää käyttää.

- `pdflatex` ja `latex` ovat yksinkertaisia komentoriviohjelmia, joille annetaan käännettävän kooditiedoston nimi (`.tex`-päätteen voi jättää pois).
- Esimerkiksi jos käännettävä koodimme on tiedostossa `hello.tex`, komentamalla tiedoston sisältävässä hakemistossa `pdflatex hello.tex`, ilmaantuu samaan hakemistoon toivomamme `hello.pdf`.
- Vaikka komentorivi olisi outo ja pelottava käsite, ei kannata hätääntyä! TEXNicCenteristä ja muista IDEistä löytyy nappula, jota painamalla koodi syötetään `pdflatex`ille.
- Sekä komentoriviltä että nappuloiden kautta käytettynä `pdflatex` tulostaa kääntämisen yhteydessä suuren määrän kääntämisen etenemisestä kertovaa tekstiä. Tähän lokitietoon kannattaa perehtyä, erityisesti jos kääntämisessä menee jotain vikaan.

- LaTeXin levitys tapahtuu yleensä valmiina *distribuutioina*, joissa on paketoitu samaan nippuun erilaisia TeX- ja LaTeX-kääntäjiä, kääntäjien tarvitsemat sekalaiset apuohjelmat, sekä iso joukko valmiita dokumenttityylejä, paketteja, fontteja ym.
- Linux-maailmassa käytetään yleensä TeX Live -distribuutiota (`http://www.tug.org/texlive/`), joka on saatavilla myös Windowsille. Ikivanhoista Linuxeista saattaa löytyä vielä distribuutio nimeltä teTeX.
- Windows-maailmassa laajasti käytetty distribuutio on MiKTeX (`http://miktex.org/`), joka on asennettu myös fysiikan laitoksen ATK-luokan koneille. Siitä on myös laajennettu, helpommin asennettava versio proTeXt (`https://www.tug.org/protext/`).
- Mac OS X:lle on ainakin TeX Liveen pohjautuva MacTeX (`http://www.tug.org/mactex/`).

- Luennoilla tai demoissa ei käydä yksityiskohtaisesti läpi distribuution asentamista kotikoneelle. Apua saa ja kannattaa kuitenkin kysyä jos asennuksessa tulee ongelmia.
- Distribuutiot tarjoavat kattavat asennusohjeet, joihin pääsee käsiksi edellisen kalvon linkeistä.
- Myöskään editorien asentamista ei käydä luennoilla tarkemmin läpi. Erilaisiin editoreihin kannattaa kuitenkin ehdottomasti tutustua.

Älä tyydy muistioon!

Jokainen Notepadia monimutkaisemman editorin opetteluun käytetty tunti tulee kymmenkertaisena takaisin säästetyn työn muodossa.

Osa II

Leipätekstiä LaTeXilla

Aivan tavallinen teksti

Esipuhe

Kaiken kirjallisen hengentuotoksen ydinaineen muodostaa tietysti aivan tavallinen teksti. Näin ollen ennen matemaattisten kaavojen kimppuun käymistä on syytä opetella latomaan leipätekstiä.

Mikäli aikaisempaan esimerkkidokumenttiimme vaihtaa tervehdyksen hieman suomalaisemmaksi, huomaa pian, että tulosteessa on jotain vikana:

Suomalaistettu esimerkki

```
\documentclass[a4paper,12pt]{article}
\begin{document}
Päivää maailma!
\end{document}
```

Tuloste

Piv maailma!

Jokin on syönyt ääkköset! Ongelmana on aikaisemminkin mainittu kysymys tekstin *merkistöstä* – LaTeX käyttää oletuksena amerikkalaista `ASCII`-merkistöä, johon ääkköset eivät kuulu.

- Tietokone ymmärtää vain ykkösiä ja nollia. Sille on erikseen kerrottava, minkälainen bittijono tulkitaan miksikin kirjoitusmerkiksi. Tätä ”karttaa” biteistä merkkeihin kutsutaan *merkistöksi*.
- Aikaisemmin eri kielet ovat käyttäneet eri merkistöjä:
  - Muinaisten amerikkalaisten `ASCII` ei sisällä ääkkösiä eikä muitakaan epäamerikkalaisia merkkejä.
  - Länsieurooppalaisten `latin1` sisältää ääkköset, mutta ei esimerkiksi kyriilisiä aakkosia tai japanilaisia kanji-merkkejä.
- Nykyään on olemassa kaikki ihmiskunnan koskaan kuvittelemat merkit sisältävä Unicode-standardi, joka hiljalleen syrjäyttää kielikohtaisia merkistöjä. Vanhat merkistöt ovat kuitenkin vielä laajalti käytettyjä.

Merkistöongelma on helposti ratkaistu – lataamme vain paketin `inputenc`, joka ottaa valinnaisena argumenttinaan merkistön nimen. On vain tiedettävä millä merkistöllä teksti on kirjoitettu.

- Ensisijaisesti käytettyä merkistöä kannattaa lähteä selvittämään käyttämänsä tekstieditorin asetuksista.
- Vanhemmissa koneissa merkistönä on usein ISO 8859-1 eli `latin1` tai sen laajennos `latin9`.
- ☹ Windows-ympäristöissä edellisen tilalla on joskus Microsoftin oma, epästandardi hirviö Windows-1252 eli `ansinew`.
- Koska `latin9` ja `ansinew` eroavat `latin1`:stä vain harvemmin käytettyjen merkkien osalta, argumentti `latin1` yleensä toimii, vaikka merkistö olisikin jompi kumpi edellisistä.
- Ainakin Linux-ympäristöissä on jo oletuksena Unicoden `utf8`.
  - ☹ LaTeXin korkeasta iästä johtuen koko Unicoden repertuaarin käyttäminen LaTeXissa ei kuitenkaan ole aivan yksinkertaista. Parempi tuki löytyy uudemmista laajennoksista XeLaTeX ja LuaLaTeX.

Paketin `inputenc` lataaminen oikealla merkistöllä saa myös edellisen esimerkin ääkköset näkyviin:

Suomalaistettu esimerkki oikealla merkistöllä

```
\documentclass[a4paper,12pt]{article}
\usepackage[utf8]{inputenc}
\begin{document}
Päivää maailma!
\end{document}
```

Tuloste

Päivää maailma!

Ääkkösien käyttöön liittyy toinenkin huomionarvoinen seikka:

- LaTeX käyttää oletuksena sisäisenä kirjainvalikoimanaan varsin suppeaa `OT1`-fonttikodausta, jolloin se ”parsii” kokoon ä:n lisäämällä esimerkiksi kaksi pistettä a-kirjaimen päälle.
- Koska tällainen parsiminen saattaa aiheuttaa ongelmia esimerkiksi tavutuksen kannalta, kannattaa LaTeXiin ladata aina laajempi fonttikoodaus `T1`.

T1-fonttikoodauksen käyttöönotto

```
\usepackage[T1]{fontenc}
```

- Pelkkien merkistöjen lisäksi LaTeXin monikielisyystuki menee pidemmälle: LaTeX osaa esimerkiksi
  - Tavuttaa, kun se vain tietää mistä kielestä on kysymys.
  - Tuottaa sisällysluetteloita ja kansilehtiä käyttäjän haluamalla kielellä.
- Kaiken takana on osuvasti nimetty paketti `babel`, joka ottaa valinnaisina argumentteinaan dokumentissa käytetyt kielet. Dokumentin oletuskieli on kielilistan viimeinen kieli.
  - Uudemmissa babelin versioissa oletuskielen voi ilmaista myös lisäämällä tämän kielen eteen teksti `main=`.

babel-paketin lataaminen

```
\usepackage[english,finnish]{babel}
```

Kesken dokumentin kieltä voidaan vaihtaa:

- komennolla `\selectlanguage`, joka ottaa argumenttina kielen nimen.
- ympäristöllä `otherlanguage`, joka myös ottaa argumenttinaan käytetyn kielen.
- komennolla `\foreignlanguage`, joka ottaa ensimmäisenä argumenttinaan kielen ja toisena ladottavan tekstin.
  - Tämä komento ottaa käyttöön hetkellisesti vain kyseisen kielen tavutuksen ja mahdolliset typografiset erityispiirteet.

babel-esimerkki

```
Kun täällä on \today, on Vatikaanissa
\begin{otherlanguage}{latin}\today\end{otherlanguage}.
```

Tuloste

Kun täällä on 22. kesäkuuta 2015, on Vatikaanissa XXII Iunii MMXV .

Aina suomea kirjoittaessa kannattaa dokumentin alustuksessa ladata vähintään seuraavat paketit:

LaTeXin suomentavat rivit

```
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[finnish]{babel}
```

Korvaa kuitenkin edellisen `utf8`, mikäli käytät jotain toista merkistöä!

Kokeillaanpa nyt kirjoittaa jotain hieman pidempää proosaa.
Oletetaan, että alustuksessa on jo ladattu `inputencit` ja muut paketit.

Pidempi esimerkki

```
Maijalla oli karitsa,
karitsa, karitsa.
Maijalla   oli   karitsa,
lumivalkea.
```

Tuloste

Maijalla oli karitsa, karitsa, karitsa. Maijalla oli karitsa, lumivalkea.

Esimerkistä huomataan, että:

- Leipäteksti näkyy sellaisenaan ilman erityisiä temppuja.
- LaTeX jättää huomiotta koodissa olevan rivityksen ja mahdolliset ylimääräiset sanavälit. Koodia kirjoittaessa ei tarvitse siis huolehtia tasaisista marginaaleista, vaan voi jättää rivityksen LaTeXin tehtäväksi.

- Jos riviä *ei* saa vaihtaa jonkin sanavälin kohdalla, korvaa koodissa välilyönti aaltoviivalla (~).
  - Esimerkiksi lausahduksissa *Katso kuva 2* tai *Funktio vaihtaa merkkiään kohdassa* $x = 0$ näyttäisi varsin tökeröltä, jos rivi vaihtuisi sanojen *kuva* tai *kohdassa* jälkeen.
  - Koska aaltoviivalla on oma merkityksensä LaTeXissa, jos haluat tuottaa tulosteeseen aaltoviivan, käytä komentoa `\textasciitilde`.
- Jos taas haluat LaTeXin vaihtavan riviä jossain kohdassa, käytä komentoa `\\`.
  - *Tätä komentoa kannattaa käyttää harkiten*. Jos LaTeX näyttää rivittävän huonosti jossain kohdassa, on usein syypäänä jokin sana, jota LaTeX ei osaa tavuttaa. Tämänkaltaisten manuaalisten ohituskomentojen käyttö saattaa aiheuttaa ongelmia myöhemmin, kun esimerkiksi lisää tekstiä ennen rivityskomentoa, mutta LaTeX edelleen nöyränä vaihtaa riviä `\\`-komennon kohdalla.

- Kappaleen vaihtuminen ilmaistaan jättämällä koodiin *tyhjä rivi*.
- Kappaleen vaihtuminen näkyy tekstissä yleensä pystysuuntaisena välinä ja/tai uuden kappaleen ensimmäisen rivin sisennyksenä, dokumenttityylistä riippuen.
- Jos dokumenttityylin oletusarvo ei miellytä, voi kappalejaon esitystä säätää mieleisekseen, josta myöhemmin lisää.
- Kappaleen ensimmäisen rivin sisennyksen voi estää myös käyttämällä kappaleen alussa komentoa `\noindent`.

Esimerkki kappalejaosta

```
Tässä on ensimmäinen tekstikappale.
                                    ←—Tyhjä rivi, eli kappale vaihtuu.
Tästä alkaa toinen tekstikappale.
```

- LaTeX on varsin pätevä tavuttamaan sanoja, kunhan sille on vain `babel`-paketin komennoilla kerrottu, mitä kieltä kirjoitetaan.
- LaTeX pyrkii tasaamaan tekstin molemmista reunoista, tavuttaen sanoja jos se on tarpeen.
- Koska kaikkien sanojen tavutus ei ole säännönmukaista, LaTeXille on joskus kerrottava, kuinka sana tavutetaan. Tällaisia sanoja ovat esimerkiksi:
  - Yhdyssanat, joissa sanajako muuttaa tavutusta (esim. *piknikevät*, *syysaamu*).
  - Yhdyssanat, joiden sanajako ei ole yksikäsitteinen (esim. *kaivosaukko*, *puunaulakatu*).
- Sanan tavutuksen voi kertoa LaTeXille lisäämällä tavujen väliin komennon `\-`, eli esim. `pik\-nik\-e\-väät`.

- Koska varsinkin fysiikan kapulakielessä poikkeuksellisen tavutuksen omaavia sanoja saattaa tekstissä esiintyä paljon, kannattaa toistuvien sanojen tavutus ilmaista komennolla `\hyphenation`, jolloin tavuviivoja ei tarvitse lisätä käsin jokaiseen kohtaan, jossa sana esiintyy.
- Alustukseen laitettava komento `\hyphenation` ottaa argumentikseen välilyönneillä erotetun listan sanoja, joissa jokainen tavutuskohta on ilmaistu eksplisiittisesti lisäämällä tavuviiva eli - tavujen väliin.
- Huomaa, että tällä tavoin voidaan myös estää joidenkin sanojen (esim. nimien) tavutus.

Esimerkki \hyphenation-komennon käytöstä

```
\hyphenation{FORTRAN Albert Einstein pik-nik-e-väät syys-aa-mu}
```

Varo!

Joissain distribuutioissa (vanhemmat MiKTEXin ja teTEXin versiot) tuki suomen tavutukselle on oletuksena poistettu käytöstä. Tämän seurauksena suomenkielinen tavutus ei toimi vaikka `babel`-paketti olisi asianmukaisesti otettu käyttöön!

- Tällöin LaTeX kyllä varoittaa kääntäessä, että tavutusohjeita kielelle suomi ei löydy, mutta tämä varoitus jää käyttäjältä helposti huomaamatta.
- ☹ Tästä ongelmasta on lähtenyt liikkeelle käsitys, jonka mukaan LaTeX ei osaa tavuttaa suomea oikein.

- Käynnistä ohjelma `texconfig`.
- Valitse Hyphenation.
- Valitse latex.
- Tarkista, että suomen kielen tuen lataava rivi ei ole merkitty ohitettavaksi kommenttimerkillä %. Rivin pitäisi näyttää tältä:

Suomen kielen tuen lataava rivi `texconfig`-ohjelmassa

```
finnish fi8hyph.tex
```

- Joissain distribuutioissa saattaa edellisen tilalla olla `fihyph.tex`, mutta `fi8hyph.tex` on näistä kahdesta uudempi ja parempi, ja sitä kannattaa käyttää mikäli se löytyy.

- Etsi Windowsin Käynnistä-valikosta käsiisi kansio MiKTeX ja sieltä MiKTeX Options tai Settings.
- Valitse ylhäältä välilehti Language.
- Tarkista, että kohdassa finnish on ruksi.

Tyylit, välit ja välimerkit

Esipuhe

Hyvä ja helppolukuinen teksti on paitsi hyvin kirjoitettua, myös hyvin ladottua. Seuraavassa kappaleessa tutustumme hieman tekstin ulkoasun muokkaamiseen yleisellä tasolla, sekä tekstissä esiintyviin erilaisiin väleihin ja välimerkkeihin.

Tärkein tekstin yleiseen ulkoasuun vaikuttava tekijä on käytetty dokumenttityyli, eli komennolle `\documentclass` annettu argumentti.

- Tieteellisiin artikkeleihin tarkoitettu `article` sopii hyvin laboratorioselostusten, erikoistöiden, gradujen ja muiden tyyliltään artikkelimaisiin dokumentteihin.
- `article` noudattaa kuitenkin amerikkalaista typografiaperinnettä, joten se saattaa näyttää eurooppalaiseen silmään hieman pröystäilevältä (jos on typografi). Alankomaalaiset `artikel1` ja `artikel3` tarjoavat eurooppalaisemman vastineen.
  - Edellisten ainoa ero on se, että `artikel1` erottaa oletuksena kappaleet sisennyksellä, ja `artikel3` puolestaan välillä.
  - ☹ Alankomaalaisten tyyleillä on valitettavasti yhteensopimattomuusongelmia joidenkin pakettien kanssa.

Muunlaisiin dokumentteihin löytyy usein oma tyylinsä, ja usein onkin parempi löytää hyvä dokumenttityyli, kuin alkaa ohittamaan esimerkiksi `article`-luokan määrityksiä joka kohdassa.

- Kirjojen (esim. väitöskirja) latomiseen löytyy `book`, sen eurooppalaistukset `boek1` ja `boek3`, sekä uudempi `memoir`.
- Kirjeisiin löytyy LaTeXin vakioluokka `letter`, mutta uudempi `newlfm` on parempi.
- Tämänkin diaesityksen tuottanut `beamer`-luokka esitellään myöhemmin.
- Valitettavasti `selkkari`-tyyliä ei ole kukaan vielä kirjoittanut, joten selkkareihin `article` pienillä muutoksilla on edelleen paras vaihtoehto.
- Muihin opinnäytteisiin, esim. kandidaatintöihin, löytyy itsensä Aku Jokisen kirjoittama `jyfl thesis`: (`http://users.jyu.fi/~akjujoki/fi/projektit/jyflthesis`)

Tämän kurssin puitteissa ei käsitellä `beameria` lukuun ottamatta ei-artikkelimuotoisen tekstin dokumenttityylejä.

- Joitain dokumentin ulkoasuun liittyviä asetuksia säädellään komennolle `\documentclass` annettavilla valinnaisilla argumenteilla. Tärkeimmät näistä ovat paperi- ja fonttikoko.
- Dokumenttityyli määrää, mitä paperi- ja fonttikokoja on tarjolla.
- Ainakin `article`-tyyliin eurooppalaistuksineen A4-kokoisen paperin saa argumentilla `a4paper`, A5-kokoisen argumentilla `a5paper`, amerikkalaisten käyttämän letter-paperikoon argumentilla `letterpaper`, jne.
- Artikkelityylit tarjoavat kolme fonttikoon valitsevaa argumenttia: `10pt`, `11pt` ja `12pt`. Lisää kokoja saa käyttöönsä paketilla `extsizes`.

Esimerkki paperi- ja fonttikoon määrittämisestä

```
\documentclass[a4paper,12pt]{artikel3}
```

- Kuten mainittua, osa dokumenttityyleistä erottaa kappaleet oletuksena sisennyksellä, ja osa jättämällä pienen tyhjän tilan kappaleiden väliin.
- Mikäli tähän haluaa vaikuttaa, voi uudelleenmääritellä pituudet `\parskip` ja `\parindent`, joista edellinen antaa kappaleiden välisen tyhjän tilan, ja jälkimmäinen sisennyksen pituuden.
- Uudellenmäärittely tehdään LaTeXin komennolla `\setlength`, jota käytetään myös monien muiden pituuksien säätelemiseen.
- `\setlength` ottaa ensimmäisenä argumenttinaan säädettävän pituuden, ja toisena pituusmääreen joissain LaTeXin ymmärtämissä yksiköissä, eli esimerkiksi millimetreissä (mm), pisteissä (pt) tai fonttikoon yksiköissä (em).

Esimerkki kappalevälin uudelleenasetuksesta

```
\setlength{\parindent}{0pt}
\setlength{\parskip}{5mm}
```

- Kokemattoman silmään dokumenttityylien oletusmarginaalit saattavat näyttää leveiltä. Leveät marginaalit kuitenkin parantavat luettavuutta merkittävästi.
- Mikäli marginaalien kokoa haluaa muuttaa, voi sen tehdä määrittelemällä pituuksia uudelleen kuten edellä, mutta helpoiten se käy käyttämällä pakettia `geometry`.
- Paketin valinnainen argumentti `hmargin={X,Y}` asettaa sisämarginaaliksi `X` ja ulkomarginaaliksi `Y` (yksipuolisessa tulostuksessa siis vasen ja oikea). Vastaavan ylä- ja alamarginaaleille tekee `vmargin={X,Y}`.
- Huomaa, että ylä- ja alamarginaalit eivät tarkoita etäisyyttä paperin reunasta tekstiin, sillä tekstin ylä- ja alapuolella tilaa vievät lisäksi ylä- ja alaviite. Näistä pääset eroon argumenteilla `nohead` ja `nofoot`.

Esimerkki `geometry`-paketista

```
\usepackage[hmargin={35mm,25mm},vmargin={20mm,25mm}]{geometry}
```

- LaTeX käyttää oletuksena normaalia julkaistavan tekstin riviväliä, eli riviväliä 1.
- Mikäli tulosteeseen on tarkoitus kirjoittaa käsin huomautuksia, voi olla perusteltua käyttää suurempaa riviväliä.
- Rivivälin puolitoista saa käyttöön lataamalla paketin `setspace` ja käyttämällä komentoa `\onehalfspacing`. Rivivälin kaksi saa vastaavasti komennolla `\doublespacing`. Normaaliin riviväliin voi palata komennolla `\singlespacing`
- Riviväliä voi myös hienosäätää tarkemmin, josta lisää luennolla 6.

- Artikkelityylit lisäävät oletuksena sivunumeron sivun alalaitaan.
- Sivunumerointia ja yleistä ylä- ja alaviitteiden ulkoasua säätelee komento `\pagestyle`.
  - `\pagestyle{plain}` tuottaa pelkät sivunumerot.
  - `\pagestyle{empty}` tuottaa tyhjät ylä- ja alaviitteet.
  - `\pagestyle{headings}` tuottaa dokumenttityylin määräämät ylä- ja alaviitteet.
- Komento `\thispagestyle` toimii kuten `\pagestyle`, mutta vaikutus ulottuu vain senhetkiseen sivuun.
- Käytännössä monimutkaisemmat viitekonstruktiot kannattaa tehdä paketin `fancyhdr` avulla. Tätä pakettia ei tässä vaiheessa käsitellä tarkemmin.

- Mikäli dokumenttinsa aikoo tulostaa kaksipuoleisesti, kannattaa dokumenttityylille antaa argumentti `twoside`. Tällöin LaTeX esimerkiksi säilyttää yhtenäiset marginaalit tekstin sisä- ja ulkoreunoilla.
- Tekstin voi jakaa kahteen palstaan antamalla dokumenttityylille argumentin `twocolumn`. Monimutkaisempaan palstoittamiseen löytyy paketti `multicol`.

- Kun dokumentin yleisestä ulkoasusta siirrytään tekstin ulkoasuun, ensimmäisenä vaikuttavana asiana on luonnollisesti käytetty fontti eli kirjasin.
- Myös fontit otetaan LaTeXissa yleensä käyttöön tutulla `\usepackage`-komennolla. Yhdessä fonttipaketissa tulee tyypillisesti mukana myös saman fontin muunnelmat kuten lihavoitu ja kursivoitu versio, joten yleensä ei ole tarvetta ladata kuin yksi fonttipaketti.

- LaTeX käyttää oletuksena Knuthin suunnittelemaa fonttia *Computer Modern*. Tätä fonttia käyttävä dokumentti on helppo tunnistaa LaTeXilla tehdyksi, mutta joidenkin silmiin *CM* on turhan laiha.
- *Palatino* on hyvä perusfontti, ja sen saa käyttöön esimerkiksi lataamalla paketin `palatino`. Esimerkiksi tämä dokumentti käyttää *Palatinon* sukuista fonttia paketista `newpxtext`.
- Kaijanahon kirja käyttää *Concrete*-fonttia, joka on myöskin varsin selkeä. Tämän saat käyttöön paketista `ccfonts`.
- Kelvollinen, mutta pahasti käytössä kulunut *Times* löytyy paketista `times`.
- *Charter* on myös mukava, mainitsemisen arvoinen fontti.
- Lisää vaihtoehtoja löytyy esimerkiksi osoitteesta `http://www.tug.dk/FontCatalogue/`.

- Mikäli jotain osaa tekstissä halutaan *korostaa*, kannattaa käyttää komentoa `\emph`, joka ottaa pakolliseksi argumentikseen korostettavan tekstin.
- Tyypillisesti korostettava teksti ladotaan *kursiivilla*, mutta esimerkiksi jo korostetun tekstin sisällä `\emph`-komennolla korostaminen vaihtaa hetkeksi normaaliin kirjasintyyppiin.

Esimerkki `\emph`-komennosta

```
Rakastan \LaTeX ia \emph{valtavasti!}
```

Muistutus

Käytä korostusta säästeliäästi – se kuluu käytössä!

- Tavallisimmat välimerkit, eli pilkun, pisteen, huutomerkin ja kysymysmerkin LaTeX latoo sellaisenaan.
- Englannin kielessä virkkeiden välissä käytetään suurempia välejä kuin saman virkkeen sanojen välissä. LaTeX olettaa, että virke päättyy pisteeseen, huutomerkkiin tai kysymysmerkkiin, jota seuraa väli ja jota ei edellä iso kirjain.
  - Jos ei haluta vaihtaa virkettä tällaisessa kohdassa, pakotetaan normaali sanaväli komennolla `\␣`.
  - Jos taas halutaan lopettaa virke ison kirjaimen jälkeen, kirjoitetaan komento `\@` heti ison kirjaimen perään.
  - Huomaa, että nämä poikkeukset eivät koske suomen kieltä, jossa sanaväli on sama myös virkkeiden välissä.
- Joissain tapauksissa (kuten tuhaterottimena) on tapana käyttää normaalia lyhyempää sanaväliä. Tämä onnistuu komennolla `\,` (kenoviiva pilkku).

- Näppäimistössä oleva - tuottaa LaTeXissa yhdysviivan.
- Kaksi peräkkäistä väliviivaa eli -- tuottaa ajatusviivana ja numeroiden välissä käytetyn n-viivan (engl. *en dash*).
- Kolme väliviivaa tuottaa vielä pidemmän m-viivan (engl. *em dash*), jota käytetään myös joskus ajatusviivana.
- Huomaa, että mikään näistä ei ole miinusmerkki! Siihen palataan myöhemmin matemaattisen materiaalin yhteydessä.

Esimerkki viivoista

```
vara-avain
4--6 henkilöä
Bosen--Einsteinin kondensaatti
```

Tuloste

vara-avain
4–6 henkilöä
Bosen–Einsteinin kondensaatti

Kuten sanottua, LaTeX osaa tavuttaa sanat itse, mutta…

- Yhdysviivan jo sisältävien sanojen tavuttamista neuvotaan usein välttämään, ja siksi LaTeX ei oletuksena tavuta yhdysviivan sisältävää sanaa.
- Suomen kielessä usein käytettyjen pitkien yhdyssanojen vuoksi tällaista joutuu joskus kuitenkin tekemään, jolloin tarvitaan seuraavia komentoja:
  - Muualla sanassa tavutuksen salliva yhdysviiva saadaan komennolla "-.
  - Tavutuksen salliva, sanan aloittava yhdysviiva saadaan komennolla "=.

Esimerkkejä tavutuksen sallivien yhdysviivojen käytöstä

```
maa"-artisokkagratiini
luomutomaatit ja "=kukkakaalit
```

- Englannin kielessä käytettävät vasemmat lainausmerkit saadaan kirjoittamalla kaksi takahipsua peräkkäin, ja oikeat kirjoittamalla kaksi hipsua peräkkäin, eli ‘‘ ja ’’.
- Suomen kielessä käytetään molemmilla puolilla oikeanpuoleisia hipsuja, eli ’’-hipsuja.

Esimerkkejä lainausmerkeistä

```
Suomessa käytetään
’’lainausmerkkejä’’.
In English we use ‘‘quotation
marks’’.
```

Tuloste

Suomessa käytetään
”lainausmerkkejä”.
In English we use “quotation
marks”.

Varoitus

Näppäimistöstä löytyvä tuuman merkki " ei ole lainausmerkki missään kielessä!

- Kolmen pisteen tuottamiseen käytä komentoa `\ldots`.
- Koska monilla erikoismerkeillä on oma merkityksensä LaTeX-koodissa, niiden tuottamiseen on usein oma komentonsa. Tällaisia ovat esimerkiksi %, &, $, #, { ja }, joista kukin saadaan tuotettua lisäämällä kenoviiva merkin eteen, eli esim. komento \& tuottaa &-merkin.
- Kenoviivan saat komennolla `\textbackslash`.
- Erittäin perusteellinen kokoelma kaikista LaTeXista löytyvistä erikoismerkeistä on koottu teokseen *The Comprehensive LaTeX Symbol List*, joka löytyy osoitteesta `http://www.ctan.org/tex-archive/info/symbols/comprehensive/symbols-a4.pdf`

- Esimerkiksi lainasanoja ja erikoisempia nimiä kirjoittaessa tulee usein tarve kirjaimille, joita ei omasta näppäimistöstä (tai edes käytetystä merkistöstä) löydy. Tällöin on kätevä käyttää aksentointikomentoja, jotka lisäävät hipsuja ja pisteitä olemassaolevien merkkien päälle tai alle.
- Kattava lista aksentointikomennoista ja löytyy esimerkiksi Kaijanahon kirjasta sivulta 47 tai edellisen kalvon linkin merkkilistasta sivulta 14.

Esimerkkejä eksoottisista kirjaimista

```
T\=oky\=o
na\"{\i}ve
\TH\'orr
H\‘an Th\'{\^e} Th\‘anh
```

Tuloste

Tōkyō
naïve
Þórr
Hàn Thế Thành

Manuaalinen muotoilu ja sijoittelu

Esipuhe

Joissain valikoiduissa tilanteissa (esim. kansilehden sommittelu) voi olla tarpeen määrätä LaTeX koristelemaan tai sijoittelemaan tekstiä juuri tietyllä tavalla tai vaihtamaan sivua tietyssä kohdassa.
Käytä näitä komentoja harkiten! Manuaalisia fonttityylin- tai fonttikoonvaihtokomentoja ja tietyn mittaisia tyhjiä välejä joutuu viljelemään koodin sekaan vain jos tekee jotain väärin. On LaTeXin filosofian mukaista mieluummin määritellä koodissa *mikä* jokin tekstipätkä on, ja dokumenttityylissä (tai alustuksessa) määritellä – vain kerran – *miten* se ladotaan.

- Kirjasimen ulkoasua voidaan muuttaa seuraavilla komennoilla. Kukin komento ottaa argumentikseen korostettavan tekstin.
  - **lihavoida** komennolla `\textbf`.
  - *kursivoida* komennolla `\textit`.
  - *kallistaa* komennolla `\textsl`.
  - kirjoittaa pääteviivattomalla fontilla komennolla `\textsf`.
  - `kirjoittaa vakiolevyisellä fontilla` komennolla `\texttt`.
  - KIRJOITTAA PIKKUTIKUILLA komennolla `\textsc`.
- Komentojen nimet tulevat termeistä *boldface*, *italic*, *slanted*, *sans serif*, *teletype* ja *small capitals*.
- Komentoja voidaan myös yhdistellä, mikäli halutaan esimerkiksi ***lihavoitua kursiivia***. Mahdollisten yhdistelmien määrä riippuu käytetystä fontista.

- Edellisistä fontinvaihtokomennoista on myös versiot, jotka eivät ota argumentteja, vaan muuttavat kirjasimen ulkoasun komennosta aina ympäristön tai aaltosuluilla erotetun koodinpalan loppuun asti. Nämä ovat:
  - `\bfseries` **lihavointiin**
  - `\itshape` *kursivointiin*
  - `\slshape` *kallistukseen*
  - `\sffamily` päätteettömään fonttiin
  - `\ttfamily` `vakiolevyiseen fonttiin`
  - `\scshape` PIKKUTIKKUIHIN
- Esimerkki: `{\bfseries\itshape lihavoitua kursiivia}`.
- Mikäli on tottunut käyttämään edellisen kalvon komentoja, näille on harvemmin tarvetta. Ne kannattaa kuitenkin tunnistaa, sillä jotkut niitä käyttävät.
- Lisää LaTeXin fonttien sielunelämästä voi lukea esimerkiksi osoitteesta `http://www.cl.cam.ac.uk/~rf10/pstex/latexcommands.htm`.

- Kirjasimen kokoa ei yleensä tarvitse vaihtaa kesken dokumentin, sillä LaTeX osaa itse latoa esimerkiksi otsikot isommalla ja alahuomautukset pienemmällä fontilla.
- Mikäli kuitenkin jostain syystä tulee tarve vaihtaa fonttikokoa, se onnistuu seuraavilla komennoilla:
  `\tiny`, `\scriptsize`, `\footnotesize`, `\small`, `\normalsize`, `\large`, `\Large`, `\LARGE`, `\huge`, `\Huge`
- Koonvaihtokomennot eivät ota argumentteja, vaan vaihtavat tekstin koon komennosta alkaen. Mikäli fonttikoko halutaan vaihtaa takaisin, voidaan käyttää komentoa `\normalsize`, tai rajata komennon vaikutusalue näin: `{\small pientä tekstiä}`.

Muistutus

Ethän käytä fonttikoon vaihtamista korostuskeinona. Se näyttää erittäin typerältä.

- Leipätekstissä teksti tasataan tyypillisesti molempiin reunoihin. Esim. kansilehdessä on usein kuitenkin toivottavaa esimerkiksi keskittää otsikko tai tasata tekijän nimen sisältävä tekstipätkä oikeaan reunaan.
- Keskittäminen ja tasaus onnistuu seuraavilla ympäristöillä:
  - `center` keskittää sisältönsä vaakasuunnassa.
  - `flushright` tasaa sisältönsä vain oikeaan reunaan.
  - `flushleft` tasaa sisältönsä vain vasempaan reunaan.
- Yo. ympäristöjen sisältö tulkitaan erilliseksi tekstikappaleeksi, eli se esimerkiksi erotetaan muusta sisällöstä pystysuuntaisella välillä.
  - Jos tämä ei käy laatuun, voi käyttää niiden vastineena komentoja `\centering`, `\raggedleft` ja `\raggedright`.

- Komento `\newpage` vaihtaa sivua. Komento `\clearpage` tekee samoin, mutta lisäksi varmistaa, että kaikki tähän mennessä määritelty materiaali on jo ladottu – LaTeX laitetaan usein esimerkiksi sijoittelemaan kuvat itse, jolloin koodissa jo vastaan tullutta kuvaa ei olla vielä sivunvaihdon kohdalla välttämättä ladottu.
- Kaksipuoleisessa tulostuksessa (eli `twopage` päällä) komento `\cleardoublepage` toimii kuten `\clearpage`, mutta varmistaa, että seuraava sivu on sivunumeroltaan pariton (eli kirjan aukeamalla oikealla puolella), eli komento vaihtaa tarvittaessa sivua kaksi kertaa peräkkäin.

Seuraavat sijoittelukomennot ovat erityisesti hyödyllisiä kansilehden latomisessa, mutta niille saattaa löytyä käyttöä muuallakin.

- Komennot `\hspace` ja `\vspace` tekevät vaaka- ja pystysuuntaista tyhjää tilaa. Molemmat ottavat yhden pakollisen argumentin, joka kertoo tyhjän tilan pituuden joissain LaTeXin ymmärtämissä yksiköissä. Esimerkiksi yhden senttimetrin vaakasuuntaisen välin luo `\hspace{1cm}`.
- Eräs LaTeXin ymmärtämä pituus on `\fill`, joka venyy niin pitkäksi kuin sivulle mahtuu. LaTeX tarjoaa komennot `\hfill` ja `\vfill`, jotka tarkoittavat samaa kuin `\hspace{\fill}` ja `\vspace{\fill}` vastaavasti.
- Pituusmääre voi myös olla negatiivinen!
- Oletuksena LaTeX poistaa manuaalisesti luodut välit, joiden toinen reuna osuu sivun reunoihin. Tämän voi estää käyttämällä komentojen `\hspace` ja `\vspace` ”tähdellisiä versioita” eli komentoja `\hspace*` ja `\vspace*`.

Esimerkki

```
Tähän\hspace{1em}horisontaalista väliä.\\
\vspace{1em}
Yläpuoliseen riviin vertikaalista väliä.\\
Negatiivinen tila\hspace{-2em}on hassua.\\
Fill\hfill paisuu\hfill kuin\hfill ilmapallo.
```

Tuloste

Tähän horisontaalista väliä.

Yläpuoliseen riviin vertikaalista väliä.
Negatiivinen tilaon hassua.
Fill paisuu kuin ilmapallo.

Listat, lainaukset ja huomautukset

Esipuhe

Leipätekstin lisäksi tavallisessa tekstissä käytetään esimerkiksi listoja ja lainauksia, joiden latomiseen löytyy omat ympäristönsä ja komentonsa. Näitä tulee harvoin käytettyä selkkareita kirjoittaessa, mutta muussa materiaalissa kylläkin.

Luetteloiden tekeminen LaTeXilla on suoraviivaista – Ympäristö `itemize` määrittelee numeroimattoman luettelon, ja tämän ympäristön sisällä komento `\item` aloittaa uuden luetteloalkion. Komento `\item` ottaa valinnaisen argumentin, joka määrää alkion eteen ladottavan symbolin.

Esimerkki luettelosta

```
\begin{itemize}
  \item Vompatti
  \item Tursas
  \item[\textleaf] Koppelo
\end{itemize}
```

Tuloste

- Vompatti
- Tursas
- Koppelo

Luetteloalkion eteen ladottavan oletussymbolin määrää dokumenttityyli.

Mikäli listan alkiot halutaan numeroida, voidaan ympäristö `itemize` korvata numeroinnin itse tekevällä ympäristöllä `enumerate`.

Esimerkki numeroidusta listasta

```
\begin{enumerate}
  \item Vompatti
  \item Tursas
  \item Koppelo
\end{enumerate}
```

Tuloste

1. Vompatti
2. Tursas
3. Koppelo

Tässäkin tapauksessa numeroinnin ulkoasun määrää dokumenttityyli. Artikkelityyleissä numerointi on tyypillisesti vain numero ja piste.

Termien määrittelyjen ja muiden kuvailevien listojen esittämistä varten LaTeX tarjoaa ympäristön `description`. Tämä ympäristö toimii kuten `itemize`, mutta `\item`-komennon valinnainen argumentti tulkitaan määriteltäväksi termiksi.

Esimerkki kuvailevasta listasta

```
\begin{description}
  \item[Vompatti] Pussieläin
  \item[Tursas] Nilviäinen
  \item[Koppelo] Naarasmetso
\end{description}
```

Tuloste

Vompatti Pussieläin

Tursas Nilviäinen

Koppelo Naarasmetso

- Usein on tarvetta numeroiduille listoille, joissa pelkän numeron tilalla on esimerkiksi roomalaisia numeroita, numeroita sisältäviä ilmauksia, ym. Tällaiset listat on kätevä tehdä pakettien `enumerate` tai `enumitem` avulla.
- Paketti `enumerate` määrittelee uudelleen ympäristön `enumerate` siten, että se ottaa valinnaisen argumentin.
- Ympäristölle annettu argumentti ladotaan jokaisen luetteloalkion eteen siten, että argumentissa mahdollisesti olevat kirjaimet `a`, `A`, `i`, `I` ja numero `1` korvataan juoksevalla numeroinnilla käyttäen aakkosia, roomalaisia numeroita tai arabialaisia numeroita.
- Mikäli jokin näistä erikoiskirjaimista halutaan säilyttää sellaisenaan, tulee se ympäröidä aaltosuluilla.
- ☹ Paketilla `enumerate` on ikäviä yhteensopimattomuusongelmia esim. `artikel`-tyylien kanssa.
- Uudempi paketti `enumitem` on monipuolisempi, ja sen avulla voi loihtia jos jonkinlaisia listauksia.

Esimerkki paketista enumerate

Esimerkki

```
\begin{enumerate}[Es{i}merkk{i} A]
  \item Hömötiainen
  \item Rusakko
\end{enumerate}

\begin{enumerate}[Elä{i}n I]
  \item Vompatti
  \item Mufloni
\end{enumerate}

\begin{enumerate}[\itshape (i)]
  \item Koppelo
  \item Tursas
\end{enumerate}
```

Tuloste

Esimerkki A Hömötiainen
Esimerkki B Rusakko

Eläin I Vompatti
Eläin II Mufloni

(i) Koppelo
(ii) Tursas

- Alahuomautusten teko LaTeXilla on harvinaisen helppoa. Huomautuksen kohdalle laitetaan vain komento `\footnote`, joka ottaa argumentikseen huomautustekstin[1].
- Alahuomautuksia kannattaa käyttää säästeliäästi[2], sillä ne keskeyttävät lukemisen ikävästi[3].

[1] Tältä se näyttää.
[2] Ja tarkkaan harkiten.
[3] Varsinkin jos joka välissä joutuu vilkaisemaan sivun alalaitaan.

Joskus esimerkiksi ohjelmakoodia ladottaessa olisi toivottavaa latoa teksti aivan sellaisenaan, ilman että LaTeX tulkitsee niissä olevia erikoismerkkejä ja komentoja. Tähän auttaa komento `\verb` sekä `verbatim`-ympäristö.

- Komento `\verb` on syntaksiltaan hieman erikoinen: Välittömästi komennon jälkeen annetaan jokin erikoismerkki, joka ei ole *. LaTeX latoo tekstin sellaisenaan seuraavaan tämän merkin esiintymiskohtaan asti.
- Komento `\verb` on tarkoitettu vain lyhyille ilmauksille, eikä se esimerkiksi suvaitse rivinvaihtoja. Pidemmille ilmauksille kannattaa käyttää `verbatim`-ympäristöä, joka vastaavalla tavalla latoo ympäristön sisällä olevan tekstin sellaisenaan.
- Molemmat komennot latovat tekstin vakiolevyisellä fontilla.

Esimerkki \verb-komennosta ja verbatim-ympäristöstä

Esimerkki

```
\LaTeX-logon saa komennolla \verb=\LaTeX=.\\
Koodissa oli siis \verb&\verb=\LaTeX=&.
\begin{verbatim}
Erikoismerkkejä \{}#$%&_^

\end{verbatim}
```

Tuloste

LaTeX-logon saa komennolla `\LaTeX`.
Koodissa oli siis `\verb=\LaTeX=`.

```
Erikoismerkkejä \{}#$%&_^
```

Aikaisemmassa esimerkissäkin vilahtanut quote-ympäristö on tarkoitettu lainausten latomiseen. Alla oleva esimerkki kertonee kaiken oleellisen.

Esimerkki

```
\begin{quote}
Understanding is a three-edged
sword.\\
--- Ambassador Kosh
\end{quote}
```

Tuloste

> *Understanding is a*
> *three-edged sword.*
> *— Ambassador Kosh*

Osa III

Rakennetta dokumenttiin

Osat ja otsikot

Esipuhe

Dokumentin luettavuuden ja ymmärrettävyyden kannalta oleellista on tekstin jako loogisiin osiin. Kukaan ei lue kymmeniä sivuja yhteen pötköön kirjoitettua leipätekstiä, joten tekstiin on syytä tuoda *rakennetta* lisäämällä otsikoita, väliotsikoita, ja ehkä myös houkutteleva kansilehti. Tässä kappaleessa pääsemme myös lopulta kokeilemaan LaTeX-taitojamme käytännössä pienen selkkariesimerkin myötä.

- Kuten moni varmasti jo arvaa, LaTeXissa tekstin jakamista osiin ja osien otsikointia ei tehdä käsin nyhräämällä lihavoituja rivejä tekstin sekaan – kirjoittaja kertoo LaTeXille missä vaiheessa halutaan aloittaa uusi luku tai alaluku, ja LaTeX hoitaa otsikoiden latomisen ja numeroinnin.
- Otsikoinnin ulkoasun määrää dokumenttityyli, joten otsikointityylin vaihtaminen esim. artikkelimuotoisesta kirjamuotoiseen käy käden käänteessä.
- Tämän seurauksena kone myös tietää joka vaiheessa, mihin dokumentin osaan kirjoitettu teksti kuuluu. Tästä on suuresti hyötyä sisäisten viittausten (esim. ”katso kappale 3”) automatisoinnissa, mistä myöhemmin lisää.

- LaTeXin yleisimmät dokumenttityylit tarjoavat seuraavat tekstin osittelukomennot, joista kukin ottaa argumentikseen käytettävän otsikon.

1. `\part`
2. `\chapter`
3. `\section`
4. `\subsection`
5. `\subsubsection`
6. `\paragraph`
7. `\subparagraph`

Esimerkki osittelukomennoista

```
\part{Sormuksen ritarit}
\part{Kaksi tornia}

\section{Johdanto}
\section{Teoreettiset lähtökohdat}
\section{Mittauslaitteisto}
```

- Kaksi ylimmän tason osittelukomentoa löytyvät vain laajempien tekstien dokumenttiluokista, kuten `book`.
- Komennot ottavat myös valinnaisen argumentin, jonka avulla voi tarjota lyhyemmän, vaihtoehtoisen otsikon esim. sisällysluetteloa varten.

- `paragraph`- ja `subparagraph`-tason otsikoinnille ei yleensä ole tarvetta.
- Myös `subsubsection`-tason jaottelu on usein liiallista. Esimerkiksi selkkarissa `section` ja `subsection` yleensä riittävät.
- Kustakin osittelukomennosta on myös vaihtoehtoinen versio, joka saadaan käyttöön lisäämällä asteriski eli * komennon perään ennen argumentteja (ns. *tähdellinen versio*). Nämä osittelukomennot eivät ota tekstin osaa huomioon numeroinnissa.
- Numerottomia osia selkkareissa ovat esimerkiksi *Viitteet* sekä *Liitteet*.

Esimerkki osittelukomennoista

```
\section{Johtopäätökset}
\section*{Viitteet} % Tätä ei siis numeroida
```

- LaTeXin dokumenttityylit sisältävät myös valmiit dokumenttipohjat kansilehden ladontaan.
- Dokumenttityylin kansilehtipohjan saa ladottua komennolla `\maketitle`, kunhan ensin on kertonut LaTeXille kansilehdessä käytetyt tiedot kuten dokumentin otsikon ja tekijän, osuvasti nimetyillä komennoilla `\title` ja `\author`.
- Esimerkiksi Kaijanahon kirjassa sivuilla 59–61 on näytetty esimerkit kansilehdistä eri dokumenttityyleillä.
- Valmiit dokumenttityylit eivät kuitenkaan (vielä) tee esimerkiksi selkkariohjesäännön mukaista kansilehteä, ja siksi kansilehden voi joutua tekemään itse. Myöhemmässä esimerkissä käymme läpi kuinka tämä tapahtuu. Tämä on se paikka, jossa manuaalisille sijoittelukomennoille tulee käyttöä.

- LaTeX osaa tehdä itse sisällysluettelon käyttäjän tarjoamien osittelukomentojen ansiosta – laita vain komento `\tableofcontents` siihen kohtaan, johon haluat sisällysluettelon tulevan. Helppoa ja mukavaa!
- Dokumenttityyli määrää sisällysluettelon oletustyylin. Sisällysluettelon tyyliin pääsee vaikuttamaan paketin `tocloft` avulla.
- Jo paketin `tocloft` lataaminen tekee sisällysluetteloista paremman näköisiä (ainakin luennoitsijan mielestä).

- Tässä vaiheessa kurssia LaTeX-tietämystä on jo sen verran karttunut, että voimme käydä läpi hieman käytännöllisemmän esimerkin: laboratorioselostuksen rungon.
- Selkkaria tehdessä on tietysti syytä pitää päällimmäisenä mielessä työosaston ohjeet ja säädökset, jotka löytyvät työosaston nettisivuilta.
- Liian orjallisesti työosaston ohjeita ei kuitenkaan kannata noudattaa. Esimerkiksi työosaston tarjoaman (Wordilla tehdyn) malliselkkarin tyyliä ei välttämättä kannata imitoida aivan sellaisenaan.

- Yleisilmeeltään selkkari jäljittelee tieteellistä artikkelia, joten esimerkiksi `article` tai `artikel3` ovat sopivia dokumenttityylejä.
- Ensimmäisenä selkkarissa tulee kansi. Kannen jälkeen on hyvä jättää yksi tyhjä sivu kaksipuoleista tulostusta varten.
- Selkkareissa ei yleensä ole tarvetta sisällysluettelolle.
- Ensimmäisenä kappaleena on *Johdanto*, jonka jälkeen tulevat muut kappaleet, eli tyypillisesti *Teoreettiset lähtökohdat*, *Mittauslaitteisto ja kokeelliset menetelmät*, *Havainnot ja laskut* ja *Johtopäätökset*.
- Alakappaleita (`\subsection`) voi ja kannattaa lisätä tarpeen mukaan.
- Viimeisen kappaleen jälkeen tulee viiteluettelo, jonka jälkeen luetellaan liitteet. Selkkarin lopuksi lisätään liitteet, eli vähintään mittauspöytäkirja.

Alkeellinen selkkaripohja

```
\documentclass[a4paper,twoside,12pt]{article}
\usepackage[T1]{fontenc}
\usepackage[latin1]{inputenc}
\usepackage[finnish]{babel}
\usepackage{enumerate}

\begin{document}
\thispagestyle{empty}
Tähän kansilehti.
\cleardoublepage

\section{Johdanto}

  Tähän tulee Johdanto-kappaleen teksti.

\section{Teoreettiset lähtökohdat}
  \subsection{Alalukuja tarpeen mukaan}
  \subsection{Alalukuja tarpeen mukaan}
  \subsection{Alalukuja tarpeen mukaan}

\section{Mittauslaitteisto ja kokeelliset
menetelmät}
  \subsection{Alalukuja tarpeen mukaan}
  \subsection{Alalukuja tarpeen mukaan}
```

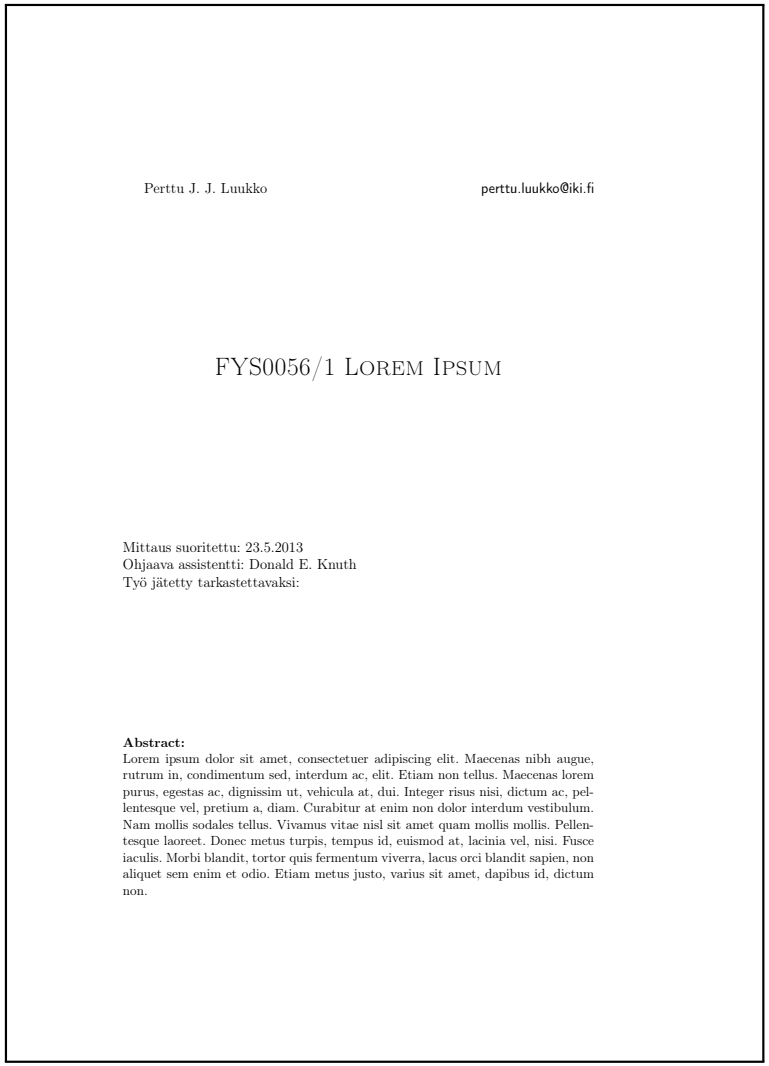
```
\section{Havainnot ja laskut}
  \subsection{Alalukuja tarpeen mukaan}
  \subsection{Alalukuja tarpeen mukaan}

\section{Johtopäätökset}

\clearpage
Tähän lähdeluettelo.

\section*{Liitteet}
\begin{enumerate}[L{ii}te 1:]
  \item Mittauspöytäkirja
\end{enumerate}

\end{document}
```

Vain sisältö puuttuu ☺.

Selkkarin kansilehti

```
Perttu J. J. Luukko\hfill
\textsf{perttu.luukko@iki.fi}
\vfill
\begin{center}
\textsc{\LARGE FYS0056/1 Lorem Ipsum}
\end{center}
\vfill
Mittaus suoritettu: 23.5.2015\\
Ohjaava assistentti: Donald E. Knuth\\
Työ jätetty tarkastettavaksi:
\vfill
\small
\noindent\textbf{Abstract:}\\
Lorem ipsum dolor sit amet, consectetuer
adipiscing elit...
```

Valmis!



Viitteet ja viittaukset

Esipuhe

Tieteellisen dokumentin ytimessä on viittaukset: niin viittaukset muihin teoksiin kuin viittaukset dokumentin sisälläkin. LaTeX sisältää monia keinoja sisäisten ja ulkoisten viitteiden helpottamiseksi, ja näihin keinoihin tutustumme seuraavaksi.

- Sisäisillä viittauksilla tarkoitetaan viittauksia dokumentin sisällä, eli sellaisia lausahduksia kuin *katso kuva 5*, *sijoitetaan yhtälöön 6 sivulla 49* tai *asiaan perehdytään syvemmin kappaleessa 3*.
- Oikean sivu- tai yhtälönumeron liittäminen viittaukseen kannattaa jättää tietokoneen huoleksi, sillä dokumenttia muokkaamalla päätyy helposti tilanteeseen, jossa entinen yhtälö 6 sivulla 49 onkin nyt yhtälö 8 sivulla 55, ja viittaukset tähän yhtälöön osoittavat nyt jonnekin aivan muualle.
- Viittausta tehdessäsi annat viitattavalle asialle eräänlaisen nimilapun, *viittausavaimen*, ja viittaat sivu- tai yhtälönumeron sijasta tähän viittausavaimeen LaTeXin omilla komennoilla. Näin yhtälöiden ja kuvien liikkuessa paikasta toiseen viittaukset pysyvät aina ajan tasalla.

- Viittausavain johonkin kohtaan tekstiä luodaan komennolla `\label`. Tämä komento ottaa argumentikseen kirjoittajan valitseman viittausavaimen, eli käyttäjän valitseman sanan.
- Viitatessa tekstikohtaan LaTeXille annetaan tämä sama viittausavaimen sana, jolloin LaTeX osaa itse katsoa millä sivulla tai missä kappaleessa viittauksen kohde tällä hetkellä sijaitsee.
- Viittausavain saa sisältää kirjaimia, numeroja ja joitain erikoismerkkejä, mutta ei kuitenkaan ääkkösiä.
- Viittausavain on mielivaltainen, mutta oman elämänsä helpottamiseksi kannattaa käyttää selkeitä viittausavaimia. Esimerkiksi Kaijanaho suosii tapaa, jossa viittausavaimen aloittaa viittaukseen kohteen tyyppiä kuvaava lyhennysmerkintä.

- Esimerkiksi jos dokumenttimme sisältää matemaattisen yhtälön, joka on virheen yleinen etenemislaki, voisimme liittää tähän viittausavaimen käyttämällä komentoa `\label{eq:virhelaki}`, tai jos haluamme viitata laitteiston kokoonpanosta otettuun kuvaan, viittausavain voisi olla `\label{fig:laitteisto}`.
- Johdantokappaleeseen voisimme liittää viittausavaimen lisäämällä komennon `\label{sec:johdanto}` jonnekin johdantokappaleen aloittavan `\section`-komennon jälkeen.
- Myös numeroidun luettelon (`enumerate`) alkioihin voi viitata.

- Kun viittausavain on paikallaan, siihen voidaan viitata komennolla `\ref`, joka ottaa argumentikseen viittausavaimen tekstin. Viittauskomennon paikalle tekstissä ladotaan kappaleen, yhtälön tai taulukon numero.
- Viittausavaimen paikan sivunumeroon voi viitata komennolla `\pageref`.
- Viittauksissa on hyvä käyttää rivinvaihdon kieltävää välilyöntiä ~ ennen numeroa.

Näin viitataan

```
Virheen laskemiseksi sijoitetaan mittaustulokset
yhtälöön~\ref{eq:virhelaki} sivulla~\pageref{eq:virhelaki}.
Mittauslaitteiston kokoonpano on esitetty
kuvassa~\ref{fig:laitteisto}.
```

Esimerkki sisäisistä viittauksista

Esimerkki

```
\begin{enumerate}
  \item Vompatti \label{vompatti}
  \item Tursas \label{tursas}
  \item Koppelo \label{koppelo}
\end{enumerate}

Eläin numero~\ref{tursas} on
nilviäinen, mutta
eläimet~\ref{vompatti}
ja~\ref{koppelo} eivät.
```

Tuloste

1. Vompatti
2. Tursas
3. Koppelo

Eläin numero 2 on nilviäinen, mutta eläimet 1 ja 3 eivät.

- Koodia käännettäessä LaTeX lukee koodin alusta loppuun, joten jos lisäät uuden viittausavaimen, ja viittaat tähän aiemmasta kohdasta tekstiä, viittaus ei tule ensimmäisellä kerralla ladottua, sillä LaTeX ei ole vielä nähnyt viittausavainta.
- Tällaisessa tilanteessa LaTeX varoittaa, *"Label(s) may have changed. Rerun to get cross-references right."*, eli käännettäessä toisen kerran viittaus menee paikalleen.
- Useimmat LaTeXin käyttöön tarkoitetut editorit tunnistavat LaTeXin kehotuksen kääntää koodi kahdesti, ja tekevät niin vaivaamatta käyttäjää.
- Kääntämisen automatisointiin voi käyttää myös esimerkiksi komentoriviohjelmia `latexmk` tai Rubber:
  `http://ctan.org/pkg/latexmk`
  `https://launchpad.net/rubber`

- Sisäisten viittauksien lisäksi tieteellinen teksti tietysti vilisee myös viittauksia muuhun kirjallisuuteen.
- Sisäisten viittausten tavoin myös kirjallisuusviitteille annetaan tietty viittausavain, jota käyttämällä viittaukset saadaan ladottua tekstin sekaan asianmukaisesti.
- LaTeX osaa myös latoa automaattisesti kirjallisuusluettelon niistä kirjallisuusviitteistä, joita tekstissä on käytetty.

- Kirjallisuuslista ladotaan ympäristöllä `thebibliography`.
- Ympäristön sisällä listataan kirjallisuusviitteitä komennolla `\bibitem`, joka ottaa pakolliseksi argumentikseen viittausavainteksti, ja valinnaiseksi argumentikseen tekstin, joka ladotaan tekstiin viittauksen tilalle. Mikäli valinnainen argumentti jätetään pois, käytetään viittaustekstinä vain numeroa.
- Komennon `\bibitem` jälkeen kirjoitetaan tästä kirjallisuusviitteestä se teksti, joka halutaan näkyviin kirjallisuusluetteloon.
- Ympäristö `thebibliography` vaatii latomista varten pakolliseksi argumentikseen *pisimmän* viittaustekstin.
- Koska on tarpeetonta laittaa ihmistä muotoilemaan kirjallisuusluetteloa, on kaiken automatisointiin myös keino nimeltä BibTeX, joka esitellään lyhyesti piakkoin.

Esimerkki

```
\begin{thebibliography}{Luukko~2015}
\bibitem[VR~2015]{YKVR2015} YK:n varaaniraportti 2015.
\bibitem[Luukko~2015]{luukko2015} Perttu Luukko.
\emph{Tieteellisen tekstin tuottaminen \LaTeX illa}.
Luentomuistiinpanot, 2015.
\end{thebibliography}
```

Tuloste tässä dokumenttityylissä


YK:n varaaniraportti 2015.


Perttu Luukko. *Tieteellisen tekstin tuottaminen LaTeXilla*. Luentomuistiinpanot, 2015.

Kirjallisuuteen viitataan komennolla `\cite`, joka ottaa pakolliseksi argumentikseen viittausavaimen ja valinnaiseksi argumentiksi viittauksen yhteyteen ladottavan lisätekstin, jolla voidaan ilmaista esimerkiksi viittaus tiettyyn sivuun tai kappaleeseen.

Esimerkki

```
Lisää tietoa viittaamisesta saa esimerkiksi \LaTeX-kurssin
muistiinpanoista~\cite[osa~3]{luukko2015}.\\
Myös YK on nostanut varaanit suurimmaksi ihmiskuntaa uudella
vuosituhannella kohtaavaksi uhaksi~\cite{YKVR2015}.
```

Tuloste

Lisää tietoa viittaamisesta saa esimerkiksi LaTeX-kurssin muistiinpanoista [Luukko 2015, osa 3].
Myös YK on nostanut varaanit suurimmaksi ihmiskuntaa uudella vuosituhannella kohtaavaksi uhaksi [VR 2015].

- BibTeX vähentää kirjallisuusluettelon latomiseksi tarvittavaa manuaalista työtä. Tiedot kirjallisuudesta kerätään omaan tiedostoonsa eräänlaiseen kirjallisuustietokantaan. Tässä tietokannassa on lueteltu kunkin teoksen tiedot, kuten tekijä, otsikko, vuosi ym.
- Varsinainen kirjallisuusluettelon latominen tehdään automaattisesti, ja kirjallisuusluettelon sekä viitteiden ulkoasun määrää erillinen tyylitiedosto.
- Näin esimerkiksi pitkän kirjallisuusluettelon tyyliä voidaan vaihtaa helposti käymättä käsin läpi jokaista kirjallisuusviitettä erikseen.
- ☹ BibTeX on toteutukseltaan ehkä vanhentunein ja sekavin LaTeX-maailman ohjelma. Nykyihmisen kannattaa perehtyä myös (tai sen sijasta) ohjelmaan biblatex (`http://www.ctan.org/pkg/biblatex`), joka on korvaamassa vanhaa BibTeXiä.

- BibTeXin kirjallisuustietokanta tallennetaan tekstitiedostoon, jonka pääte on `.bib`.
- Tämä tietokanta noudattaa tarkkaa syntaksia, jota ei tässä käydä kovin syvällisesti läpi – katso esimerkiksi sivut 79–81 Kaijanahon kirjasta tai `http://en.wikibooks.org/wiki/LaTeX/Bibliography_Management#BibTeX`.
- Monet Internetin artikkelitietokannat tarjoavat kustakin teoksesta valmiin BibTeX-pätkän, jonka voi liittää suoraan työnsä `.bib`-tiedostoon.

esimerkki BibTeX-kirjallisuustietokannassa olevasta määrittelystä

```
@book{Knuth86,
  author = {Knuth, Donald E[rvin]},
  title = {The {\TeX book}},
  publisher = {Addison-Wesley},
  address = {Reading, Massachusetts},
  year = {1986}
}
```

- Kirjallisuusluettelon ja viittausten tyyli määritellään komennolla `\bibliographystyle`. Komento ottaa argumentikseen tyylitiedoston nimen. Tavallisimpia tyylejä ovat esimerkiksi `plain` ja `unsrt`.
- Esimerkkejä eri luettelotyyleistä löytyy esimerkiksi osoitteesta `http://www.univie.ac.at/nuhag-php/bibtex/bibstyles.pdf`.
  - Tyylitiedostoja voi tehdä myös itse – jos on valmis opettelemaan *jännän* ohjelmointikielen ☺/☹.
- Selkkareihin käy hyvin esimerkiksi `plain`-tyylistä suomennettu `finplain` tai `unsrt`.
- Kun kirjallisuusluettelon tyyli on määritelty, komento `\bibliography` latoo valmiin viiteluettelon. Komento ottaa argumentikseen kirjallisuustietokannan tiedostonimen (`.bib`-päätteen voi jättää pois).

- BibTeXiä käytettäessä tarvittavien käännöskertojen määrä kasvaa entisestään. Jos kirjallisuusviitteet muuttuvat, tarvitaan seuraavat operaatiot:
  Oletetaan, että esimerkissä LaTeX-koodi on tiedostossa `gradu.tex`
  1. LaTeX-koodi käännetaan normaalisti: `pdflatex gradu.tex`
  2. BibTeX ajetaan komennolla `bibtex gradu` (huomaa: *ei päätettä*). Nyt BibTeX tietää mihin kaikkeen tiedostossa `gradu.tex` viitataan, ja tekee `thebibliography`-ympäristön valmiiksi.
  3. LaTeX-koodi käännetään vielä kahdesti, jolloin LaTeX osaa latoa viittaukset tekstin sekaan oikein.
- Hyvät LaTeX-editorit, Rubber ja `latexmk` osaavat toki automatisoida tämänkin.
- BibTeXin tai biblatexin käyttöä ei tämän kurssin puitteissa opeteta tämän syvällisemmin – se vaatisi todennäköisesti kokonaan oman luentonsa. Asiaan kannattaa kuitenkin perehtyä itsenäisesti jos omat kirjallisuusluettelot alkavat olla pitkiä.

- PDF on siitä mukava dokumenttiformaatti, että se tukee esimerkiksi hyperlinkkejä, eli suomeksi sanottuna sitä, että klikkaamalla jotain kohtaa pääsee jonnekin muualle.
- LaTeXin tuotteisiin hyperlinkit saa käyttämällä pakettia `hyperref`. Jo paketin lataamalla saa hyperlinkit automaattisesti esimerkiksi sisäisiin viittauksiin.
- `hyperref` on varsin monipuolinen paketti, joten sitä ei luentojen puitteissa käsitellä aivan kokonaan. Kiinnostuneet lukekoot dokumentaatiota sivulta `http://www.tug.org/applications/hyperref/`.
- ☹ `hyperref` muuttaa LaTeXin sisäistä toimintaa melko radikaalisti, mikä johtaa joskus ongelmiin. Ongelmatapauksissa auttaa usein `pdflatex`in luomien väliaikaistiedostojen poistaminen (.aux jne).

- URL-osoitteita ei voi latoa aivan kuten pitkiä sanoja, joten niiden kanssa kannattaa käyttää valmista pakettia `url`.
- Paketti tarjoaa komennon `url`, joka ottaa argumentikseen ladottavan osoitteen.
- Osoitteiden ladonnan tyyliin voi vaikuttaa alustukseen laitettavalla komennolla `\urlstyle`: esimerkiksi `\urlstyle{rm}` latoo osoitteet tavallisella fontilla, ja `\urlstyle{sf}` pääteviivattomalla fontilla eli kuten komento `\textsf`.
- Paketti `hyperref` lataa tämän paketin automaattisesti. Lisäksi `hyperref`in kanssa käytettynä `\url`-komennolla ladotut osoitteet ovat myös hyperlinkkejä.

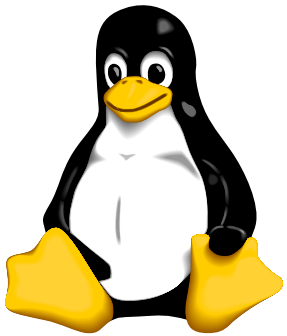
- LaTeX osaa myös latoa aakkostetut hakemistot helposti ja kätevästi, siten että sivunumerot pysyvät järjestyksessä.
- Automatisoitu hakemisto saadaan paketilla `makeidx`, ja lisäämällä alustukseen komento `\makeindex`.
- Lyhykäisyydessään hakemiston kokoaminen onnistuu siten, että tekstin sekaan liitetään `\index`-komentoja haluttuihin kohtiin. Komento ottaa argumenttinaan hakemistoon ladottavan tekstin.
- Tämän jälkeen komento `\printindex` latoo valmiin hakemiston.
- Tämän enempää hakemistoja ei tämän kurssin puitteissa käydä. Innokkaat voivat tutustua asiaan lisää lukemalla esimerkiksi kappaleen 4.3 *Pitkänpuoleisesta johdannosta*.

Kuvat ja taulukot

Esipuhe

Aina leipäteksti ei ole paras keino välittää haluamaansa viestiä. Tällöin kirjoittaja turvautuu usein tekstin seassa esitettäviin kuviin ja taulukoihin, joiden latomista käsittelemme seuraavaksi. Kuvien ja taulukoiden kanssa kannattaa kuitenkin olla tarkkana, sillä vaikka hyvä kuva kertoo enemmän kuin tuhat sanaa, huono kuva sekoittaa lukijan pään pahemmin kuin tuhat sanaa valhetta.

- Kuvia pääsee liittämään dokumenttiinsa lataamalla paketin `graphicx`.
- Kuvan liittäminen käy tämän jälkeen niinkin helposti kuin käyttämällä komentoa `\includegraphics`, joka ottaa argumenttinaan kuvan tiedostonimen. Tiedostopäätteen voi jättää pois.
- Kuva näkyy tulosteessa sellaisenaan. Kuvien skaalaamisesta ja kuvatekstien lisäämisestä lisää myöhemmin.



Esimerkki:

Tässä on teille esimerkiksi kuva muuan pingviinistä.
Kuva saatiin tähän komennolla `\includegraphics{Tux}`.

Siltä varalta että joku kysyy, kuvan on piirtänyt Larry Ewing GIMP-kuvankäsittelyohjelmalla. Vektoroidun version on tehnyt Simon Budig.

- LaTeX ei varsinaisesti käsittele tai tulkkaa liitettäviä kuvia – se yksinkertaisesti liittää ne lopullisen dokumentin mukaan. Näin ollen `\includegraphics`in hyväksymien kuvaformaattien määrää rajoittaa lopullisen dokumentin tiedostoformaatti.
- PDF-dokumentteihin voi liittää kuvia ainakin PDF- JPEG- ja PNG-muodoissa.
- Mikäli lopullisena dokumenttiformaattina on DVI (ts. käytetään kääntämiseen `latex`-ohjelmaa), PDF-kuvat eivät käy, mutta esimerkiksi EPS-kuvat käyvät.

- PDF ja EPS tukevat vektorigrafiikkaa, eli kuvia, jotka koostuvat geometrisista elementeistä (viiva, laatikko, pala tekstiä…). Vektorigrafiikka soveltuu erityisesti viivapiirroksiin ja kuvaajiin.
- JPEG ja PNG ovat bittikarttaformaatteja, eli kuva koostuu pikseleistä.
  Bittikarttaformaatteja käytetään, kun kuvaa ei voida esittää geometrisina elementteinä (esim. valokuvat).
- Vektorimuotoinen kuva ei kärsi skaalauksesta, joten vektorigrafiikkaa kannattaa käyttää aina kun suinkin mahdollista.

Varo suhrua!

JPEG on valokuville tarkoitettu kuvaformaatti! Formaatin käyttämä pakkausalgoritmi tuhraa suuria kontrasteja sisältävät kuvat (esimerkiksi matemaattiset kuvaajat tai viivapiirrokset).

- Jos kuvasi on kuvaaja, pyri tallentamaan se PDF-muodossa mikäli käytät `pdflatex`-kääntäjää, ja EPS-muodossa mikäli et.
  - Mikäli et käytä `pdflatex`-kääntäjää, kysy itseltäsi miksi.
- Mikäli kuvasi on kuvaaja, mutta PDF-tiedostosta tulisi valtavan suuri (esim. miljardeja datapisteitä), tallenna kuvasi PNG-muotoon. PNG on *hävikitön* formaatti, eli se ei suttaa kuvia kuten JPEG. Varmista kuitenkin riittävän suuri resoluutio ettei kuva pikselöidy.
- Mikäli kuvasi on valokuva, tallenna se JPEG-muodossa. Skaalaa kuvasi kuitenkin valmiiksi sopivan kokoiseksi kuvankäsittelyohjelmalla, jolloin vältyt kasvattamasta dokumenttisi kokoa tarpeettomasti.

- Useinkaan kuva ei ole sellaisenaan halutun kokoinen. Tällöin kuvaa voidaan skaalata komennon `\includegraphics` valinnaisilla argumenteilla. Komennolle voi antaa useita valinnaisia argumentteja pilkulla erotettuna listana.
- Argumenteilla `height`=*pituus* tai `width`=*pituus* voidaan skaalata kuva tietyn korkuiseksi tai levyiseksi. Esimerkiksi aikaisempi pingviinikuva saatiin oikeasti komennolla `\includegraphics[height=6em]{Tux}`.
- `\includegraphics` osaa myös sopivilla argumenteilla esimerkiksi pyörittää kuvia tai rajata kuvasta tietyn alueen.

Vinkki

LaTeXin pituusmääreinä voidaan käyttää myös LaTeXin tuntemia, dokumentin kokoa kuvaavia pituussuureita. Näin esimerkiksi voidaan skaalata kuva täyttämään 75% tekstin leveydestä antamalla `\includegraphics`-komennolle valinnainen argumentti `width=0.75\textwidth`. Näppärää!

Tämän kurssin puitteissa ei käsitellä piirto- tai kuvankäsittelyohjelmia. Alla kuitenkin pari vinkkiä.

- Työosaston selkkariohjeen kehuma Microsoft Paint ei ihan aikuisten oikeasti ole kovin hääppöinen ohjelma.
- Myöskään monet muut yliopiston koneilta löytyvät piirto-ohjelmat eivät tuota niin hyvää laatua, kuin hintalapun perusteella voisi kuvitella.
- Kuvaajien piirtämiseen esimerkiksi Gnuplot (`http://www.gnuplot.info/`) on monien mielestä varsin näppärä. Pythonia osaavien kannattaa puolestaan tutustua PyX-kirjastoon.
- Kaavioiden ja muiden viivapiirrosten tekemiseen Inkscape (`http://www.inkscape.org/`) on erinomainen valinta.

Taulukkomuotoisen materiaalin latomiseen käytetään LaTeXin ympäristöä `tabular`.

- `tabular` ottaa pakollisen argumentin, jossa kuvataan sarakkeiden määrä ja kunkin sarakkeen tasaus. Kutakin saraketta kohti argumenttiin lisätään kirjain seuraavasti:

  `l` Vasemmalle tasattu sarake
  `c` Keskitetty sarake
  `r` Oikealle tasattu sarake

  Näin esimerkiksi `\begin{tabular}{llr}` aloittaisi kolmisarakkeisen taulukon, jossa sarakkeista kaksi ensimmäistä tasattaisiin vasempaan reunaan ja kolmas oikeaan reunaan.
- Ympäristön sisällä alkiot erotetaan toisistaan &-merkillä, ja rivinvaihto kuvataan normaalilla \\-komennolla.

Alla esimerkki LaTeXilla luodusta taulukosta. Esimerkissä on käytetty jo booktabs-paketin vaativia vaakaviivoja, jotka esitellään pian.

Esimerkki

```
\begin{tabular}{lr}
  \toprule
  Eläin & Jännite (V) \\
  \midrule
  Vompatti & 320 \\
  Tursas   & 1000 \\
  Koppelo  & 110 \\
  \bottomrule
\end{tabular}
```

Tuloste

Eläin	Jännite (V)
Vompatti	320
Tursas	1000
Koppelo	110

Artikkelissaan *Publication quality tables in LaTeX* Simon Fear on kiteyttänyt selkeiden ja julkaisulaatuisten taulukoiden latomisen seuraaviin viiteen helppoon nyrkkisääntöön:

1. Älä käytä pystyviivoja.
2. Älä käytä kaksoisviivoja.
3. Kirjoita yksiköt sarakkeiden otsikkoon, älä taulukoon.
4. Älä jätä desimaalipilkkua tai -pistettä edeltävää nollaa pois: ei ”,1” vaan ”0,1”.
5. Älä käytä ”ks. yllä”-merkintöjä. Jos tyhjä paikka taulukossa ei riitä toiston osoittamiseksi, toista eksplisiittisesti.

- Kuten sovimme, selkeissä taulukoissa ei käytetä pystyviivoja. Vaakaviivoja puolestaan kannattaa käyttää, mutta hillitysti (eli *ei* jokaisen rivin jälkeen!).
- Vaakaviivoja voi taulukkoon lisätä komennolla `\hline`, mutta siistimpiä viivoja saa aikaan itsensä Simon Fearin `booktabs`-paketilla.
- `booktabs`-paketin kolme perusviivaa ovat `\toprule`, joka aloittaa taulukon, `\bottomrule` joka lopettaa taulukon, sekä `\midrule`, joka erottaa sarakeotsikot muusta taulukosta.
- Lisäksi joskus on hyvä käyttää otsikossa vain muutaman sarakkeen levyistä vaakaviivaa esimerkiksi otsaketietojen ryhmittelemiseen. Tämän saa `booktabs`-paketin komennolla `\cmidrule`.

- Komento `\cmidrule` ottaa yhden pakollisen ja yhden valinnaisen argumentin. Pakollinen argumentti kertoo sen, minkä sarakkeiden kohdalle viiva tulee piirtää: esim. `1-2` tai `2-5`.
- Valinnaisella argumentilla viivaa voidaan hieman typistää jommasta kummasta (`r` tai `l`) tai molemmista (`lr`) päistä. Tyypillisesti viivaa typistetään hieman niistä päistä, jotka eivät osu taulukon reunaan.
- Poikkeuksellisesti typistysargumentti kirjoitetaan kaarisulkujen sisään!

Oikeasta reunasta hieman typistetty vaakaviiva sarakkeisiin 1 ja 2

```
\cmidrule(r){1-2}
```

Molemmista reunoista hieman typistetty vaakaviiva sarakkeisiin 3, 4 ja 5

```
\cmidrule(lr){3-5}
```

- Komennolla `\multicolumn` voi määritellä taulukkoalkioita, jotka ulottuvat useamman sarakkeen kohdalle. Komentoa käytetään `tabular`-ympäristön sisällä siinä kohdassa, johon tällainen alkio halutaan.
- Komento ottaa kolme pakollista argumenttia, joista ensimmäinen on käytettävien sarakkeiden lukumäärä, toinen kertoo tasauksen (`l`, `c` tai `r`), ja kolmas antaa alkion sisällön.
- Komennon käytöstä on annettu esimerkki seuraavan kalvon taulukossa.

Toinen esimerkkitaulukko

Esimerkki

```
\begin{tabular}{llr}
  \toprule
  \multicolumn{2}{c}{Eläin} \\
  \cmidrule(r){1-2}
  Laji & Sukupuoli & Jännite (V) \\
  \midrule
  Vompatti & uros   & 320  \\
  Tursas   & uros   & 1000 \\
  Metso    & uros   & 50   \\
           & naaras & 110  \\
  \bottomrule
\end{tabular}
```

Tuloste

Eläin		
Laji	Sukupuoli	Jännite (V)
Vompatti	uros	320
Tursas	uros	1000
Metso	uros	50
	naaras	110

Näillä ohjeilla saanee jo esimerkiksi selkkareihin varsin tyylikkäät ja ymmärrettävät taulukot. LaTeXin repertuaari ei tietenkään tähän kuitenkaan pääty.

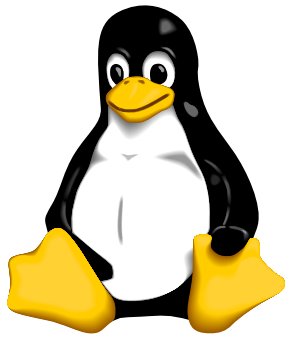
- Paketilla `multirow` saa `\multicolumn`ia vastaavan komennon, jolla voi määritellä monirivisiä alkioita.
- Paketilla `rotating` saa (muun muassa) ladottua suuret taulukot vaakasuunnassa.
- Paketilla `tabularx` saa sarakkeita, joiden leveys ”venyy” kunnes koko taulukko on halutun levyinen.
- Paketilla `dcolumn` voi tasata alkiot tietyn merkin (esim. desimaalipilkun) kohdalle.
- Katso esimerkiksi osoitteesta `http://en.wikibooks.org/wiki/LaTeX/Tables` lisää vinkkejä taulukoiden ladontaan.

- Kuvat ja taulukot ovat tavallisin esimerkki leijuvaisista (eng. *floats*), eli tekstistä erillisistä palasista, joiden sijoittelu voidaan antaa LaTeXin huoleksi.
- Näin kirjoittajan ei tarvitse huolehtia siitä, onko esimerkiksi jollain sivulla tarpeeksi tilaa tietynkokoiselle kuvalle tai taulukolle, sillä LaTeX osaa itse löytää leijuvaisille sellaiset paikat, johon ne hyvin mahtuvat, ja jossa ne eivät häiritse tekstin lukemista liiallisesti.
- Kuvien ja taulukoiden automattinen sijoittelu on ehkä suurin yksittäinen aihe, josta LaTeXia puutteellisesti osaavat purnaavat. Tyypillisesti syynä on se, että ei ymmärretä kuinka sijoittelualgoritmi toimii, ja kuinka siihen voi vaikuttaa.

- Ympäristö `figure` määrittelee leijuvan kuvan. Tämän ympäristön sisällä tyypillisesti liitetään itse kuvatiedosto `\includegraphics`-komennolla, sekä annetaan kuvalle kuvateksti sekä mahdollinen viittausavain.
- Kuvateksti määritellään komennolla `\caption`, joka ottaa argumentikseen kuvatekstin.
- Komento `\centering` keskittää ympäristön sisällä olevan sisällön. Komento `\begin{center}` aloittaisi uuden kappaleen, mikä loisi ylimääräistä tilaa pystysuunnassa.

Esimerkki

```
\begin{figure}
  \centering
  \includegraphics[height=6em]{Tux}
  \caption{Muuan pingviini.}
  \label{fig:tux}
\end{figure}
```



Kuva 1: Muuan pingviini.

- Ympäristö `table` määrittelee vastaavalla tavalla leijuvan taulukon. Itse taulukko tehdään edelleen käyttämällä `tabular`-ympäristöä `table`-ympäristön sisällä.
- Komento `\caption` antaa vastaavalla tavalla taulukon kuvaustekstin. Jos taulukon kuvateksti halutaan taulukon yläpuolelle (kuten selkkareilta vaaditaan), tulee komento kirjoittaa ennen `tabular`-ympäristöä.
- Kuten tarkkasilmäinen lukija varmasti huomaa, jos taulukon kuvauksen siirtää taulukon yläpuolelle, ei taulukon ja kuvauksen välinen väli näytä oikealta. Tämä johtuu siitä, että LaTeX ei huomaa tätä vaihdosta vaan latoo kuvauksen alle edelleen yhtä suuren välin, kuin jos kuvaus olisi taulukon alla ja teksti jatkuisi kuvauksen jälkeen. Välit saa kuntoon lataamalla paketin `ftcap` tai paketin `caption` argumentilla `tableposition=top`.

Esimerkki

```
\begin{table}
  \caption{Eläinten jännitteet.}
  \centering
  \begin{tabular}{lr}
    \toprule
    Eläin & Jännite (V) \\
    \midrule
    Vompatti & 320 \\
    Tursas & 1000 \\
    Koppelo & 110 \\
    \bottomrule
  \end{tabular}
  \label{tab:esimerkki}
\end{table}
```

Taulukko 1: Eläinten jännitteet.

Eläin	Jännite (V)
Vompatti	320
Tursas	1000
Koppelo	110

- Kuten aikaisemmista esimerkeistä nähtiin, kuvateksteihin ei itse tarvitse lisätä numeroita, sillä LaTeX numeroi kuvat ja taulukot itse.
- LaTeX myös itse lisää kuvatekstiin sanan ”Kuva” tai ”Taulukko”, tietysti `babel`-paketilla valitusta kielestä riippuen.
- Kuvat ja taulukot numeroidaan erikseen. Jotkin isommille dokumenteille tarkoitetut dokumenttiluokat numeroivat kuvat ja taulukot lisäksi kappalekohtaisesti siten, että esimerkiksi luvun 5 ensimmäinen kuva on kuva 5.1, ja luvun 6 toinen kuva 6.2.
- Numerointityyliä voi myös itse vaihtaa, esimerkiksi jos haluaa artikkeliinsa kappalekohtaisen numeroinnin tai jos haluaa taulukot numeroitavan roomalaisilla numeroilla.

Ensisijainen keino vaikuttaa leijuvaisten sijoitteluun on `figure`- ja `table`-ympäristöille annettava valinnainen argumentti, joka koostuu seuraavista sijoittelukirjaimista:

- `h` Sijoittelu tekstin sekaan jonnekin lähistölle (*here*).
- `t` Jonkin sivun ylälaitaan (*top*).
- `b` Jonkin sivun alalaitaan (*bottom*).
- `p` Erilliselle sivulle, jossa on vain muita leijuvaisia (*page of floats*).
- `!` Sijoittele ohittaen joitain sijoittelun rajoituksia.
- `H` Juuri tähän, ohittaen sijoittelualgoritmin kokonaan (vaatii paketin `float`).

Näin siis esimerkiksi `\begin{figure}[htp]` aloittaa kuvan, joka voidaan sijoittaa tekstin sekaan, jonkin läheisen sivun ylälaitaan, tai vain leijuvaisia sisältävälle sivulle. Kirjaimien järjestyksellä ei ole väliä.

- Sijoittelussaan LaTeX pyrkii säilyttämään tekstin luettavuuden, eli LaTeX ei esimerkiksi laita sivuja liian täyteen kuvia, eikä lisää mielellään liian suuri kuvia tekstin sekaan.
- Mikäli kuvien sijoittelussa kohtaa ongelmia, esimerkiksi latoessa kymmeniä kuvia sisältävää lyhyttä selkkaria, kannattaa ensisijaisesti miettiä tiedon esittämistä.
- Jos dokumentissa on paria tekstiriviä kohti yksi kuva, kuvien liittäminen tekstin sekaan saisi lopputuloksen näyttämään vähintäänkin sekavalta. Tällöin kannattaa erityisesti harkita leijuvaissivujen (sijoittelukirjain `p`) tai kuvakollaasien (esitellään myöhemmin) käyttöä.
- LaTeXin sijoittelualgoritmin parameterejä pääsee myös vapaasti säätämään, mikäli haluaa esimerkiksi sallia sivut, joilla on vain vähän tekstiä ja yksi suuri kuva. Tämäkin esitellään myöhemmin.

- Kuviin ja taulukoihin voi tehdä sisäisiä viittauksia lisäämällä leijuvaiselle viittausavaimen käyttämällä aiemmin esiteltyä `\label`-komentoa ympäristön sisällä.
- Koska leijuvaiselle annetaan numero komennon `\caption` kohdalla, `\label`-komennon on tultava `\caption`-komennon jälkeen!

Esimerkki

Aikaisemmassa esimerkissä annoimme kuvalle viittausavaimen `fig:tux`. Nyt esimerkiksi koodipätkä

```
Kuvassa~\ref{fig:tux} näkyi muuan pingviini.
```

tuottaa näkyviin

Kuvassa 1 näkyi muuan pingviini.

Osa IV

Matemaattinen materiaali

Johdatus matemaattisen materiaalin ladontaan

Esipuhe

Matemaattisten lausekkeiden ja merkintöjen tyylikäs ladonta on LaTeXin vahvimpia puolia, ja eräs syy LaTeXin vahvaan suosioon matemaatikoiden ja luonnontieteilijöiden keskuudessa. Tässä osassa opettelemme lisäämään kirjoitelmiin matemaattista höystettä, joka ei eksytä lukijaa tai särje tämän silmiä, vaan tukee tekstiä ja edistää asiamme ymmärtämistä.

- TeX ja LaTeX tarjoavat sellaisenaan varsin mittavan joukon työkaluja matemaattisen materiaalin ladontaan.
- Amerikkalainen matemaatikkoyhdistys *American Mathematical Society* eli AMS oli eräitä TeXin varhaisia kannattajia, ja yhdistys onkin sittemmin vaikuttanut suuresti TeXin ja LaTeXin matematiikkapuoleen.
- AMS on tuottanut mittavia matematiikan ladontaan tarkoitettuja paketteja ja dokumenttiluokkia LaTeXiin, ja LaTeXia näillä paketeilla varustettuna kutsutaankin joskus nimellä AMS-LaTeX.
- Jatkossa kerrotaan erikseen kun jokin komento tarvitsee jonkin AMS:n paketeista. Yleisesti ottaen vähintään AMS-LaTeXin peruspaketti `amsmath` kannattaa ladata aina matemaattista materiaalia sisältävissä dokumenteissa.

- Koska matemaattinen materiaali on typografian kannalta aivan oma ”kielensä”, joka noudattaa aivan eri sääntöjä kuin tavanomainen teksti, matemaattiselle materiaalille on LaTeXissa aivan oma matematiikkatilansa.
- Ns. tekstimatematiikkatila, joka on tarkoitettu matemaattisten merkintöjen latomiseen tekstin sekaan, aloitetaan komennolla `\(` ja lopetetaan komennolla `\)`. Näiden komentojen välissä olevan koodin LaTeX tulkitsee matemaattiseksi lausahdukseksi.
- Suuremmille merkinnöille on lisäksi oma näyttömatematiikkatilansa, joka latoo matemaattiset merkinnät korostuksenomaisesti omalle rivilleen.
- Tekstimatematiikkatilan voi erottaa tekstistä myös dollarimerkeillä (`$`), joita TeX käyttää, mutta puttuva dollarimerkki koodissa aiheuttaa vaikeammin ymmärrettäviä virheilmoituksia kuin puuttuva `\)`.

Matematiikkatilaa ei kannata käyttää tekstiä varten eikä tekstitilaa matematiikkaa varten:

- Matematiikkatilassa kirjoitetusta tekstistä on hävinnyt sanavälit, ja kirjaimet ovat eri fontilla.
- Tekstitilassa kirjoitettu matematiikka puolestaan näyttää varsin amatöörimäiseltä: yhtälön kirjaimet eivät erotu tekstistä, plus- ja miinusmerkin ympärille ei ole jätetty tilaa, ja miinusmerkin paikalla onkin tavuviiva.

Esimerkki

```
\( hauki on kala \)
x+y-1 = 2
\( x+y-1 = 2 \)
```

Tuloste

haukionkala
x+y-1 = 2
$x + y - 1 = 2$

- Näyttömatematiikkatilaa käytetään yleensä silloin, kun ladottava lauseke on liian suuri mahtuakseen tekstin sekaan, tai kun halutaan erityisesti korostaa jotain tiettyä matemaattista lausahdusta.
- Näyttömatematiikkatila aloitetaan komennolla `\[` ja päätetään komennolla `\]`.
- Useimmat dokumenttityylit keskittävät näyttömatematiikatilassa ladotut lausekkeet. Lausekkeet saa siirtymään hieman sisennettynä vasempaan reunaan antamalla paketille `amsmath` valinnaisen argumentin `fleqn`.
- Kuten kaikkia korostuskeinoja, myös näyttömatematiikkatilaa kannattaa käyttää säästeliäästi.
- Jos lauseke on liian suuri mahtuakseen yhdelle riville, tarvitaan erityisiä temppuja ja ympäristöjä, jotka käsitellään myöhemmin.

Cauchyn kaava analyyttisten funktioiden derivaatoille on hieman liian suuri ladottavaksi tekstin sekaan, sillä siellä se näyttäisi tältä: $f^{(n)}(z) = \frac{n!}{2\pi\mathrm{i}} \oint_\gamma \frac{f(\xi)}{(\xi-z)^{n+1}}\,\mathrm{d}\xi$. Omalla rivillään tekstistä erotettuna se kuitenkin pääsee täyteen loistoonsa:

$$f^{(n)}(z) = \frac{n!}{2\pi\mathrm{i}} \oint_\gamma \frac{f(\xi)}{(\xi - z)^{n+1}}\,\mathrm{d}\xi.$$

Edellisen yhtälön tuottanut koodi

```
\[
  f^{(n)}(z) = \frac{n!}{2\piup\mathrm{i}} \oint_\gamma
  \frac{f(\xi)}{(\xi - z)^{n+1}}\,\mathrm{d}\xi
\]
```

- Tavallisten numeroiden ja kirjaimien tuottaminen matematiikkatilassa ei eroa tekstitilasta kuin ulkoasun osalta: `$x$` tuottaa x ja `$1=2$` tuottaa $1 = 2$.
- On kuitenkin syytä huomata, että matematiikkatilan kirjaimet näyttävät erilaiselta kuin tekstitilan kirjaimet – kuten pitääkin. Itse asiassa LaTeX latoo matematiikkatilan eri fontilla erottaakseen sen tavallisesta tekstistä, ja yleensä tehdäkseen matematiikasta matematiikan näköistä.
- Matematiikassa ei ole yleensä tapana käyttää esimerkiksi ääkkösiä symboleina, joten tällaiset kirjaimet usein puuttuvat matematiikkafonteista.

- Matematiikan latomiseen käytetään omaa fonttiaan, joka on suunniteltu matematiikan ladontaan.
- Oletuksena LaTeX käyttää matematiikan ladontaan oletustekstifontin *Computer Modern* kumppania *Computer Modern Math*.
- Helppo tapa pilata kaunis ulkoasu on valita huonosti tekstifonttiin sopiva matematiikkafontti. Näin käy helposti, jos esimerkiksi vaihtaa tekstifontin, mutta pitää matematiikkafontin ennallaan.
- Hyvä ja moderni yhdistelmä Palatinon sukuisella tekstifontilla on paketit `newpxtext` ja `newpxmath`.
- Jos Times miellyttää, kokeile pakettia `stix`.
- Lisää hyviä fontteja ja fonttiyhdistelmiä voi shoppailla jo mainitusta kuvastosta `http://www.tug.dk/FontCatalogue/`.

Suomenkielisessä tekstissä ei ole valinnanvaraa desimaalierottimen suhteen. Koska väärää desimaalierotinta näkee kuitenkin selkkareissa säännöllisesti, on ehkä syytä jakaa seuraava muistutus:

Muistutus desimaalierottimesta

Suomen kielessä desimaalierotin on pilkku.

Pilkun käyttöön desimaalierottimena liittyy pieni huomionarvoinen seikka: LaTeX olettaa, että pilkkua käytetään matematiikkatilassa ilmaisuissa kuten $F(x, y, z) = 0$, ja siksi jättää automaattisesti hieman väliä pilkun jälkeen. Jos pilkkua käytetään myös desimaalierottimena, tämä ei tietenkään käy. Paketti `icomma` korjaa tämän.

Jos kirjoitat suomea ja käytät matematiikkatilassa desimaalilukuja, muista seuraava.

```
\usepackage{icomma}
```

- Matematiikka vilisee kreikkalaisia aakkosia, joten niitä on syytä osata lisätä yhtälöihinsä. Kullekin aakkoselle on oma komentonsa, joka koostuu kenoviivasta ja aakkosen englanninkielisestä nimestä.
- Suuraakkosversio saadaan korvaamalla nimen ensimmäinen kirjain isolla kirjaimella.
- Huomaa, että kukin komento toimii vain matematiikkatilassa.
- Lista kreikkalaisia aakkosia tuottavista komennoista löytyy esimerkiksi Kaijanahon kirjasta sivulta 133.

Esimerkki

Vanha kunnon π saadaan komennolla `\pi`, ja iso Π komennolla `\Pi`.
Alfa eli α on englanniksi *alpha*, eli se saadaan tuotettua komennolla `\alpha`.
Pikku-myy eli μ puolestaan saadaan komennolla `\mu`.

- Yläindeksi saadaan erikoismerkillä `^` ja alaindeksi erikoismerkillä `_`.
  Näin esimerkiksi `a^b` antaa a^b ja `a_b` antaa a_b.
- Myös nämä komennot toimivat vain matematiikkatilassa.
- Molemmat erikoismerkit vaikuttavat seuraavaan merkkiin tai komentoon. Mikäli ylä- tai alaindeksiin halutaan enemmän tavaraa, tulee käyttää aaltosulkuja:
  Esimerkiksi `a^\alpha` antaa a^α, mutta `a^bcd` nostaa vain b:n yläindeksiin. Jos kaikki kolme halutaan yläindeksiin, tulee kirjoittaa `a^{bcd}`.
- Ylä- ja alaindeksejä voi myös yhdistellä: `x_a^y` antaa x_a^y.
- Ylä- ja alaindeksimerkkejä käytetään myös esim. integraalien ja summien rajojen määräämisessä.

- Murtomerkinnät LaTeX tekee matematiikkatilan komennolla `\frac`, joka ottaa kaksi argumenttia – osoittajan ja nimittäjän. Esimerkiksi `\frac{1}{2}` latoo $\frac{1}{2}$.
- Juuret puolestaan saadaan komennolla `\sqrt`, joka ottaa pakolliseksi argumentikseen juurrettavan, ja valinnaiseksi argumentikseen juuren asteen.
  Esimerkiksi `\sqrt{2}` latoo $\sqrt{2}$ ja `\sqrt[n]{2}` latoo $\sqrt[n]{2}$.
- Molemmat merkinnät osaavat venyä fiksusti: esimerkiksi `\[ \sqrt[n]{\frac{a+b}{a-b}} \]` antaa

$$\sqrt[n]{\frac{a+b}{a-b}}$$

- Myöskin matematiikkatilassa kolmen pisteen latomiseen on oma komentonsa: komento `\dotsc` tuottaa kolme pistettä rivin alareunaan ja `\dotsb` rivin keskelle. Ensimmäistä käytetään esimerkiksi pilkulla erotettavissa listoissa ja jälkimmäistä summa- ja tulomerkintöjen kanssa.
  - Huom! Nyt `b` ei ole *bottom* eikä `c` ole *center*. Hämäävää!
- AMS-paketti `amsmath` tarjoaa komennon `\dots`, joka osaa haistella komennon jälkeen tulevasta merkistä tulevatko pisteet alas vai keskelle.
- Komento `\cdot` tuottaa yhden keskelle ladottavan pisteen, jota käytetään esimerkiksi kertomerkkinä.

- Myös matematiikkatilassa käytetään korostuksena erilaisia fontteja, ja lisäksi joissain matemaattisissa merkinnöissä tietty fonttityyli on vakiintunut, kuten lukujoukoissa $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$ ja $\mathbb{C}$.
- Normaali matematiikkafontti on tyypillisesti kursiivityylinen. Jotkin asiat on kuitenkin matematiikkatilassakin tapana latoa pystykirjaimin (esim. differentiaalien d). Tämä onnistuu komennolla `\mathrm`. Esim. `\mathrm{d}x`.
- Lihavoidun fontin saa komennolla `\mathbf`. Ainakin `eulervm`-matematiikkafontti tarjoaa myös lihavoidun kursiivifontin.
- Esimerkiksi Lagrangen funktiolle $\mathcal{L}$ käytetään tyypillisesti koukeroista kalligrafiafonttia. Tämä onnistuu komennolla `\mathcal`.
- Lukujoukkojen symboleissa käytetyn ”liitutaululihavoinnin” saa pakettien `amsfonts` tai `amssymb` kautta komennolla `\mathbb`.

- Myös matemaattisten symbolien kanssa käytetään erilaisia hipsuja ja pisteitä, kuten aikaderivaattaa kuvaava piste $\dot{a}$ tai vektoreissa käytetty nuoli $\vec{v}$.
- Piste jonkin symbolin päälle saadaan komennolla `\dot` ja kaksi pistettä komennolla `\ddot`. Molemmat komennot ottavat argumentikseen merkin, jonka päälle aksentti ladotaan.
- Vastaavasti toimivat `\bar` ja `\vec`, jotka tuottavat merkin päälle viivan ja pienen nuolen.
- Esimerkiksi operaattorien kanssa käytetty hattu saadaan komennolla `\hat`.
- Lista aksentointikomennoista löytyy esimerkiksi Kaijanahon kirjasta sivulta 101. Samat ja paljon lisää tarjoaa esimerkiksi *The Comprehensive LaTeX Symbol List* (`http://www.ctan.org/tex-archive/info/symbols/comprehensive/symbols-a4.pdf`).

Joitain tavallisen tekstin sanoja kuten *kun*, *ja* ym. on tapana käyttää matematiikkatilan sisällä. Tällaisia varten on matematiikkatilan komento `\text`, joka latoo argumenttinsa tavallisena tekstinä. Tavallisen tekstin ja matematiikan väliin kannattaa jättää reilusti tilaa, esimerkiksi komennolla `\quad`.

Esimerkki

```
\[
a - b = 0 \quad\text{kun}\quad a = b.
\]
```

Tuloste

$$a - b = 0 \quad \text{kun} \quad a = b.$$

Yleisimpiä matematiikkatilan komentoja

Esipuhe

Erilaisten matemaattisten konstruktioiden latomiseksi tarvitaan iso liuta uusia komentoja. Suunnattomia paineita ulkoaopettelusta ei kannata kuitenkaan ottaa, sillä komennot eksoottisten matemaattisten hieroglyfien tuottamiseksi ovat silkkaa taulukkotavaraa, joka opitaan vasta käytön myötä. Tärkeintä on luottaa siihen, että jos joku matemaatikko on jonkinlaisen venkuran tai hässäkän joskus paperille tehnyt, on sen esiinloihtimiseen myös oma LaTeX-komentonsa.

- Matematiikassa käytettäviä symboleita on liikaa luennolla lueteltaviksi – jotain merkkiä kaivatessaan voi silmäillä Kaijanahon kirjan sivuja 133–140 tai jo muutamaan kertaan mainittua teosta *The Comprehensive LaTeX Symbol List*.
- Symbolien löytämiseen on myös erinomainen web-työkalu `http://detexify.kirelabs.org`.
- Joitain fyysikkoja kiinnostavia merkkejä löytyy seuraavasta taulukosta:

$+$	`+`	$-$	`-`	$<$	`<`
$\hbar$	`\hbar`	$\ll$	`\ll`	$\neq$	`\neq`
∇	`\nabla`	$\gg$	`\gg`	$\sim$	`\sim`
∂	`\partial`	$\leq$	`\leq`	$\approx$	`\approx`
$\pm$	`\pm`	$\geq$	`\geq`	$\in$	`\in`
∞	`\infty`	$\propto$	`\propto`	$\varnothing$	`\varnothing`

- Matematiikan seassa esiintyy tiettyjä nimettyjä funktioita kuten sin ja cos, joiden nimet kirjoitetaan pystykirjaimilla. Nämä voisi tehdä käyttämällä aiemmin esiteltyä komentoa `\mathrm`, mutta useimmille on määritelty omat komentonsa – esimerkiksi `\sin` matematiikkatilassa latoo sin, ja `\log` latoo log.
- Lista näistä nimetyistä funktioista löytyy esimerkiksi Kaijanahon kirjasta sivulta 99.
- Paketti `amsmath` tarjoaa komennon `\DeclareMathOperator`, jolla voi määritellä uusia nimettyjä funktioita. Esimerkiksi komento `\DeclareMathOperator{\Tr}{Tr}` määrittelee uuden komennon `\Tr`, joka latoo matematiikkatilassa Tr pystykirjaimilla.

- Kaari- ja hakasulkeet saa käyttämällä koodissa normaalisti `(`- ja `[`-merkkejä. Aaltosulut saa tutuilla komennoilla `\{` ja `\}`.
- Suoran pystyviivan saa näppäimistöstäkin löytyvällä `|`-merkillä. Diracin notaatiossa käytetyt kulmasulkeet $\langle$ ja $\rangle$ saa komennoilla `\langle` ja `\rangle`. Merkit < ja > eivät ole kulmasulkeita!
- Suurten sulkumerkintöjen kanssa kannattaa käyttää komentoja `\left` ja `\right`, joiden molempien jälkeen kirjoitetaan haluttu sulkumerkki. Näin määritellyt sulut venyvät sisältönsä mukana.
  - Jos haluaa sulkumerkin vain toiselle puolelle, voi toisella puolella käyttää sulkumerkin paikalla pistettä.
- Jotta `\left` ja `\right` toimisivat myös kun välissä on rivinvaihtoja, lataa paketti `breqn`.

Muutamia esimerkkejä sulkeista. Kukin esimerkki on ladottu näyttömatematiikkatilassa.

Esimerkkejä

```
\langle\phi|\cdot|\psi\rangle =
\langle\phi|\psi\rangle
```

$$\langle\phi|\cdot|\psi\rangle = \langle\phi|\psi\rangle$$

```
\left(\frac{\partial^2 f}{\partial
x^2}\right)^2
```

$$\left(\frac{\partial^2 f}{\partial x^2}\right)^2$$

```
\chi_T = \left.\frac{\partial
M}{\partial H}\right|_T
```

$$\chi_T = \left.\frac{\partial M}{\partial H}\right|_T$$

```
\Delta \in \left]-\frac{1}{2},
\frac{1}{2}\right[
```

$$\Delta \in \left]-\frac{1}{2}, \frac{1}{2}\right[$$

$$y = \sin(x) + \underbrace{x \cdot \sin^2(\epsilon)}_{O(\epsilon^2)}$$

- Kun halutaan jollain tavalla ryhmitellä tai kommentoida jotain matemaattisen lausahduksen pätkää, käytetään usein merkinnän alle tai päälle ladottavaa suurta aaltosuljetta.
- Merkinnän päälle ladottava aaltosulje saadaan matematiikkatilan komennolla `\overbrace`, joka ottaa argumentikseen kaaren alle tulevan pätkän koodia. Jos kaaren päälle haluaa jonkin merkinnän, sen saa käyttämällä `\overbrace`-komennon jälkeen yläindeksiä.
- Vastaavasti toimii `\underbrace`, joka tuottaa merkinnän alle kaaren, ja jonka alaindeksi puolestaan ladotaan kaaren alle.

Esimerkin tuottanut koodi

```
y = \sin(x)+\underbrace{x\cdot\sin^2(\epsilon)}_{\mathcal{O}(\epsilon^2)}
```

- Myös muiden merkintöjen päälle ja alle voidaan latoa selventäviä merkintöjä. Paketissa `amsmath` tulevat komennot `\overset` ja `\underset` latovat ensimmäisen argumenttinsa toisen argumenttinsa päälle ja vastaavasti alle.
- Kommentoitujen nuolien tekemiseen sama paketti antaa komennot `\xleftarrow` ja `\xrightarrow`, jotka latovat pitkän nuolen, ja sen päälle komennolle annettavan pakollisen argumentin. Komennoille voi myös antaa valinnaisen argumentin, joka ladotaan nuolien alle.

Esimerkki

```
a + b \overset{a=b}{=} 2a
\xrightarrow{a\rightarrow 0} 0
```

Tuloste

$$a + b \overset{a=b}{=} 2a \xrightarrow{a\rightarrow 0} 0$$

- Tavallinen integraalimerkki saadaan komennolla `\int`. Integraalimerkin ylä- ja alaindeksit ladotaan integraalin rajoiksi.
- Integraalin rajat menevät näyttömatematiikkatilassakin oletuksena integraalimerkin vierelle. Päälle ja alle ne saa antamalla `amsmath`-paketille argumentin `intlimits`.
- Tekstimatematiikkatilassa integraalin rajat ladotaan integraalimerkin vierelle, tosin integraali on sen verran suuri merkintä, että se kannattaa yleensä kirjoittaa näyttömatematiikkatilassa.
- Kaksinkertaisen integraalin tuottaa `\iint`, kolminkertaisen `\iiint`, ja myös erikoisemmille integraalimerkeille on tietenkin omat komentonsa.

- Integrandin perään tyypillisesti merkitään integrointimuuttuja (tai -mitta) kirjoittamalla pieni d ja sen jälkeen integrointimuuttuja, eli esimerkiksi $\mathrm{d}x$. Standardien mukaan tämä d tulee kirjoittaa pystykirjaimin eli `\mathrm{d}`.
- Itse integrandin ja $\mathrm{d}x$-merkinnän väliin on hyvä latoa pieni väli komennolla `\,` kuin myös useamman muuttujan integraaleissa $\mathrm{d}x$-merkintöjen väliin.

Esimerkki

```
\int_{-\infty}^{\infty} e^{-x^2}\,
\mathrm{d}x = \sqrt{\pi}
```

Tuloste

$$\int_{-\infty}^{\infty} e^{-x^2}\,\mathrm{d}x = \sqrt{\pi}$$

Summamerkkinä toimiva iso sigma saadaan komennolla `\sum`, ja tulomerkki komennolla `\prod`. Molempien ylä- ja alaindeksit ladotaan merkkien päälle ja alle. Myös muille vastaavanlaisille merkinnöille löytyy omat komentonsa.

Esimerkejä

```
\sum_{i=0}^\infty a_i

\prod_{k=1}^n q_k

\mathcal{H} = \bigotimes_{i=1}^n \mathcal{H}_i
```

Tuloste

$$\sum_{i=0}^{\infty} a_i \qquad \prod_{k=1}^{n} q_k \qquad \mathcal{H} = \bigotimes_{i=1}^{n} \mathcal{H}_i$$

Myös erilaisten raja-arvojen tapauksessa on tapana latoa tekstiä suoraan merkinnän alle. Tämän vuoksi esimerkiksi raja-arvomerkin `\lim` alaindeksiksi merkitty pätkä menee näyttömatematiikkatilassa lim-merkinnän alle.

Esimerkki

```
\limsup_{n\to\infty} x_n = \lim_{n\to\infty}\sup_{m\geq n} x_n =
\inf_n \sup_{m\geq n}
```

Tuloste

$$\limsup_{n\to\infty} x_n = \lim_{n\to\infty} \sup_{m\geq n} x_m = \inf_n \sup_{m\geq n} x_m$$

- Matriisien latomiseen `amsmath`-paketti tarjoa omaperäisesti nimetyn `matrix`-ympäristön. Ympäristön sisällä matriisin alkiot luetellaan samoin kuin taulukoiden `tabular`-ympäristössä, eli sarakkeet erotetaan toisistaan &-merkeillä ja riviä vaihdetaan \\-komennolla.
- Sulut voi matriisin ympärille tehdä tavallisin keinoin, tai korvaamalla `matrix`-ympäristön `pmatrix`-ympäristöllä, joka latoo lukutaulukon ympärille automaattisesti kaarisulut, tai `bmatrix`-ympäristöllä, joka tekee hakasulut.
- Jos haluaa vaikuttaa sarakkeiden tasaukseen, voi käyttää ympäristöä `array`, joka toimii muuten kuten `tabular`, mutta latoo sisältönsä matematiikkatilassa.

Esimerkki

```
\left|\,\begin{matrix}
        1 & 2 \\
        3 & 4 \end{matrix}\,\right| = -2
```

Tuloste

$$\left|\begin{matrix} 1 & 2 \\ 3 & 4 \end{matrix}\right| = -2$$

Paloittaisille määrittelyille `amsmath`-paketti tarjoaa ympäristön `cases`. Ympäristön sisällä &-merkki määrää sarakkeenvaihtokohdan ja \\ vaihtaa riviä. Aaltosulku vasempaan reunaan tulee kaupan päälle.

Alla esimerkkinä signum-funktion määrittely.

Esimerkki

```
\DeclareMathOperator{\sgn}{sgn}
\sgn(x) = \begin{cases}
          -1 & \text{kun $x<0$} \\
          0  & \text{kun $x=0$} \\
          1  & \text{kun $x>0$}
\end{cases}
```

Tuloste

$$\operatorname{sgn}(x) = \begin{cases} -1 & \text{kun } x < 0 \\ 0 & \text{kun } x = 0 \\ 1 & \text{kun } x > 0 \end{cases}$$

Matematiikka dokumentin osana

Esipuhe

Selkeässä dokumentissa matematiikka – jos sitä käytetään – ei ole vain irrallisia yhtälönpätkiä tekstirivien välissä, vaan oleellinen osa kerrontaa. Kuvien ja taulukoiden tavoin keskeisimmät matemaattiset lausekkeet numeroidaan ja niihin viitataan. Seuraavassa kappaleessa opimme naittamaan matemaattiset lausekkeet osaksi kirjoitelman rakennetta siististi ja kätevästi. Lisäksi käsitellään hieman monirivisten lausekkeiden latomista.

- Numeroitu lauseke tehdään `equation`-ympäristöllä. Ympäristön sisältö ladotaan näyttömatematiikkatilassa – erillisiä `\[...\]` ei siis tarvita.
- Lausekkeelle voi antaa viittausavaimen käyttämällä tuttua `\label`-komentoa ympäristön sisällä.
- Nimestään huolimatta ympäristöä voi käyttää myös esim. epäyhtälöille ☺.
- Jokaista käyttämäänsä lauseketta ei tietenkään kannata numeroida – vain ne, joihin myöhemmin viittaa.

Esimerkki

```
\ldots eli kun $p = 0$,
\begin{equation}
  \label{eq:emc2}
  E = mc^2
\end{equation}
```

Tuloste

…eli kun $p = 0$,

$$E = mc^2 \tag{1}$$

- Kun lausekkeelle on annettu viittausavain `label`-komennolla, voi siihen viitata tutulla `ref`-komennolla, joka latoo yhtälön numeron.
- Koska lausekkeihin viitatessa lausekkeen numero merkitään usein sulkuihin, paketti `amsmath` tarjoaa komennon `\eqref`, joka toimii kuten `\ref`, mutta lisää sulut.

Esimerkki

```
Yhtälö~\eqref{eq:emc2} on ehkä ainut fysiikan yhtälö, joka on
löytänyt tiensä populaarikulttuuriin.
```

Tuloste

Yhtälö (1) on ehkä ainut fysiikan yhtälö, joka on löytänyt tiensä populaarikulttuuriin.

Ympäristöllä `equation` määriteltyjen lausekkeiden numerointiin voi vaikuttaa seuraavilla komennoilla:

- Komento `\notag` poistaa numeroinnin.
- Komennolla `\tag` voidaan lausekkeelle antaa numeron sijasta jokin muu symboli tai nimi. Haluttu nimi annetaan komennon pakolliseksi argumentiksi.
- `\tag`-komento lisää oletuksena sulut lausekkeelle annetun nimen ympärille. Tähdellinen versio `\tag*` ei näin tee.

Esimerkki

```
\begin{equation}
  a \geq \frac{A}{2} + 7
  \tag{$\heartsuit$}
  \label{eq:creepiness}
\end{equation}
```

Tuloste

$$a \geq \frac{A}{2} + 7 \tag{$\heartsuit$}$$

- Pelkkien numeroitujen lausekkeiden lisäksi matemattisen kirjoitelman osana saattaa olla myös numeroituja lauseita, lemmoja ja muita matemaattisen päättelyn peruspalikoita.
- Keinot tällaisten tyylikkääseen ladontaan ja käsittelyyn löytyvät AMS-LaTeXin paketista `amsthm`. Tämä paketti ei suoraan tarjoa ympäristöjä lauseiden ja korollaarien ladontaan, vaan tarjoaa komennon tällaisten ympäristöjen määrittelyyn.
- Fysiikan alan kirjoitelmissa numeroituja lauseita harvemmin näkee, mutta koska ainakin sivuaineissa fyysikkokin saattaa joutua niitä tekstiinsä latomaan, käsitellään niiden luominen tässä.

- Paketin `amsthm` tarjoama komento `\newtheorem` määrittelee uusia teoreemaympäristöjä.
- Komento ottaa kaksi pakollista argumenttia, joista ensimmäinen on ympäristölle annettava nimi (eli se mitä käytetään `\begin`-komennon argumenttina), ja toinen on ympäristön otsikon paikalle ladottava teksti, eli esimerkiksi ”`Lause`” tai ”`\textbf{Korollaari}`”.
- Komennosta on myös tähdellinen versio eli `\newtheorem*`, jolla määriteltyjä ympäristöjä ei numeroida.
- Kun ympäristö on määritelty, voidaan sitä käyttää normaalisti `\begin`- ja `\end`-komentojen kanssa. Ympäristö ottaa myös valinnaisen argumentin, joka ladotaan ympäristön otsikkoon ikään kuin lisähuomautukseksi, kuten ”Lause 5 (Gaussin lause)”.
- Sisäisiä viitteitä teoreemaympäristöihin voi tehdä tutuilla `\label`- ja `\ref`-komennoilla.

Esiteltyjen pakollisten argumenttien lisäksi komennolle voidaan antaa *korkeintaan toinen* seuraavista valinnaisista argumenteista:

- Jos valinnainen argumentti on pakollisten jälkeen (`\newtheorem{}{}[]`), ja jos argumentti on jokin otsikointitaso (`section`, `subsection`...), numeroidaan nyt määriteltävä ympäristö omalla numeroinnillaan aina kunkin tätä otsikkotasoa edustavan tekstin osan alla.
- Jos valinnainen argumentti on pakollisten argumenttien välissä (`\newtheorem{}[]{}`), ja jos argumentti on jonkin aiemmin määritellyn teoreemaympäristön nimi, juoksee nyt määriteltävän ympäristön numerointi aiemmin määritellyn ympäristön numeroinnin mukana.

Esimerkki

```
\newtheorem{lause}{Lause}[section]          ⟵lauseille oma numerointi joka sectionissa
\newtheorem{maar}[lause]{Määritelmä}        ⟵määritelmille sama numerointi kuin lauseille
```

- Paketti `amsthm` määrittelee kolme erilaista teoreemaympäristöjen tyyppiä, `plain`, `definition` ja `remark`, jotka kukin ladotaan hieman eri tavalla.
- Määriteltävän teoreemaympäristön tyyppiä voi vaihtaa käyttämällä komentoa `\theoremstyle` ennen `\newtheorem`-komentoa. Komento `\theoremstyle` ottaa argumentikseen yhden edellä mainituista kolmesta tyypistä.
- Nimiensä mukaisesti oletustyyli `plain` on tarkoitettu lauseille, lemmoille, korollaareille ym., `definition` määritelmille ja määritelmän kaltaisille ympäristöille ja `remark` huomioille ja huomautuksille.
- Jotta kaikki olisi mahdollisimman monimutkaista (mutta toisaalta yleistä), on myös komento `\newtheoremstyle`, johon ei nyt kuitenkaan mennä sen tarkemmin.

- Yhden ympäristön `amsthm` kuitenkin määrittelee valmiiksi, ja tämä on todistuksille tarkoitettu `proof`.
- Ympäristö ottaa valinnaisen argumentin, joka ladotaan todistuksen alkuun sanan ”Todistus” tilalle.
- Ympäristöä käyttämällä saa kaupan päälle todistuksen loppuun matemaatikoille tyydytystä tuovan neliön.

Esimerkki itse määritellystä teoreemaympäristöstä

Esimerkki

```
\newtheorem{lause}{Lause}[section]
\theoremstyle{definition}
\newtheorem{maar}[lause]{Määritelmä}

\begin{maar}
Luonnollinen luku on
\emph{alkuluku}, mikäli se...
\end{maar}
\begin{lause}
Alkulukuja on äärettömän monta.
\end{lause}
\begin{proof}
Oletetaan, että alkulukuja on
äärellinen määrä. Nyt luku, joka
saadaan kertomalla kaikki alkuluvut
keskenään ja lisäämällä yksi...
\end{proof}
```

Tuloste

Määritelmä 13.1. Luonnollinen luku on *alkuluku*, mikäli se on vähintään kaksi, ja se ei ole jaollinen muilla luonnollisilla luvuilla kuin itsellään ja luvulla yksi.

Lause 13.2. *Alkulukuja on äärettömän monta.*

Todistus. Oletetaan, että alkulukuja on äärellinen määrä. Nyt luku, joka saadaan kertomalla kaikki alkuluvut keskenään ja lisäämällä yksi, on suurempi kuin mikään alkuluku, mutta ei jaollinen millään niistä. Näin ollen se on joko uusi alkuluku tai jaollinen sellaisella, sillä jokainen luonnollinen luku on jaollinen jollain alkuluvulla. □

- Joskus matemaattinen lauseke ei mahdu yhdelle tai edes kahdelle riville (selkkareiden esimerkkisijoitukset…☹). Monirivisiä lausekkeita tarvitaan myös, kun jotkin lausekkeet kuuluvat kiinteästi samaan kokonaisuuteen (esim. Maxwellin yhtälöt).
- Tavallinen näyttömatematiikkatila ja `equation`-ympäristö eivät pidä rivinvaihdoista. AMS-LaTeX kuitenkin tarjoaa joukon ympäristöjä monirivisten lausekkeiden ja yhtälöryhmien latomiseen. Nämä ympäristöt saa käyttöönsä lataamalla paketin `amsmath`.
- Numerointiin vaikuttavat komennot kuten `\label` ja `\tag` toimivat näiden ympäristöjen kanssa normaalisti.

- Ympäristö `multline` on tarkoitettu sellaisten matemaattisten lausekkeiden latomiseen, jotka eivät mahdu yhdelle riville.
- Oleellisesti `multline` toimii kuten `equation`, mutta sallii rivinvaihdot `\\`-komennolla.
- Ympäristöstä on myös tähdellinen versio `multline*`, joka ei anna lausekkeelle numeroa.

Huom!

Huomaa, että ympäristön nimi ei ole `multiline` vaan `multline`!

- Myös ympäristö `split` on tarkoitettu monirivisten lausekkeiden latomiseen. Edellisestä poiketen `split`in avulla voidaan kuitenkin tasata esimerkiksi yhtäsuuruusmerkit samaan kohtaan.
- Tasauskohta merkitään &-merkillä, ja rivinvaihto `\\`-komennolla.
- Toisin kuin esimerkiksi `equation` ja `multline`, `split`-ympäristö ei itse aloita näyttömatematiikkatilaa, vaan se on itsessään laitettava näyttömatematiikkatilan sisään.

- Ympäristöt `gather` ja `align` on tarkoitettu useamman lausekkeen kokoelmien latomiseen. Ympäristöjen sisällä annetaan joukko `\\`-komennolla erotettuja lausekkeita, jotka LaTeX latoo omille riveilleen.
- Ympäristöjen erona on se, että `align` tarjoaa mahdollisuuden tasaukseen &-merkillä samoin kuin `split`.
- Molemmat ympäristöt numeroivat lausekkeet kuten `equation`. Molemmista ympäristöistä on kuitenkin tähdelliset versiot `gather*` ja `align*`, jotka jättävät numeroinnin pois.
- Kullekin lausekkeelle voi antaa oman viittausavaimensa käyttämällä `label`-komentoa kullakin rivillä.

Esimerkki monirivisestä lausekekokonaisuudesta

Esimerkki

```
\begin{align}
\vec{\nabla}\cdot\vec{E} &=
  \frac{\rho}{\epsilon_0}
  \label{eq:m1} \tag{\scshape Mi} \\
\vec{\nabla}\cdot\vec{B} &= 0
  \label{eq:m2} \tag{\scshape Mii} \\
\vec{\nabla}\times\vec{E} &=
  -\frac{\partial\vec{B}}{\partial t}
  \label{eq:m3} \tag{\scshape Miii} \\
\vec{\nabla}\times\vec{B} &=
  \mu_0\vec{J} + \mu_0\epsilon_0
  \frac{\partial\vec{E}}{\partial t}
  \label{eq:m4} \tag{\scshape Miv}
\end{align}
Yhtälöistä~\ref{eq:m3} ja~\ref{eq:m4}
saadaan sähkömagneettinen aaltoyhtälö.
```

Tuloste

$$\vec{\nabla}\cdot\vec{E} = \frac{\rho}{\epsilon_0} \qquad \text{(Mi)}$$

$$\vec{\nabla}\cdot\vec{B} = 0 \qquad \text{(Mii)}$$

$$\vec{\nabla}\times\vec{E} = -\frac{\partial\vec{B}}{\partial t} \qquad \text{(Miii)}$$

$$\vec{\nabla}\times\vec{B} = \mu_0\vec{J} + \mu_0\epsilon_0\frac{\partial\vec{E}}{\partial t} \qquad \text{(Miv)}$$

Yhtälöistä Miii ja Miv saadaan sähkömagneettinen aaltoyhtälö.

Osa V

Näyttäviä esityksiä LaTeXilla

Johdatus Beamer-esityksiin

Esipuhe

Beamer-dokumenttityyli on uudehko, esitysgrafiikan tekoon suunniteltu dokumenttityyli. Beameria käyttämällä LaTeXin hyvät ominaisuudet, kuten laadukas matematiikan ladonta, saadaan valjastettua esitysten tekoon.

- LaTeXin kyky tuottaa hyvin jäsenneltyjä ja ulkoasultaan huoliteltuja dokumentteja on valjastettu myös esityskalvojen eli *"powerpointtien"* tekoon.
  - Suullinen esitys on kuitenkin kirjallista dokumenttia huomattavasti vapaampi ja villimpi taiteenlaji, joten LaTeX ei ole aina aivan omillaan esityksiä tehdessä.
- Esimerkiksi kun gradu on tehty LaTeXilla, on kätevä tehdä myös graduseminaariesitys LaTeXilla, sillä tällöin voi gradusta siirtää suoraan käyttövalmiita yhtälöitä, taulukoita ym.
- Lisäksi LaTeXilla tehty esitys on vain tavallinen PDF-tiedosto, jonka näyttämiseen riittää mikä tahansa PDF-lukija.
- LaTeXilla voi esityskalvoja tehdä monin tavoin, mutta näistä Till Tantaun Beamer-dokumenttityyli on uusin ja monipuolisin.

- Sisällön tekeminen onnistuu hyvin pitkälti tutuilla LaTeX-komennoilla, mikä tekee Beamerista varsin helposti opittavan.
- Pelkän `beamer`-dokumenttityylin lisäksi Beamer sisältää valmiita dokumenttipohjia erilaisille ja erimittaisille esityksille, sekä joukon valmiita tyylejä, teemoja ja värimäärittelyitä joilla esityksestään saa helposti mieleisensä näköisen.
- Beamerin erinomaiseen käyttöoppaaseen kannattaa jokaisen ehdottomasti tutustua, sillä moni asia jää luennoilla käsittelemättä. Opas löytyy osoitteesta `http://tug.ctan.org/tex-archive/macros/latex/contrib/beamer/doc/beameruserguide.pdf`.

- Beamerin käyttö alkaa ottamalla käyttöön `beamer`-dokumenttityyli, eli aloittamalla koodinsa loitsulla `\documentclass{beamer}`.
- Jo dokumenttityylin lataaminen ottaa käyttöön useita paketteja, kuten `hyperref` ja `amsthm`. Muut tarvitsemansa paketit voi ladata normaalisti.
- Oletuksena Beamer käyttää kalvojen latomiseen pääteviivatonta (*sans serif*) fonttia. Fonttipaketin pääsee tietysti lataamaan itse.

- Koska valkokankaalle heijastettavan dokumentin kohdalla ei ole mielekästä puhua fyysisistä etäisyyksistä (kuinka varmistat kankaalle 20 mm korkuisen kuvan?), milleissä ja senteissä määritellyillä etäisyyksillä ei ole mitään merkitystä Beamerissa.
- Itse asiassa Beamerin ”paperikoko” on 128 mm × 96 mm. Tämän vuoksi Beamerin kanssa voi käyttää ”normaalin” kokoisia fontteja – 12 pisteen fontti näyttää kalvolla sopivan suurelta.
- Fonttikoon voi määrätä dokumenttityylin argumentilla kuten aikaisemmin. Esimerkiksi argumentti `10pt` lataa 10 pisteen fontin esityksille (kuten tämä), jossa kalvolle mahdutetaan aivan liian paljon tekstiä.

- Beamerissä kullekin kalvolle haluttava materiaali kirjoitetaan oman `frame`-ympäristönsä sisään.
- Hyvä esitys suunnitellaan kalvo kerrallaan. Tästä syystä Beamer ei myöskään oletuksena vaihda kalvoa automaattisesti, vaan käyttäjän on itse pidettävä huoli siitä, että `frame`-ympäristön sisään ladottava materiaali mahtuu kalvolle.
- Beamer latoo kunkin kalvon pystysuunnassa keskitettynä. Yläreunaan kalvon materiaalin voi tasata antamalla `frame`-ympäristölle valinnaisen argumentin `t`.
- Kalvon sisällön lisäksi Beamer lisää oletuksena kullekin kalvolle navigointielementit sekä informatiiviset ylä- ja alaviitteet. Paljaan kalvon saa antamalla `frame`-ympäristölle valinnaisen argumentin `plain`.

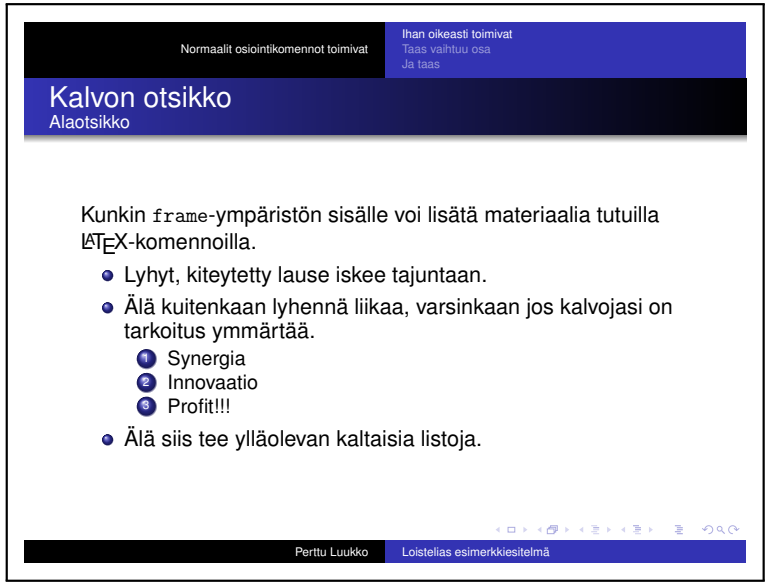
- Kullekin kalvolle saa otsikon käyttämällä komentoa `\frametitle` kalvon sisällä. Komento ottaa argumentikseen halutun otsikon. Vastaavasti alaotsikon saa komennolla `\framesubtitle`.
- `frame`-ympäristön sisälle voi kalvon sisällön luoda käyttäen tuttuja LaTeX-komentoja. Listat, yhtälöt, kuvat ym. toimivat kuten tähän asti on opittu.
- Komentojen ulkoasuun Beamer saattaa kyllä vaikuttaa tehdäkseen esimerkiksi listoista hieman pirteämmän näköisiä kuin artikkeliluokissa.

Beamer ja verbatim

Jos käytät `verbatim`-ympäristöä tai `\verb`-komentoa kalvon sisällä, sinun on kerrottava siitä `frame`-ympäristölle esimerkiksi valinnaisilla argumenteilla `containsverbatim` tai `fragile`.

Esimerkki

```
\begin{frame}
\frametitle{Kalvon otsikko}
\framesubtitle{Alaotsikko}
Kunkin \texttt{frame}-ympäristön sisälle voi
lisätä materiaalia tutuilla \LaTeX-komennoilla.
\begin{itemize}
  \item Lyhyt, kiteytetty lause iskee
tajuntaan.
  \item Älä kuitenkaan lyhennä liikaa,
varsinkaan jos kalvojasi on tarkoitus ymmärtää.
  \begin{enumerate}
    \item Synergia
    \item Innovaatio
    \item Profit!!!
  \end{enumerate}
  \item Älä siis tee ylläolevan kaltaisia
listoja.
\end{itemize}
\end{frame}
```



Kalvon otsikko
Alaotsikko

Kunkin `frame`-ympäristön sisälle voi lisätä materiaalia tutuilla LaTeX-komennoilla.

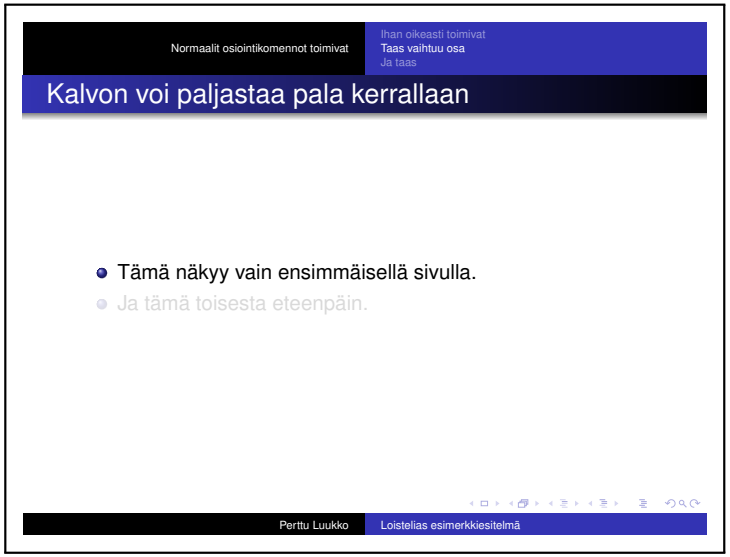
- Lyhyt, kiteytetty lause iskee tajuntaan.
- Älä kuitenkaan lyhennä liikaa, varsinkaan jos kalvojasi on tarkoitus ymmärtää.
  1. Synergia
  2. Innovaatio
  3. Profit!!!
- Älä siis tee ylläolevan kaltaisia listoja.

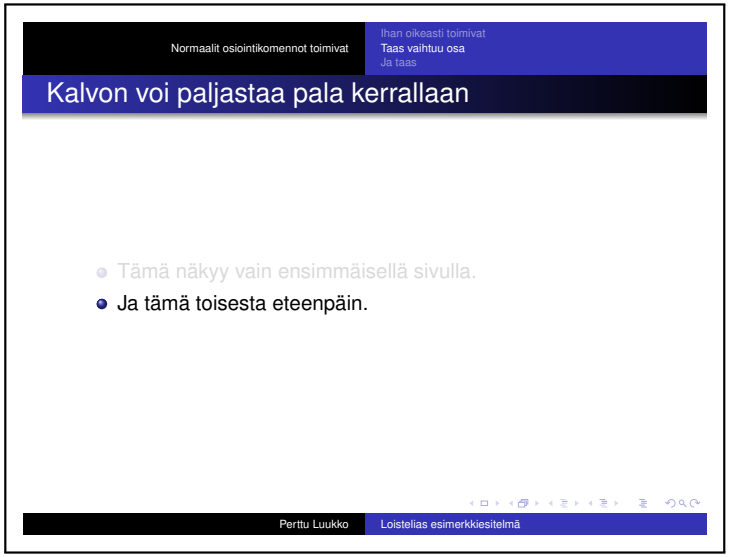
- Tässäkin esityksessä paljon käytetty efekti on kalvon paljastaminen osa kerrallaan. Beamerissa tällaiset efektit ovat varsin helppoja. Lopullisessa PDF-tiedostossa tämä efekti näkyy siten, että kutakin kalvoa vastaa useampi sivu, joita selatessa kalvon eri osat paljastuvat.
- Beamer määrittelee uudelleen monia LaTeXin komentoja ottamaan uudenlaisen *overlay*-argumentin, joka kirjoitetaan <>-merkkien sisään. Argumentti kertoo Beamerille (muun muassa), millä tämän kalvon sivuilla komennon tulostus näytetään.
- Kalvon voi myös paljastaa pala kerrallaan yksinkertaisemmin käyttämällä komentoa `\pause` palojen välissä.

Esimerkki kalvon paljastamisesta pala kerrallaan

Esimerkki

```
\begin{frame}
\frametitle{Kalvon voi paljastaa pala
kerrallaan}
\begin{itemize}
  \item<1> Tämä näkyy vain ensimmäisellä
    sivulla.
  \item<2-> Ja tämä toisesta eteenpäin.
\end{itemize}
\end{frame}
```





- Overlay-argumentilla voi siis määrätä millä kalvon sivuilla komennon vaikutus näkyy. Argumentti voi olla numero tai pilkulla erotettu lista numeroita. Lisäksi ”johonkin sivuun asti” ja ”jostain sivusta eteenpäin” voidaan ilmaista viivalla.
- Se, millä tavalla ”piilotetut” sivut näkyvät, voidaan määrätä komennolla `\setbeamercovered`. Komento ottaa esimerkiksi seuraavat argumentit:

  `invisible` Piilotetut osat näkymättömiä (oletus)
  `transparent` Piilotetut osat läpinäkyviä (kuten tässä)

Lista-alkio, joka näkyy sivuilla 1, 3 ja sivusta 5 eteenpäin.

```
\item<1,3,5-> Vompatti
```

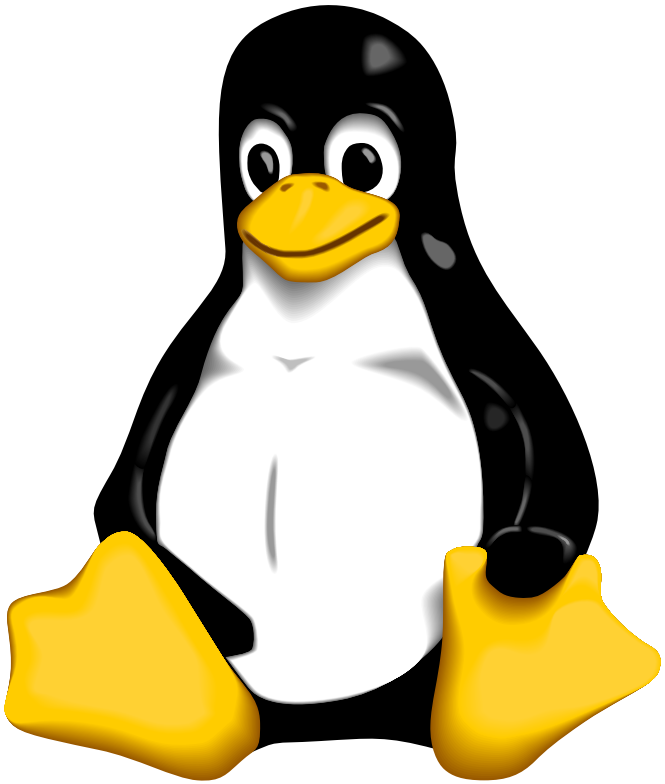
- Kalvoesityksissä saa (tiettyyn rajaan asti) leikkiä väreillä melko räikeästi. Muutenkin räikeässä taustassa tavanomaiset korostuskeinot, kuten `\emph` tai `\textbf` eivät kuitenkaan välttämättä korostu riittävästi.
- Tämän vuoksi Beamer määrittelee uuden korostuksen, `\alert`, joka tyypillisesti latoo sisältönsä kirkuvan punaisella. Komento ottaa argumentikseen korostettavan tekstin kuten `\emph`.
- Komento ottaa myös overlay-argumentin, jolla voidaan korostaa teksti esimerkiksi vain tietyllä sivulla:
  `\alert<3>{Tämä korostetaan vain kalvolla 3.}`
- Tietyllä kalvon sivulla korostaminen voidaan myös antaa overlay-argumenttina tyyliin
  `\item<alert@4> Tekstiä.`

Viisauden murunen

Usein on hyvä latoa jokin esityksen keskeinen sanoma värilliseen laatikkoon, jota yleisö voi pysähtyä meditoimaan. Esimerkiksi tämä laatikko saatiin Beamerin `block`-ympäristöllä, joka ottaa pakolliseksi argumentikseen laatikon otsikon.

- Valmiita värillisiä laatikoita sisällön ryhmittelyyn tarjoavat Beamerin ympäristöt `block`, `exampleblock` ja `alertblock`. Kukin laatikkoympäristö ottaa pakolliseksi argumentikseen laatikon otsikon.
- Kullekin laatikolle voi myös antaa overlay-määreen, jolloin laatikon saa esimerkiksi näkymään vain tietyllä kalvon sivulla.
- Mikäli haluaa lisää ja erikoisempia laatikoita, kannattaa katsastaa Beamerin käyttöoppaan osa *Framed and Boxed Text*.

- Kalvojen jako palstoihin tulee kysymykseen erityisesti silloin, kun haluaa liittää samalle kalvolle sekä kuvan että sanallista selitystä kuvan sisällöstä.
- Beamerissa kalvon jako palstoihin onnistuu ympäristöllä `columns`, jonka sisälle kirjoitetaan kutakin palstaa kohden `column`-ympäristö.
- `column`-ympäristö ottaa pakolliseksi argumentikseen palstan leveyden.
- Myös palstajoukon leveyden voi asettaa antamalla `columns`-ympäristölle valinnainen argumentti `totalwidth=`*`leveys`*. Oletuksena `columns` täyttää koko kalvon. Argumentti `totalwidth=\textwidth` voidaan lyhyesti ilmaista `onlytextwidth`.



- Vasemmalla näkyy muuan pingviini, jota kaikki rakastavat.
- Kuva kannattaa yleensä laittaa vasemmalle palstalle, sillä kulttuureissa, joissa luetaan vasemmalta oikealle, huomio kiinnittyy ensin vasempaan puoleen.

Esimerkki palstoituksesta, kuinka se tehtiin

Esimerkin tuottanut koodi

```
\begin{columns}[c,onlytextwidth]
\begin{column}{0.39\textwidth}
  \includegraphics[width=12em]{Tux}
\end{column}
\begin{column}{0.58\textwidth}
  \begin{itemize}
    \item Vasemmalla näkyy muuan pingviini, jota...
    \item Kuva kannattaa yleensä laittaa vasemmalle...
  \end{itemize}
\end{column}
\end{columns}
```

- Palstojen sisältö keskitetään oletuksena pystysuunnassa. Tähän voi vaikuttaa `columns`-ympäristön valinnaisella argumentilla:
  - `c` keskittää.
  - `b` tasaa alareunat.
  - `t` tasaa ylimpien tekstirivien pohjat.
  - `T` tasaa yläreunat.
- Huomaa ero `t`- ja `T`-kirjainten välillä! Tekstin tapauksessa rivien pohjien tasaaminen on hyvä idea, mutta kuvien kanssa on parempi suoraan tasata kuvien yläreunat.
- Tasauksen voi määrätä myös palstakohtaisesti antamalla samat argumentit `column`-ympäristölle.

Esityksen rakentaminen

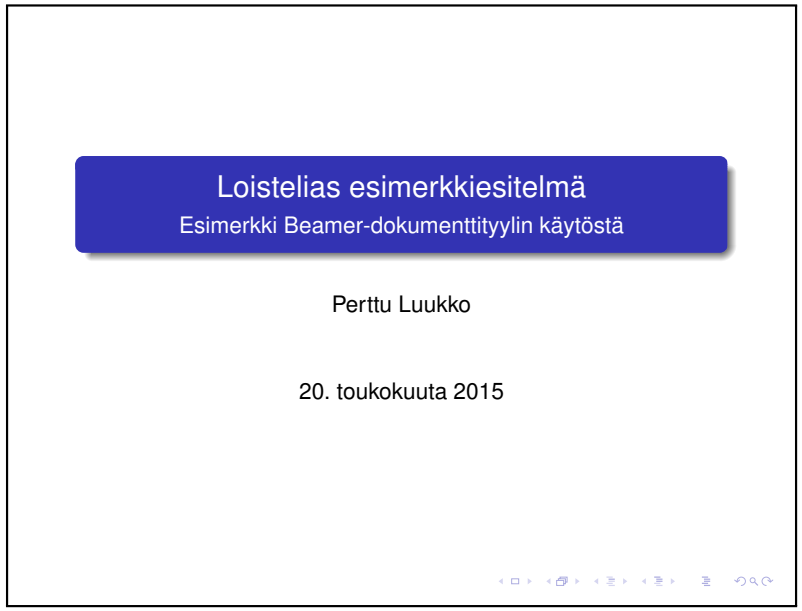
Esipuhe

Seuraavassa kappaleessa tutustumme kokonaisen esityksen – etusivuista liitteisiin – rakentamiseen Beamerilla. Samalla käsitellään hieman yleisiä ohjeita, jotka kannattaa pitää mielessä esitysgrafiikkaa laatiessaan.

- Muiden dokumenttityylien tavoin Beamer osaa itse rakentaa esityksen etusivun käyttäjän tarjoamista tiedoista.
- Esityksen tiedot voi kertoa seuraavilla alustuksessa käytettävillä komennoilla:
  - `\author` kertoo tekijän,
  - `\title` otsikon,
  - `\subtitle` alaotsikon (jos sellainen on),
  - `\date` ajan tai muun ajankohdan (esim. Ortotopologian päivät 2015),
  - `\institute` tekijän laitoksen tai muun instituutin,
  - `\titlegraphic` etusivussa käytettävän kuvan. Tämän komennon argumentiksi annetaan siis sopiva `\includegraphics`-komento.
- Tämän jälkeen komento `\titlepage` jonkin kalvon sisällä käytettynä latoo valmiin etusivun.

Esimerkkietusivu

```
\author{Perttu Luukko}
\title{Loistelias
esimerkkiesitelmä}
\subtitle{Esimerkki
Beamer-dokumenttityylin käytöstä}

\begin{document}

\begin{frame}[plain]
  \titlepage
\end{frame}
```



- Beamerissä esitys pilkotaan kappaleisiin tutuilla `\section`- ja `\subsection`-komennoilla.
- Kappaleiden ja alakappaleiden loogiseen etenemiseen kannattaa kiinnittää erityistä huomiota esitysten tapauksessa, sillä esityksessä yleisön tulee ymmärtää asia mieluiten kertalukemalla.
- Kappaleita ei kannata käyttää liikaa. Hyvä jako on esimerkiksi 2–4 alakappaletta per kappale, ja korkeintaan 2–4 kappaletta per esitys.
- Jos esitys on hyvin pitkä, voi esityksen jakaa lisäksi `\part`-otsikointikomennoilla pienempiin osiin. Kullekin osalle saa oman etusivun komennolla `\partpage`. Kukin osa käyttäytyy hyvin itsenäisesti. Esimerkiksi niillä on omat sisällysluettelot.

- Sisällysluettelot esityksessä tarjoavat esittäjälle mahdollisuuden kertoa lyhyesti mitä tuleman pitää. Näin yleisö on alusta lähtien selvillä, mihin puhuja pyrkii. Tässäkin esityksessä olette nähneet sisällysluettelosivuja aina uuden luvun alussa.
  - Tästä tulostusversiosta nämä on tarpeettomina poistettu.
- Kuten aiemmin, sisällysluettelo tuotetaan komennolla `\tableofcontents`. Beamerin sisällysluettelokomento ottaa kuitenkin valinnaisia argumentteja, joista hyödyllisimmät alla:
  - `currentsection`: Vain nykyisen kappaleen sisällysluettelo esitetään, ja muut näytetään läpinäkyvinä.
  - `currentsubsection`: Sama alakappaleelle.
  - `pausesections`: Sisällysluettelo näytetään kappale kerrallaan.
  - `pausesubsections`: Sama alakappaleille.

- Kuten moni varmasti taas arvaa, sisällysluettelot kunkin kappaleen alkuun saa kätevästi ja automaattisesti.
- Automaattiset sisällysluettelot ja muut temput saa aikaan kätevästi Beamerin komennoilla `\AtBeginSection`, `\AtBeginSubsection` ja `\AtBeginPart`, jotka latovat ainoan pakollisen argumenttinsa aina jokaisen kappaleen, alakappaleen tai osan alussa.

Kappaleen sisällysluettelo aina kappaleen alussa

```
\AtBeginSection{
\begin{frame}
  \frametitle{Sisällys}
  \tableofcontents[currentsection]
\end{frame}
}
```

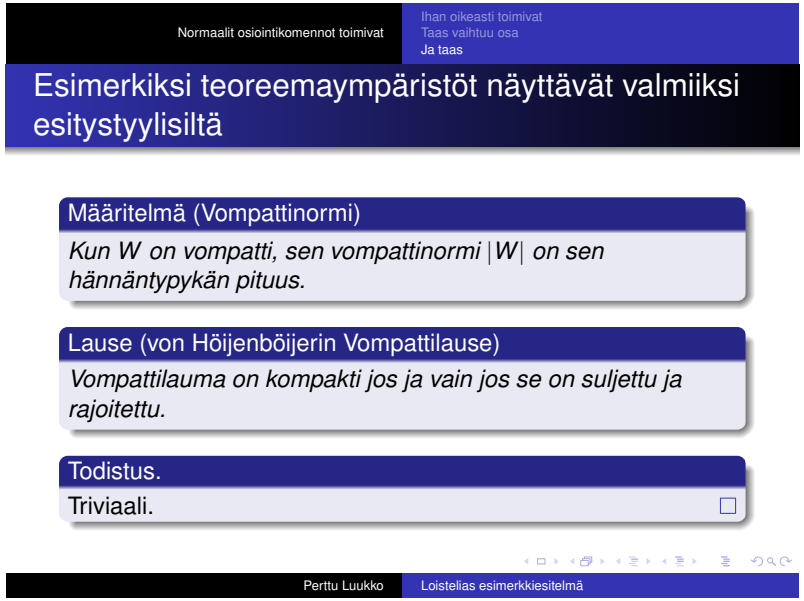
- Beamer-esityksiin saa kuvia ja taulukoita tutuilla menetelmillä.
- Artikkelityyleistä poiketen `figure`- ja `table`-ympäristöillä määriteltyjä kuvia ja taulukoita ei esityksissä (tietenkään) sijoitella automaattisesti, vaan kuva tai taulukko tulee välittömästi siihen paikkaan missä kyseinen ympäristö koodissa alkaa.
- Esityksessä ei ole yleensä mielekästä numeroida kuvia, ellei yleisöllä ole käytettävissään esimerkiksi tulostettavaa versiota esityksestä. Tällöin numeroinnin saa komennolla `\setbeamertemplate{caption}[numbered]`.

- Myös matematiikkaa voi esityksiinsä lisätä tutuilla komennoilla.
- Kannattaa muistaa, että `beamer`-dokumenttityyli lataa automaattisesti esimerkiksi `amsthm`-paketin.

Varoitus

Matematiikkaa kannattaa esityksissä käyttää varovasti! Suuret kaavahirviöt – varsinkin liian nopeasti väläytettyinä – tiputtavat yleisön kärryiltä nopeasti, tehokkaasti ja lopullisesti.

- Beamerin kanssa käytettynä paketin `amsthm` teoreemaympäristöt näyttävät valmiiksi ”esitystyylisiltä”.
- Beamer määrittelee valmiiksi ympäristöt `definition` ja `theorem`, mutta lisää saa määriteltyä tutulla `\newtheorem`-komennolla.



- Esityksissä harvemmin esitellään yksityiskohtaisia kirjallisuusluetteloita, mutta joskus voi olla hyödyllistä viitata muutamaan teokseen esimerkiksi tarjotakseen yleisölle lisälukemista aiheesta.
- Kirjallisuusluettelon voi toteuttaa samalla `thebibliography`-ympäristöllä kuin aiemminkin. Joka tapauksessa kirjallisuusluettelo kannattaa pitää lyhyenä, joten BibTeXille ei pitäisi tulla tarvetta.
- Beamer tekee myös kirjallisuusluettelosta valmiiksi esitystyylisen. Kirjallisuusluettelon ulkoasun viilaamisesta kannattaa lukea lisää Beamerin käyttöoppaasta.


Till Tantau. *User's Guide to the Beamer Class, Version 3.06*.


Antti-Juhani Kaijanaho.
LaTeX ja AMS-LaTeX. Opus asiatekstin ladonnasta.
Jyväskylän yliopiston ATK-keskus.
Toinen laitos, 2003.

Esimerkkikirjallisuusluettelon tuottanut koodi

```
\begin{thebibliography}{beamer}
\bibitem{beamer} Till Tantau. \textit{User's Guide to the Beamer Class,
Version 3.06}.
\end{thebibliography}
\setbeamertemplate{bibliography item}[book]
\begin{thebibliography}{kaijanaho}
\bibitem{kaijanaho} Antti-Juhani Kaijanaho.
  \newblock \textit{\LaTeX\ja \AmS-\LaTeX. Opus asiatekstin ladonnasta}.
  \newblock Jyväskylän yliopiston ATK-keskus.
  \newblock Toinen laitos, 2003.
\end{thebibliography}
```

- Oma liiteosio esityksessä on hyvin varautuneen puhujan hätävara. Liiteosioon on hyvä lisätä tavaraa, jota itse esityksessä ei todennäköisesti ehdi käymään läpi, mutta josta voi olla hyötyä esimerkiksi esityksen jälkeisiin kysymyksiin vastatessa.
- Samalla jos muu esitys tulee käytyä läpi odottamattoman nopeasti, voi sulavasti ryhtyä käymään läpi jotain liitteistä löytyvää lausekkeen johtoa, joka itse puheessa tuli ohitettua tarpeettoman nopeasti.
- Tällainen jako on huomattavasti parempi kuin se tyypillinen strategia, jossa esitysaikaa tuhlataan kaikenlaiseen vähemmän oleelliseen nippelitietoon, minkä seurauksena esityksen pääasia ohitetaan ajanpuutteen vuoksi.
- Beamerissa liiteosa erotetaan muusta esityksestä komennolla `\appendix`. Niin helposti se käy.

- Seuraavat ohjeet on koottu Beamerin käyttöoppaasta (kappale 5) löytyvistä viisauden murusista, jotka Till Tantau on koonnut esitystä suunnitteleville.
- Ensinnäkin: **Tiedosta aikarajat.** Älä yritä mahduttaa esitykseesi enempää materiaalia kuin mitä ehdit kiirehtimättä käsitellä.
  - Jättämällä epäoleelliset yksityiskohdat pois esityksestäsi saat kuitenkin pääkohdat välitettyä yleisölle.
  - Kiirehtimällä niin pääkohdat kuin yksityiskohdatkin jäävät yleisöltä hämärän peittoon.
- Esimerkiksi tämä esitys etenee nopeudella, joka on karkeasti 0,7 kalvoa per minuutti.

- Jaa esityksesi kappaleisiin ja alakappaleisiin mahdollisimman loogisesti siten, että esityksen sisällysluettelo on kuin tukiranka, joka kannattelee muuta esitystä.
- Ideaalitapauksessa sisällysluettelon perusteella voidaan ymmärtää esityksesi pääkohdat.
- Aloita esityksesi kertomalla mitä esityksesi käsittelee. Näin yleisö on mukana alusta lähtien, eivätkä kuulijat joudu käyttämään ensimmäistä kymmentä minuuttia päätelläkseen, mistä oikein on kyse.

- Esityksessä kannattaa tehdä mieluummin paljon, vähän asiaa sisältäviä kalvoja kuin päinvastoin.
  - Tätä sääntöä tämä esitys rikkoo brutaalisti.
- Tantau suosittelee noin 20–80 sanaa per kalvo.
- Älä kikkaile monimutkaisilla lauseilla tai hirviömäisillä matemaattisilla lausekkeilla. Yleisön on ymmärrettävä kalvo siinä ajassa, kun se on näkyvillä.
- Älä oleta, että jokainen yleisössäsi on alan ekspertti. Sen sijaan oleta Tietämättömän Yleisön Laki:

Tietämättömän Yleisön Laki

Joku yleisössä tietää vähemmän kuin mitä oletat kaikkien tietävän, vaikka ottaisit Tietämättömän Yleisön Lain huomioon.

Tyyliä & efektejä

Esipuhe

Beamerissä tyylikkään esityksen saa helpoimmillaan vain lataamalla miellyttävän teeman. Jos valmiit teemat eivät miellytä, Beamer tarjoaa kuitenkin myös mahdollisuuden yksityiskohtaisempaan säätämiseen. Tyyliseikkojen lisäksi opimme tässä kappaleessa myös esimerkiksi luomaan esityksestä tulostettavan version helposti ja näppärästi.

- Beamer sisältää suuren joukon valmiita teemoja, joilla esityksen ulkoasua voi helposti säätää. Halutessaan esityksen tyyliä voi myös viilata käsin, mikä on tehty yllättävän helpoksi.
- Beamerin teemoja on viittä eri mallia:
  - Esitysteema (*presentation theme*) muokkaa koko esityksen ulkonäköä.
  - Väriteema (*color theme*) muokkaa esityksen värejä.
  - Fonttiteema (*font theme*) muokkaa fontteja.
  - Sisäteema (*inner theme*) muokkaa kalvon *sisällä* olevia elementtejä, kuten listoja, teoreemoja ym.
  - Ulkoteema (*outer theme*) muokkaa kalvojen ulkonäköä, eli esimerkiksi ala- ja yläviitteitä ja navigointielementtejä.
- Näin laiskempi esityksenpitäjä voi vain lätkäistä esitykseen mieleisensä esitysteeman, mutta hieman tarkempi voi yhdistellä mieleisensä ulkoasun sopivista väri- fontti- sisä- ja ulkoteemoista.

- Teemat otetaan käyttöön komennoilla `\usetheme`, `\usecolortheme` jne. Kukin komento ottaa pakolliseksi argumentikseen teeman nimen.
- Esimerkkejä eri teemoista kannattaa etsiskellä Beamerin käyttöoppaasta.
- Kerrottakoon, että esimerkiksi tämän esityksen luentoversio käyttää *Luebeck*-esitysteemaa, jonka päälle on lätkäisty *rose*- ja *seahorse*-väriteemat. Myös jotain omia viilauksia on tietysti myös tehty.
- Tulosteversio puolestaan käyttää selkeää *Rochester*-esitysteemaa harmaasävyisillä *dove*-väriteeman väreillä.

- Erilaiset manuaaliset ulkoasun säädöt on Beamerissa tehty pirullisen helpoiksi. Jokaista Beamerin elementtiä kuten ”`block`-ympäristön teksti” tai ”ala- tai yläviitteessä näkyvä tekijän nimi” vastaa oma fontti- ja värimääritys, jotka on helppo ohittaa.
- Vastaavasti monia elementtejä, kuten ”kirjallisuusviite” tai ”alaviite”, vastaa jokin koodipätkä, jolla Beamer latoo kyseisen elementin. Myös nämä koodipätkät voi korvata mieleisillään.
- Tällaisia manuaalisia säätöjä tehdessään on tietysti hyvä varmistaa, että itse esitys on sisällöltään jo kunnossa.

- Värejä voi määritellä uudelleen komennolla `\setbeamercolor`. Komento ottaa ensimmäiseksi argumentikseen Beamerin väripohjan nimen, kuten `block body` tai `author in head/foot`, ja toiseksi argumentikseen värimääreen.
- Väripohjan nimen löytää helpoiten Beamerin käyttöohjeesta.
- Värimääre koostuu pilkuilla erotetuista osista `fg=`*`väri`* ja `bg=`*`väri`*, joista ensimmäinen asettaa tekstin värin ja toinen taustan värin.
- Värit noudattavat paketin `xcolor` syntaksia. Esimerkiksi `black!3` sekoittaa 3% mustaa ja loput valkoista, ja `green!40!yellow` sekoittaa 40% vihreää ja loput keltaista.

Esimerkki: tulostusversiota varten haalea harmaa tausta kalvolle

```
\setbeamercolor{background canvas}{bg=black!3}
```

- Fonttien määrittäminen käsin tapahtuu samoin kuin värien tapauksessa. Komento tähän on `\setbeamerfont`, joka ottaa ensimmäiseksi argumentikseen Beamerin fonttipohjan nimen, kuten taaskin `block body` tai `author in head/foot`, ja toiseksi argumentikseen fonttimääreen.
- Fonttimääre koostuu pilkuilla erotetuista osista `size=`*`koko`*, `shape=`*`muoto`*, `series=`*`sarja`* ja `family=`*`perhe`*, jotka kukin säätävät fontin eri ominaisuuksia.
- Fonttimääreessä
  - *`koko`* on LaTeXin koonvaihtokomento kuten `\small` tai `\large`,
  - *`muoto`* on fontinvaihtokomento kuten `\scshape`,
  - *`sarja`* on fontinvaihtokomento kuten `\bfseries`
  - ja *`perhe`* on fontinvaihtokomento kuten `\ttfamily`.

- Tehtävä: Halutaan `alertblock`-laatikon otsikko lihavoidulla, pienellä fontilla.
- Beamer-manuaalin kappaleesta *"Block Environments"* löytyy `alertblock`-ympäristön kohdalta teksti

  Beamer-Color/-Font `block title alerted`

- Fonttipohjan nimi on siis `block title alerted`.
- Pieneen fonttiin vaihtava komento on `\small`, ja lihavointiin vaihtava komento on `\bfseries`.

`alertblock`-laatikon otsikko lihavoidulla, pienellä fontilla

```
\setbeamerfont{block title alerted}{size=\small,series=\bfseries}
```

- Ainakin listaympäristöt kuten `itemize`, `enumerate` ja `description` ottavat valinnaisen argumentin, joka annetaan jokaiselle `\item`-alkiolle overlay-argumentiksi. Argumentissa oleva plus-merkki eli + korvataan juoksevalla numeroinnilla.
- Näin esimerkiksi `\begin{itemize}[<+->]` aloittaa listan, jonka alkiot paljastetaan yksi kerrallaan.
- Pienellä kekseliäisyydellä näin voi helposti tehdä esimerkiksi listan, jonka alkiot paljastetaan yksi kerrallaan, ja korostetaan sillä sivulla jolla ne ilmestyvät esiin:
  `\begin{itemize}[<+-|alert@+>]`

- Aiemmin esiteltyjen lista-alkioiden ja laatikoiden lisäksi myös monet muut komennot ottavat overlay-määreitä.
- Tällaisia ovat esimerkiksi teoreemaympäristöt sekä suurin osa fontinvaihtokomentoja.
- Muut komennot saa helposti vain tietylle kalvolle esimerkiksi seuraavilla komennoilla:
  - `\only` ottaa overlay-määreen sekä pakollisen argumentin, joka ladotaan vain overlay-määreen määräämillä sivuilla. Esimerkiksi `\only<4>{\includegraphics{Tux}}` väläyttää Tuxia vain sivulla neljä.
  - `\visible` toimii samoin, mutta pakollisen argumentin tuottama tulostus vain piilotetaan muilta sivuilta. Myös `\invisible` löytyy.
  - `\uncover` toimii samoin, mutta muilla sivuilla piilotettu sisältö näkyy läpinäkyvänä.

- Edellisten komentojen kaltaisia, elementin näkyvyyteen vaikuttavia ohjeita voi myös käyttää overlay-argumentteina. Tällaisia argumentteja ovat `alert`, `uncover`, `only`, `visible` ja `invisible`.
- Argumentin saa voimaan vain tietyillä sivuilla lisäämällä argumentin perään `@`-merkin, jota seuraa sivu tai joukko sivuja.
- Esimerkkejä:
  - Lista-alkio, joka näkyy sivusta 3 eteenpäin, ja korostetaan kalvolla 3:
    `\item<3-|alert@3>`
  - Lista-alkio, joka lisätään listaan vain sivuissa 2 ja 3, ja sivussa 3 se on näkymätön:
    `\item<only@2-3|invisible@3>`
  - Lista-alkio, joka on näkyvä sivuilla 1, 3 ja 5 sekä näkymätön sivuilla 2 ja 4:
    `\item<visible@1,3,5|invisible@2,4>`

- Jos edellisen kalvon kikkoja haluaa soveltaa komentoihin, jotka eivät sellaisenaan ota overlay-argumentteja, voi käyttää komentoa `\action`, joka ottaa overlay-argumentin, sekä pakollisena argumenttina koodipätkän, johon se sitä soveltaa.
- Jos haluaa esitykseensä todella monimutkaisia kikkailuja, kuten yhtälöryhmän paljastaminen pala kerrallaan tai taulukon paljastaminen rivi tai sarake kerrallaan, kannattaa turvautua Beamerin käyttöoppaan kappaleeseen *"How To Uncover Things Piecewise"*.

- Esityksen tulostettavan version luonti onnistuu niinkin yksinkertaisesti kuin antamalla `beamer`-dokumenttityylille valinnainen argumentti `handout`. Tällöin Beamer automaattisesti jättää pois navigointielementit ja muut tulosteessa turhat osat.
- Beamer osaa myös enemmän: itse asiassa `handout`-tyyppiä voi käyttää myös overlay-argumenttina! Näin esimerkiksi `\item<handout>` aloittaa lista-alkion, joka löytyy vain tulostusversiosta, ja komennolla `\only<handout>{...}` voi ladata esimerkiksi tulosteversiolle oman, harmaasävyisen väriteeman.

- Tulosteversiossa halutaan usein myös pakata useampi kalvo samalle tulosteen sivulle, kuten näiden luentomuistiinpanojen tulostettavassa versiossa on pakattu neljä kalvoa per sivu.
- Tämä onnistuu paketilla `pgfpages`. Paketti määrittelee alustuksessa käytettävän komennon `\pgfpagesuselayout`, jonka käyttö selvinnee seuraavista esimerkeistä:

Kaksi kalvoa per sivu, jättäen 5 mm tyhjää kalvojen väliin

```
\pgfpagesuselayout{2 on 1}[a4paper,border shrink=5mm]
```

Neljä kalvoa per sivu

```
\pgfpagesuselayout{4 on 1}[a4paper,landscape,border shrink=5mm]
```

- Beamer määrittelee `handout`-tyylin lisäksi tyylin `trans`, joka on tarkoitettu oikeille, muoviliuskoille tulostettaville kalvoille.
  - Vainoharhainen puhuja voi tehdä esityksestään tällaisetkin, käytettäväksi jos eli *kun* tietotekniikka pettää…
- Lisäksi löytyy tyyli `article`, joka tuottaa hieman artikkelityyliä jäljittelevän ulkoasun, jossa kalvojen rajat jätetään huomiotta ja kaikki sisältö ladotaan yhteen pötköön.
- Molemmat näistä tyyleistä toimivat kuten `handout`, eli ne saa käyttöönsä antamalla `beamer`-dokumenttityylille argumentin `trans` tai `article`. Molemmat toimivat myös overlay-määreinä.
- Beamerin oletustyyli eli esitysmuotoinen tyyli on nimeltään `beamer`. Myös tätä voi käyttää overlay-argumenttina.

- Hyperlinkeillä saa kätevästi rakennettua joustavan esitelmän.
  - Esimerkiksi jos esityksesi jossain kohdassa on syytä palata taaksepäin, voit selaamisen sijasta vain klikata sopivaa nappia, joka vie suoraan oikealle sivulle.
  - Tai esimerkiksi ennen pitkää todistusta voisi esityksessäsi olla nappula ”Ohita todistus”, jota painamalla todistus sivuutetaan.
- Hyperlinkkien kohteita voit tehdä tutulla `\label`-komennolla. Muista käyttää overlay-määrettä tämän komennon kanssa, mikäli kalvo koostuu useammasta sivusta, jotta `\label` osoittaa vain yhteen paikkaan.
- Varsinaisen hyperlinkin saat komennolla `\hyperlink`, jonka ensimmäinen argumentti on kohde eli viittausavain, ja toinen on linkiksi tehtävä pala tekstiä.
- Jo esitettyjä kalvoja voi myös helposti toistaa komennolla `\againframe`, josta lisää Beamerin manuaalissa.

- Nappuloita hyperlinkeille voit tehdä komennoilla `\beamerbutton`, `\beamergotobutton`, `\beamerskipbutton` ja `\beamerreturnbutton`, jotka kukin ottavat ainoaksi pakolliseksi argumentikseen nappulaan kirjoitettavan tekstin.
- Antamalla tällaisen nappulan `\hyperlink`-komennon toiseksi argumentiksi, saat nappulan joka vie jonnekin muualle.

Esimerkkejä linkeistä ja nappuloista

◂ Takaisin hyperlinkkeihin

Muista myös tulostusversiot.

Esimerkin tuottanut koodi

```
\hyperlink{fr:hyperlinkit}{\beamerreturnbutton{Takaisin
hyperlinkkeihin}}\\
Muista myös \hyperlink{fr:tulostusversio}{tulostusversiot}.
```

- Beamerin kautta saat esityksiisi myös esimerkiksi ääniefektejä.
- Lisäksi saat massiivisia siirtymäefektejä, joilla voit häikäistä yleisösi.
- Näitä käyttäessä kannattaa kuitenkin hetki pohtia, kumpaa seuraavista efektisi ovat:
  1. Oikeasti esityksen ymmärtämistä helpottavia apukeinoja.
  2. Tyhjää ja tarpeetonta tietoteknistä kikkailua.

Osa VI

Viisauden hippusia

Usein kysyttyjä kysymyksiä & yleisimpiä virheitä

Esipuhe

Jotkin kysymykset ja ongelmat tulevat vastaan useammat kuin toiset. Seuraavan kappaleen tarkoituksena on ennaltaehkäisevästi vastata näistä yleisimpiin.

Esimerkkisijoitus virheen yleiseen etenemislakiin

Fysiikan selkkareita tehdessä tulee jokaiselle jossain vaiheessa vastaan tilanne, jossa on tehtävä esimerkkisijoitus virheen yleiseen etenemislakiin. Tämän seurauksena syntyy niin valtava neliöjuurilauseke, että se ei mahdu yhdelle eikä välttämättä edes kahdelle riville.

Toimi näin:

- Lisää lausekkeen ympärille sulut ja sulkujen jälkeen potenssi $\frac{1}{2}$.
- Korota yhtälön toinen puoli toiseen potenssiin.
- Erityisesti virheen yleisen etenemislain tapauksessa: Laske virhetermit $(\partial f/\partial x)\cdot\Delta x$ erikseen ja yhdistä lopussa. Samalla pääset tarkastelemaan eri virhelähteiden keskenäistä merkittävyyttä, mistä assarit perinteisesti tykkäävät.

Matematiikkatilan komentoja väärinkäyttämällä *voi* tehdä eräänlaisen monirivisen neliöjuuren, mutta tällainen merkintä on varmasti typerän ja epäselvän näköinen.

Suomessa on tapana merkitä integraalilaskuissa sijoitusta suurella kauttaviivalla, eli tyyliin

$$\int_a^b f(x)\,\mathrm{d}x = \Big/_a^b F(x).$$

Suuressa maailmassa tämä merkintätapa ei kuitenkaan ole käytössä, ja ehkä siksi myöskään LaTeXin matematiikkakomennoista ei suoraan löydy edellisenkaltaista merkintää latovaa komentoa.

Komennon `\sijoitus` määrittely (kiitokset Martti Nikuselle)

Siihen asti kunnes joku suomalainen hoitaa asian ja tekee LaTeXiin `sijoitus`-paketin, voi seuraavalla loitsulla määritellä uuden komennon `\sijoitus`, joka ottaa argumentikseen sijoituksen rajat.

```
\newcommand{\sijoitus}[2]{\mathop{\Big/}\limits_{\hspace{-0.85em}{#1}}^{\hspace{0.85em}{#2}}\hspace{-0.2em}}
```

Komennosta löytyviä `\hspace`ja saattaa joutua hienosäätämään jos käyttää eri fonttia. Paremmalle ratkaisulle olisi siis tilausta…

Suljekomennot `\left` ja `\right` eivät toimi rivinvaihtojen kanssa

- Tavalliset sulkeita automaattisesti kasvattavat komennot `\left` ja `\right` on toteutettu siten, etteivät ne toimi mikäli välissä on rivinvaihto.
- Vastaus: lataa paketti `breqn`.
- Tämä paketti sisältää myös muita pieniä korjauksia rivinvaihtoihin näyttömatematiikkatilassa. Pakettiin kannattaa siis tutustua, mikäli moniriviset lausekkeet aiheuttavat harmia.

- Sivunumeroinnin tyyliä voi vaihtaa kesken dokumentin komennolla `\pagenumbering`. Komento ottaa yhden argumentin:

`arabic`	Arabialaiset numerot: 1, 2, 3, ...
`roman`	Pienet roomalaiset numerot: i, ii, iii,...
`Roman`	Suuret roomalaiset numerot: I, II, III,...
`alph`	Aakkoset: a, b, c,...
`Alph`	Suuraakkoset: A, B, C,...

- Komento `\pagenumbering` palauttaa sivunumeroinnin samalla takaisin alkuun.
- Sivunumeroa puolestaan voi vaihtaa komennolla `\setcounter`. Esimerkiksi komento `\setcounter{page}{1}` palauttaa sivunumeron takaisin ykköseen.
- Jos peukaloit sivunumeroita ja käytät kaksipuoleista tulostusta, varo ettet sekoita sivujen järjestystä – aukeaman oikeanpuoleisen sivun tulee olla sivunumeroltaan pariton.
- Jos käytät `book`-luokkaa, tutustu komentoon `\frontmatter`.

- Joissain yhteyksissä on vakiintunut tapa numeroida taulukot roomalaisilla ja kuvat arabialaisilla numeroilla. Esimerkiksi jos taulukoihin ja kuviin viittaa paljon samoissa lauseissa, tällainen numerointi voi olla viisasta.
- Kutakin LaTeXin laskuria (sivunumero, taulukon numero ym.) kohti on komento, jota LaTeX käyttää latoessaan esim. sivunumeron tiettyyn kohtaan. Uudelleenmäärittelemällä tämä komento voidaan kätevästi muuttaa numeroinnin tyyliä.

Taulukoiden numerointi roomalaisilla numeroilla

```
\renewcommand{\thetable}{\Roman{table}}
```

- Ainakin suuremmissa dokumenteissa voi olla viisasta numeroida esim. matemaattiset lausekkeet ja kuvat juoksevan numeroinnin sijasta lukukohtaisesti. Suurille dokumenteille tarkoitetut dokumenttityylit tekevät tämän jo oletuksena.
- Jos esimerkiksi `article`-tyyliseen dokumenttiin haluaa lukukohtaisen numeroinnin, voi sen tehdä kikkailemalla laskurien kanssa, mutta siistimpi ratkaisu on käyttää pakettia `chngcntr`.

`section`-kohtainen yhtälönumerointi `chngcntr`-paketilla

```
\counterwithin{equation}{section}
```

UKK-osion viimeiseksi on varattu tiukasti ykkössijasta kilpaileva ongelma, eli *”nää kuvat ei mee minne mää haluun”*.

- LaTeX pyrkii leijuvaisten sijoittelussa ottamaan huomioon tekstin luettavuuden. Käytännössä tämä on toteutettu siten, että sijoittelualgoritmi ei esimerkiksi laita sivulle kuvaa, mikäli sivulle jäisi liian vähän tekstiä suhteessa kuvien määrään.
- Joskus algoritmi ja kirjoittaja ovat eri mieltä, jolloin toisen on väistyttävä. Tällöin kannattaa tietää, miten sijoittelualgoritmin parametreja voi muuttaa.

Seuraavat parametrit määrittelevät sivukohtaisesti, kuinka monta leijuvaista sivulla voi tietyissä paikoissa korkeintaan olla.

- `topnumber` sivun ylälaidassa (sijoittelukirjain `t`)
- `bottomnumber` sivun alalaidassa (sijoittelukirjain `b`)
- `totalnumber` yhteensä

Nämä argumentit ovat LaTeXille laskureita, eli niiden uudelleenmäärittely tehdään komennolla `\setcounter`. Oletuksena nämä ovat tavallisissa dokumenttityyleissä 2 ylhäällä, 1 alhaalla ja 3 yhteensä.

Esimerkki: leijuvaisten kokonaismäärän kasvattaminen viiteen per sivu

```
\setcounter{totalnumber}{5}
```

Seuraavat parametrit asettavat leijuvaisten viemälle pystysuuntaiselle tilalle rajoitukset. Kukin parametri on osuus tilan kokonaismäärästä.

- `\topfraction` määrää maksimiosuuden sivun ylälaidassa (`t`-kirjain) oleville leijuvaisille.
- `\bottomfraction` tekee saman alalaidassa oleville leijuvaisille.
- `\textfraction` määrää tekstin minimiosuuden.
- `\floatpagefraction` asettaa vähimmäisrajan kuvien osuudelle leijuvaissivusta (`p`-kirjain). LaTeX ei tee leijuvaissivua, jos se ei löydä tarpeeksi kuvia täyttämään tätä osuutta sivusta.

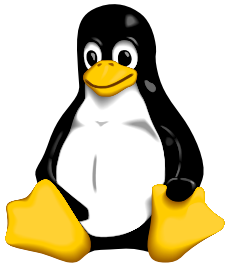
Nämä argumentit ovat komentoja, joten niiden uudelleenmäärittely tehdään `\renewcommand`-komennolla. Oletuksena nämä ovat tyypillisesti 70%, 30%, 20% ja 50%.

Esimerkki: ylitäysien sivujen salliminen

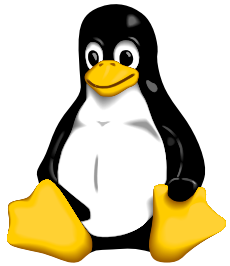
```
\renewcommand{\topfraction}{0.95}
```

- Kun siis seuraavan kerran mielessäsi herää kysymys miksi LaTeX ei jotain kuvaa johonkin kohtaan laita, pohdi edellisten oletusparametrien kannalta näkeekö LaTeX kyseisessä paikassa tilaa tälle kuvalle.
- Mikäli työssäsi on todella kolossaalisen paljon kuvia tai taulukoita:
  - Suosi leijuvaissivuja. Näin leijuvaiset eivät ole koko ajan rikkomassa lukurytmiä.
  - Mikäli kuvat liittyvät kiinteästi toisiinsa, suosi kuvakollaaseja (esitellään pian).
  - Jos kuvia ja taulukoita on *todella* paljon, laita ne liitteeiksi. Työsi seuraamista haittaa jos jokaisen tekstisivun jälkeen tulee kymmenen sivua kuvia.
- Jos jostain syystä on *aivan kertakaikkisen absoluuttisen pakko* saada kuva täsmälleen johonkin paikkaan, etkä jaksa säätää sijoittelualgoritmin parametreja, lataa paketti `float` ja käytä sijoittelukirjainta `H`.

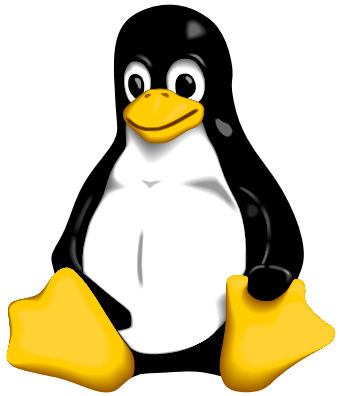
- Kuvakollaasilla tarkoitetaan useamman, toisiinsa liittyvän kuvan kokoelmaa. Kuvilla voi yhteisen kuvatekstin ja -numeron lisäksi olla omat kuvatekstinsä sekä numeronsa, siten että sekä koko kollaasiin että yksittäisiin kuviin voidaan viitata tekstissä.
- LaTeXissa kuvakollaasien toteuttamiseen on ainakin kolme pakettia, `subfigure`, `subfig` ja `subcaption`. Näistä `subcaption` on uusin ja monipuolisin.
- Kutakin pakettia käytettäessa normaalin `figure`-ympäristön sisälle lisätään alikuvia omilla komennoillaan. Sekä alikuville että koko kollaasille saadaan viittausavaimet tutulla `\label`-komennolla.
- Myös taulukkokollaaseja voi tietenkin tehdä samalla tavalla.



(a) Eka kuva.



(b) Toka kuva.



(c) Kolmas kuva.

Kuva 2: Esimerkki kuvakollaasista. Huomaa erityisesti kuva 2b.

Esimerkki kuvakollaasista (toteutettu `subcaption`-paketilla)

```
\begin{figure}
\centering
\begin{subfigure}{0.20\textwidth}        ⟵ Pakollinen argumentti antaa alikuvan leveyden
  \centering
  \includegraphics[width=4em]{Tux}
  \caption{Eka kuva.}
  \label{fig:kollaasi1}        ⟵ Kullakin alikuvalla on oma caption ja label
\end{subfigure}\qquad
\begin{subfigure}{0.20\textwidth}
  ...
\end{subfigure}\\
\begin{subfigure}{0.25\textwidth}
  ...
\end{subfigure}
\caption{Esimerkki kuvakollaasista. Huomaa erityisesti
kuva~\ref{fig:kollaasi2}.}
\label{fig:kollaasi}        ⟵ Myös koko kollaasilla on kuvateksti ja viittausavain
\end{figure}
```

- Yleisimpiä LaTeXin varoituksia ovat pahat laatikot eli "`Overfull \hbox`" ja "`Underfull \hbox`".
- Ylitäysi laatikko syntyy, kun materiaali jatkuu marginaalin yli. Tähän syynä voi olla esimerkiksi liian suuri kuva, mutta useimmin on kyse siitä, että LaTeX ei löydä tapaa tavuttaa tekstiä siten, että marginaali ei ylittyisi.
- Alitäysi laatikko puolestaan yleensä viittaa tarpeettomaan rivinvaihtoon. Tämän varoituksen voi yleensä melko huoletta ohittaa.

- Kun mikään kombinaatio tavutuskohtia ei tuota tasaisia marginaaleja ilman että sanavälit venyvät liikaa, ei LaTeX muuta voi kuin antaa varoituksen ja kysyä neuvoa käyttäjältä. Käyttäjä voi valita esimerkiksi seuraavista tavoista korjata paha laatikko:
- **Korjaa ongelmakohdat**. Ensimmäisenä kannattaa tarkastaa, ettei kyse ole esimerkiksi jostain sanasta, jota LaTeX ei yhdysviivan vuoksi tai muusta syystä tavuta.
- **Tekstin muuttaminen**. Se yksinkertaisin ratkaisu. Muuttamalla tekstiä hieman voi ongelmallinen kohta hyvinkin poistua.
- **`sloppypar`**. Kirjoittamalla ongelmallinen tekstikappale ympäristön `sloppypar` sisään, LaTeX sallii sanavälien venymisen rajatta. Tämä voi näyttää vähintäänkin yhtä rumalta kuin marginaalin yli jatkuva tekstirivi!

Suureiden ja yksiköiden merkitsemisessä näkee usein virheitä. Fysikaalisten suureiden merkintää säätelevät SI-järjestelmän säännöt sekä standardi ISO 80000. Tässä muutamia otteita:

- **Yksiköt kirjoitetaan aina pystykirjaimin:**
  ei siis 10 μm vaan 10 µm.
  Pysty-myyn saat esimerkiksi `textcomp`-paketin komennolla `\textmu`.
- **Älä käytä tuhaterottimena pilkkua tai pistettä**, vaan ohuketta eli lyhyttä väliä (komento `\,`).
- **Suureet merkitään kursiivilla**, eli LaTeXissa matematiikkatilassa:
  ei $\mathrm{E} = \mathrm{Mc}^2$ vaan $E = Mc^2$.
- Mikäli suureen merkintä sisältää esimerkiksi jonkin selventävän alaindeksin, tulee **alaindeksi kursiivilla vain jos se on jokin suure**.
  Esimerkiksi maksimipaine voisi olla p_{max} mutta ei p_{max}.

- **Lukuarvon ja yksikön väliin tulee väli**. Ei siis esimerkiksi 10m vaan 10 m. Ainoat poikkeukset tähän ovat tasokulmien mittaamiseen käytetyt asteet, minuutit ja sekunnit, jotka merkitään ilman väliä.
- Matemaattiset vakiot, kuten Neperin luku e, imaginaariyksikkö i sekä ympyrän kehän ja halkaisijan suhde π, tulee kirjoittaa pystykirjaimin (vrt. $e\ i\ \pi$).
  - Pysty-π:tä ei kaikista matematiikkafonteista valitettavasti löydy. Ainakin muutenkin laajalla `newpxmath`-fonttipaketilla sellaisen saa komennolla `\uppi`.

Lisää voi lukea esimerkiksi osoitteesta
`http://en.wikipedia.org/wiki/ISO_31`.
ISO 31 oli edellinen matemaattista merkintää käsittelevä standardi.

- Suureiden oikeaoppista merkintää helpottaa paketti `SIunits`. Pakettia käyttämällä esimerkiksi 10 µm saadaan komennolla `\unit{10}{\micro\meter}`.
- Paketti pitää huolen siitä, että merkintä on standardien ja asetusten mukainen.
- Paketin avulla suureiden merkintätyyliä voidaan vaihtaa kerralla vaihtamalla paketille annettavia argumentteja, esimerkiksi jos halutaan vaihtaa suureen ja yksikön väliin lyhyempi sanaväli.
- Kiinnostuneet selvittäköötpaketin käytön itse tarkemmin.
- Yksiköiden latomiseen on myös uudempi paketti, `siunitx`, johon myös ehkä kannattaa tutustua.

Ulkoasun viilaaminen

Esipuhe

Seuraavassa kappaleessa jaetaan hieman vinkkejä, jotka auttavat dokumentin ulkoasun hiomisessa viimeisteltyyn ja ammattimaiseen muotoon. Ennen hienosäätöä kannattaa kuitenkin tarkistaa, että itse sisältö on kunnossa.

Kun ryhdytään puhumaan ulkoasun hienosäätämisestä, kannattaa olla tarkkana ettei tekstin varsinainen tavoite hämärry.

Kun Leslie Lamportilta kysyttiin kolmea virhettä, joita LaTeX-käyttäjien tulisi varoa tekemästä, hänen vastauksensa oli:

1. *Worrying too much about formatting and not enough about content.*
2. *Worrying too much about formatting and not enough about content.*
3. *Worrying too much about formatting and not enough about content.*

Kun teksti toimii hyvin eikä dokumentin sisällössä ole enää korjattavaa, kirjoittaja voi kuitenkin hemmotella itseään ulkoasun pikkutarkalla viilaamisella. Tähän LaTeX tarjoaa hyvät keinot.

- PDF-standardi sisältää luettelon fonteista, jotka tulee löytyä jokaisesta PDF-lukijasta. Näin PDF-dokumentit, jotka käyttävät näitä vakiofontteja (mm. *Times* ja *Palatino*) saadaan mahtumaan pienempään tilaan, koska fonttia ei tarvitse liittää dokumentin mukaan.
- Tämä oli hyvä idea, mutta reaalimaailmassa PDF-lukijaohjelmat eivät lisenssimaksujen ja muiden härdellien vuoksi sisällä täsmälleen samoja fontteja.
  - "*Times*" voi Windowsissa olla oikeasti *Times New Roman* ja Linuxissa *URW Nimbus Roman*. Nämä fontit ovat hyvin samankaltaiset, mutta eivät täysin identtiset.
- Dokumenttinsa ulkoasusta huolehtivan kirjoittajan kannattaakin tästä syystä varmistaa, että *kaikki* dokumentissa käytetyt fontit liitetään dokumentin mukaan.

- Dokumenttinsa sisältämät fontit saa tarkastettua esimerkiksi Adobe Readerillä valitsemalla File→Document Properties→Fonts. Jos fontti on *Embedded*, on se liitetty dokumentin mukaan. Lisäksi jos fontin kohdalla lukee *Subset*, on fontista liitetty mukaan (viisaasti) vain käytetyt merkit.
- Jotkin TEX-distribuutiot (ainakin TEX Live) liittävät kaikki fontit PDF-tiedostoihin oletuksena. Muissa distroissa voi esimerkiksi etsiä käsiinsä asetustiedoston `updmap.cfg` ja tarkistamalla että sieltä löytyy seuraava rivi:

Kaikkien fonttien liittämisen pakottava rivi tiedostossa `updmap.cfg`.

```
pdftexDownloadBase14 true
```

Voi myös olla, että distribuutiostasi löytyy jokin helppokäyttöinen asetusvalikko, josta tämän asetuksen saa päälle.

Yleistyökalu kuvatekstien ja taulukoiden kuvauksien säätämiseen on paketti `caption`. Paketti on ominaisuuksiltaan varsin laaja, joten sitä ei tässä yksityiskohtaisesti käydä läpi.

Esimerkki `caption`-paketin asetuksista

Esimerkiksi seuraavat argumentit muokkaavat kuvatekstejä siten, että itse kuvateksti tulee pienellä fontilla, ja kuvatekstin alussa oleva ”Kuva 1” lihavoidaan ja erotetaan kuvatekstistä pisteellä. Lisäksi kuvatekstit tasataan vasempaan reunaan ja taulukoiden kuvaustekstejä oletetaan käytettävän taulukoiden yläpuolella.

```
\usepackage[labelsep=period,justification=RaggedRight,
  tableposition=top,font=small,labelfont=bf]{caption}
```

Yksi monista tavoista ilmaista ”A määritellään olemaan B” on lisätä kaksoispisteet yhtäsuuruusmerkin eteen, eli $A := B$. Tarkka lukija huomaa kuitenkin tässä vaiheessa, että kaksoispiste ei luonnostaan ole tasattu yhtäsuuruusmerkin kanssa samalle tasolle.

Myös tämä ongelma on helposti ratkaistavissa:

- Paketista `mathtools` löytyy komento `\coloneqq`, joka latoo oikeanlaisen merkinnän. Samasta paketista löytyy myös lisää vastaavanlaisia kaksoispisteitä ja ekvivalenssisymboleja lisääviä merkintöjä.
- Jotkin fonttipaketit kuten `pxfonts` ja `txfonts` sisältävät itsessään komennon `\coloneqq` ja kumppanit.

Komentoa `\coloneqq` käyttämällä merkintä näkyy oikein: $A := B$.

Uusi murtomerkintä: `\nicefrac`

- Joidenkin yksinkertaisten murtomerkintöjen kanssa olisi mukavampaa merkitä mieluummin $1/2$ kuin 1/2 tai $\frac{1}{2}$.
- Tällaisen murtomerkinnän saa komennolla `\nicefrac`, joka löytyy paketista `nicefrac`.
- Saman tekee hieman monipuolisemmin ja joillain fonteilla paremmin paketti `xfrac`.

- LaTeXin matematiikanladontatyyliä teksti- ja näyttömatematiikkatilan välillä voidaan vaihtaa myös lennossa komennoilla `\displaystyle` ja `\textstyle`. Näin voidaan esimerkiksi korostaa lausekkeen sisällä turhan vaatimattomasti ladottavaa osaa latomalla se näyttömatematiikkatilassa.
- Komentojen vaikutusaluetta voidaan rajata aaltosuluilla: `{\displaystyle ...}`.

Esimerkki \displaystyle-komennon käytöstä

$$\frac{\displaystyle\int_0^{x+\log x} \mathrm{e}^{-t^2}\,\mathrm{d}t}{\mathrm{e}^{x^2}} \quad \text{vastaan} \quad \frac{\int_0^{x+\log x} \mathrm{e}^{-t^2}\,\mathrm{d}t}{\mathrm{e}^{x^2}} \quad \left(\text{vastaan} \quad \mathrm{e}^{-x^2}\int_0^{x+\log x} \mathrm{e}^{-t^2}\,\mathrm{d}t\right)$$

- Joissain tapauksissa AMS:n matematiikkapaketit tuottavat epäoptimaalista jälkeä. Tällaisia tapauksia ovat esimerkiksi suurten ylä- ja alaindeksien latominen sekä peräkkäiset, alaindeksejä sisältävät raja-arvomerkinnät.
- Koska AMS ei mielellään muuta pakettiensa toimintaa ainakaan kovin lyhyellä aikavälillä ($\sim 10^2$ vuotta), tällaisiin tilanteisiin on AMS-LaTeXin "korjaussarja" `mathtools`.
- Paketti `mathtools` lataa paketin `amsmath` ja korjaa siitä muutamia pieniä ongelmia, joten oikeastaan nykyään kannattaa ladata aina `mathtools`.

Kaksi esimerkkiä lausekkeista, joissa on tarvetta `mathtools`-paketille.

$$X = \sum_{1\le i<j\le n} Y_{ij} \quad \longrightarrow \quad X = \sum_{1\le i<j\le n} Y_{ij} \qquad (\texttt{\textbackslash mathclap})$$

$$y = \inf_{n} \sup_{m\ge n} x_m \quad \longrightarrow \quad y = \inf_{n} \sup_{m\ge n} x_m \qquad (\texttt{\textbackslash adjustlimits})$$

- Maailmassa on liian vähän niin kauniita asioita kuin `pdftex`in mikrotypografia, eli yksittäisten kirjaimien millimetrin kymmenesosien tasolla tapahtuva hiominen. Tätä tarkempaa ei ulkoasun viilaminen voi olla! ☺
- Eräs näistä on *marginaalien optinen suoristaminen*, eli tiettyjen ”kapean näköisten” merkkien työntäminen aivan aavistuksen verran marginaalista ulos siten, että lopputulos näyttää ihmissilmälle täydellisen suoralta.
- Toinen on *kirjainten venytys*, jossa tekstin sanavälit ja rivitys pidetään harmonisen kauniina tarpeen vaatiessa venyttämällä tai supistamalla yksittäisiä kirjaimia esimerkiksi sadasosan verran.
- Molemmat saat käyttöön lataamalla paketin `microtype`. Mikrotypografia vaatii `pdflatex`-kääntäjän käyttämisen.

Sekalaisia vinkkejä

Esipuhe

Kurssin lopuksi muutamia sekalaisia vinkkejä, jotka ovat vuosien saatossa osoittautuneet hyödyllisiksi.

- Tekstin muokkausvaiheessa on hyvä kertoa LaTeXille että tässä vaiheessa tuotetaan vasta raakavedoksia. Tällöin LaTeX nopeuttaa koodin kääntämistä esimerkiksi jättämällä kuvat lopputuloksesta pois sekä ohittamalla osia tekstin hienosäädöstä. Lisäksi tällöin LaTeX merkitsee mustilla suorakaiteilla tekstiin kohdat, jossa rivi jatkuu yli marginaalin.
- Raakavedoksen saa käyttöön antamalla dokumenttityylille valinnaisen argumentin `draft`.
- Eräs raakavedoksissa ohitettavia hienosäätöjä on paketin `microtype` mikrotypografia. Tämä voi olla hankalaa esimerkiksi rivityksen hienosäätämisen kannalta, sillä mikrotypografia muuttaa rivitystä varsin oleellisesti, jolloin `draft`-asetus päällä viimeistelty dokumentti rivittyykin eri tavalla kun `draft` poistetaan. Mikrotypografian saa mukaan myös `draft`-vaiheessa antamalla paketille `microtype` argumentin `final`.

- Todella suuria dokumentteja käsiteltäessä voi kooditiedosto kasvaa epäkäytännöllisen pitkäksi. Tällöin on hyötyä komennoista `\input` ja `\include`.
- Komento `\input` ottaa argumentikseen LaTeX-koodia sisältävän tiedoston tiedostonimen (päätteen `.tex` voi jättää pois). Komento liittää käännösvaiheessa tiedoston sisällön `\input`-komennon paikalle.
- Toinen koodin pilkkomiseen soveltuva komento on `\include`, joka toimii kuten edellinen, mutta `\include`-komennolla mukaan liitetyistä kooditiedostoista voi valita käännettäväksi vain osan käyttämällä alustuksessa komentoa `\includeonly`, joka ottaa argumentikseen pilkuilla erotetun listan `\include`-komentojen argumentteja.
- Lisäksi `\include` toimii siten, että se pystyy säilyttämään oikeat sivunumerot vaikka dokumentista käännettäisiinkin vain osa. Tätä varten `\include` kuitenkin joutuu vaihtamaan sivua ennen ja jälkeen koodin liittämisen.

- Usein voi välttää suuren määrän toistoa koodissa määrittelemällä itse ovelasti uusia komentoja esimerkiksi usein käytettyjen matemaattisten merkintöjen osalta.
- Tämä onnistuu komennolla `\newcommand`. Komento ottaa neljä argumenttia, joista toinen ja kolmas ovat valinnaisia.
  1. Ensimmäinen argumentti kertoo komennon nimen. Nimen tulee alkaa kenoviivalla.
  2. Toinen, valinnainen argumentti on numero väliltä 1–9, ja se kertoo komennon argumenttien lukumäärän. Mikäli argumentti jätetään pois, määriteltävä komento ei ota argumentteja.
  3. Kolmas argumentti on myös valinnainen, ja se antaa määriteltävän komennon ensimmäisen argumentin oletusarvon. Tällöin määriteltävän komennon ensimmäisestä argumentista tulee valinnainen.
  4. Neljäs argumentti on koodipätkä, joka kertoo mitä komento tekee. Koodipätkässä `#1` korvautuu ensimmäisellä komennolle annetulla argumentilla, `#2` toisella ja niin edelleen.

- Jos halutaan korvata jo olemassaoleva komento jollain toisella, tulee `\newcommand`-komennon tilalla käyttää komentoa `\renewcommand`.
- Myös uusia ympäristöjä voi määritellä. Tähän on komento `\newenvironment`, joka toimii kuten `\newcommand`, mutta se ottaa yhteensä viisi argumenttia, joista kaksi viimeistä määrittelevät `\newcommand`-komennon neljännen argumentin tavoin mitä tapahtuu ympäristön alussa ja lopussa.

Esimerkki: pikakomento derivaatalle

```
\newcommand{\derva}[2]{\frac{\mathrm{d}#1}{\mathrm{d}#2}}
```

`\derva{f}{x}` $\longrightarrow$ $\frac{\mathrm{d}f}{\mathrm{d}x}$

Myös `\derva fx` tekee saman!

- Suuria määriä kuvia ja taulukoita sisältävissä dokumenteissa on joskus tapana liittää sisällysluettelon yhteyteen erillinen kuva- ja/tai taulukkolista, jossa näkyy kunkin kuvan kuvateksti ja kunkin taulukon kuvaus, sekä sivunumero josta kuva tai taulukko löytyy.
- Sisällysluettelon tavoin näidenkin tekeminen on helppoa: Komento `\listoffigures` luo kuvalistan ja `\listoftables` taulukkolistan.
- Molempien oletusulkoasu määrätyy dokumenttityylistä, ja hienosäätöä voi tehdä paketin `tocloft` avulla samoin kuin sisällysluettelon tapauksessa.
- Jos kuvatekstit tai taulukoiden kuvaukset ovat tarpeettoman pitkiä listauksia varten, voi molemmille antaa myös vaihtoehtoisen, listauksissa käytettävän kuvaustekstin komennon `\caption` valinnaisena argumenttina.

- PDF-formaatin etuja on se, että se tukee *metadataa*, eli dokumentin yleisten tietojen kuten tekijän ja otsikon sisällyttämistä tiedostoon koneellisesti käsiteltävässä muodossa.
- Metadataa voi lisätä dokumenttiinsa `\pdfinfo`-komennolla, joka ottaa argumentikseen tekstipätkän PDF:n omalla notaatiolla.
- Myös `hyperref`-paketti tarjoaa keinot metadatan lisäämiseen. Lataamalla `hyperref`-paketti argumentilla `pdfusetitle` metadatan tiedot täytetään komentojen `\author` ja `\title` tiedoista.

Esimerkki `\pdfinfo`-komennosta

```
\pdfinfo{
  /Author (Perttu Luukko)
  /Title (Tieteellisen tekstin tuottaminen LaTeXilla)
  /Subject (LaTeX)
  /Keywords (LaTeX;typografia;vompatit)
}
```

- Aiemmin esiteltiin dokumentin rivivälin muuttaminen puolitoista- tai kaksinkertaiseksi, mikä voi olla hyödyllistä jos rivien väliin halutaan tehdä käsin merkintöjä.
- Riviväliä voi myös hienosäätää. Joissain tapauksissa esimerkiksi tekstin luvun viimeinen rivi menee ikävästi seuraavan sivun puolelle. Pienentämällä riviväliä esimerkiksi 1 % luvun loppu sujahtaa kauniisti sivun loppuun. Kuitenkin harva lukija on niin tarkkasilmäinen, että huomaisi hyvin pienen muutoksen rivivälissä.
- Riviväliä voi LaTeXissa hienosäätää komennolla `\linespread`, joka ottaa argumentikseen kertoimen desimaalilukuna. Esimerkiksi `\linespread{1.02}` leventää riviväliä hiukan. Jos komentoa käytetään alustuksen ulkopuolella, sen jälkeen tarvitaan vielä komento `\selectfont`, jotta vaihdos astuu voimaan.
- Huomaa, että `\linespread{1.5}` ei tarkoita puolitoistakertaista riviväliä, siten kuten se usein määritellään.

Aiemmin esiteltyjen rivi- ja sivunvaihtokomentojen lisäksi LaTeXin komentovalikoima sisältää myös seuraavat:

- Komento `\\` ottaa myös valinnaisen argumentin, joka kertoo kuinka paljon ylimääräistä pystysuuntaista tilaa ladotaan ennen seuraavaa riviä, esim. `\\[1em]`.
- Edellisen tähdellinen versio `\\*` estää sivunvaihdon rivinvaihdon kohdalla.
- `\linebreak` vaihtaa riviä, mutta venyttää kesken jäänyttä riviä niin, että se ylettyy marginaaliin. Komento ottaa myös valinnaisen argumentin, joka on kokonaisluku väliltä 0–4. Tämä muuttaa rivinvaihtokäskyn pyynnöksi, ja pyynnön voimakkuus on komennolle annettu luku.
- `\nolinebreak` toimii kuten edellinen, mutta estää (tai pyytää välttämään) rivinvaihtoa.
- `\pagebreak` ja `\nopagebreak` tekevät saman kuin kaksi edellistä, mutta sivunvaihdoille

- Selkkaria tai gradua kirjoittaessa pitäisi usein muistaa esimerkiksi tuleeko ilmaus ”näin ollen” yhteen vai erikseen, tai tuleeko ilmaisuun ”pro gradu -tutkielma” yhdysviiva vai ajatusviiva.
- Kielioppi- ja merkintöasioissa hyvä lähdeteos on Jukka K. Korpelan *Nykyajan kielenopas*, joka löytyy osoitteesta `http://www.cs.tut.fi/~jkorpela/kielenopas/index.html`.
- Tämä opas voittaa laajuudessa monet kirjaksi painetut oikeakielisyysoppaat. Lisäksi opas on kirjoitettu hyvin kansantajuisesti (ja fyysikontajuisesti).

Loppusanat

Esipuhe

Kaikki loppuu aikanaan. Ennen loppua haluan kuitenkin vielä tarjota muutaman yleisen ohjeen viitoittamaan LaTeX-käyttäjän uraanne.

Harjoitus tekee mestarin

Tämä kurssi on ollut pikakurssi, ja sellaisena se on sisältänyt aivan liian vähän käytännön harjoitusta. Ei kannata kuitenkaan lannistua vaikka asiat tuntuisivat aluksi työläiltä, sillä kun olet muutaman isomman työn LaTeXilla vääntänyt, kaikki ne vaikealta tuntuneet komennot syöksyvät sormenpäistäsi ajatuksen nopeudella.

KVG

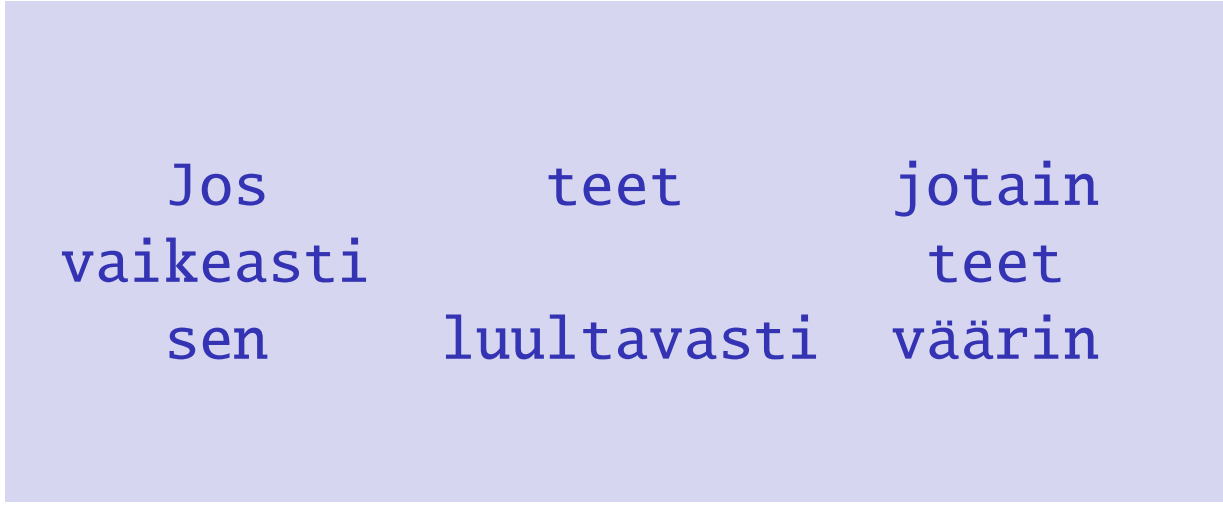
KATSO VAIKKA GOOGLESTA

Jos kohtaat ongelmia, älä tuskaile niiden kanssa yksin kammiossasi. Moniin ongelmiin paikallinen TEXperttisi, IRCNet-kanava `#LaTeX.fi`, CTAN ja valitsemasi hakukone osaavat vastata välittömästi.

```
\def\g@tlm@ff{% calculating \lm@ff % \lm@ff:=lmoff(left tree)-ltop(left tree)
% -.5\tots@p+\lt@p \g\lm@ff\l@ftlmoff \g\advance\lm@ff by-\l@ftltop \g\advanc
e\lm@ff by-\halftots@p \g\advance\lm@ff by\lt@p\relax % if ht(left tree) < ht
(right tree)% \t@mpdima:=lmoff(--Varo Suuria Muinaisia--right tree)-ltop(righ
\lm@ff:=min(\lm@ff,\t@mpdima) fi \ifnum\l@ftht<\r@ghtht\relax \g\t@mpdima\r@g
htlmoff \g\advance\t@mpdima by-\r@ghtltop \g\advance\t@mpdima by\halftots@p \
g\advance\t@mpdima by\lt@p\relax\ifdim\t@mpdima<\lm@ff\relax \g\lm@ff\t@mpdim
a \fi \fi % \lm@ff:=min(\lm@ff,0pt) \ifdim0pt<\lm@ff\relax \g\lm@ff=0pt% \fi}
```

Kaikkein pirullisimmat LaTeX-ongelmat voidaan usein jäljittää johonkin ”tosi hyvään” dokumenttipohjaan, joka on saatu perintönä vanhemmilta kollegoilta.

Nämä koodit sisältävät usein valtavan määrän tarpeettomia paketteja ja kryptisiä koodipätkiä, joiden merkitys on historian saatossa unohtunut. Kaiken tämän seassa uinuu bugeja, jotka odottavat että tähdet ovat oikeassa asennossa, ja suistavat sitten dokumenttisi tuhoon ja hulluuteen.



Jos huomaat tekeväsi paljon manuaalista työtä jonkin eteen, on haluamaasi asiaan luultavasti olemassa jokin helpompikin tapa.

Etsi se.

Hyvä ja viimeistelty ulkoasu on gradulle tai selkkarille kirsikka kakun päällä ja kromatut vanteet. Mikään määrä typografiaa ei sinua pelasta, mikäli työsi sisältö on epäjohdonmukainen ja sekava.