linux

linux: käyttäjän oikeudet

- Käyttäjällä, `username`, on käyttöoikeus rajattuun levytilaan
- `du -h /home/username/` tulostaa käytetyn levytilan. Yhteenvedon antaa `du -h /home/jetsu/ - -summarize`
- `df` tulostaa sekä vapaan että käytetyn levytilan. Käytännöllinen `alias` on

```
alias Tila='df -h /home/username/'
```

- Käyttäjillä rajatut oikeudet suorittaa komentoja ja käsitellä tiedostoja
- **linux**:ssa käyttäjä kuuluu ainakin yhteen ryhmään. `groups` kertoo mihin
- Käyttäjälle: `u`, Ryhmälle: `g` & Muille: `o` on erilaiset oikeudet seuraaviin: Luku: `r`, Kirjoitus: `w` & Suoritus: `x`
- Listaa tiedostosi komennolla `ls -ls`

linux: käyttäjän oikeudet

- Esimerkiksi `-rwxr-x- -x` tarkoittaa:
  – käyttäjällä on `rwx` oikeudet
  – ryhmällä on `r-x` oikeudet
  – muilla on `- -x` oikeudet
- `d`:llä alkava tulostus (Esim: `drwxr-x- -x`) tarkoittaa, että kyseessä on hakemisto
- Oikeuksia voi muuttaa komennolla

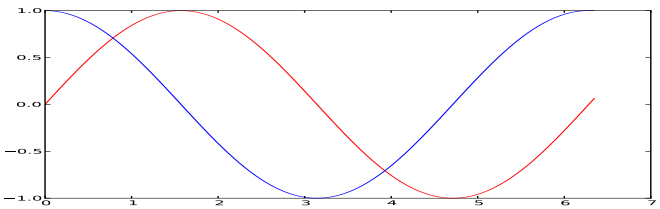
```
chmod [ugoa] (+-) [rwx] nimi
```

- `(u)ser`, `(g)roup`, `(o)thers`, `(a)ll` (`a` on kaikki `u`, `g` ja `o` yhdessä)
- `+` lisää ja `-` poistaa oikeuksia
- `(r)ead`, `(w)rite`, `e(x)ecute`
- Esimerkkki 1: `chmod g+rwx nimi` lisää ryhmällä `rwx`-oikeudet tiedostoon tai hakemistoon `nimi`
- Esimerkkki 2: `chmod go-wx nimi` poistaa ryhmältä ja muilta `wx`-oikeudet tiedostoon tai hakemistoon `nimi`

LaTeX: kuvat

LaTeX: kuvat

- Dokumentin alussa ladataan paketti. Esim:
  `\usepackage[dvips]{graphicx}`
  jossa kuvan lataus
  `\includegraphics[]{tiedosto}`
- `[]` sisään optioita: **`width`**, **`height`**, ...
- Tässä luentojen LaTeX ympäristössä syöte
  `\includegraphics[width=5.0cm,`
  `height=1.5cm]{Pharjoitus17.pdf}`
  tuottaa tämän kuvan



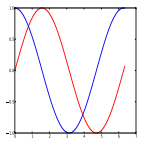
- **`\figure`** (leveys yksi kolumni) ja **`\figure*`** (leveys kaksi kolumnia) ympäristöt: numero, teksti (**`caption`**) ja viittaus tunniste (**`label`**)

LaTeX: kuvat

- Tässä luentojen LaTeX ympäristössä syöte

```
\begin{figure}
\caption[]{Kuvan teksti}
\includegraphics[width=1.0cm,
height=1.0cm]{Pharjoitus17.pdf}
\label{EF}
\end{figure}
Kuvan \ref{EF} sini-käyrät
```

  tuottaa Kuva :**1 Kuvan teksti**



Kuvan 1 sini-käyrät

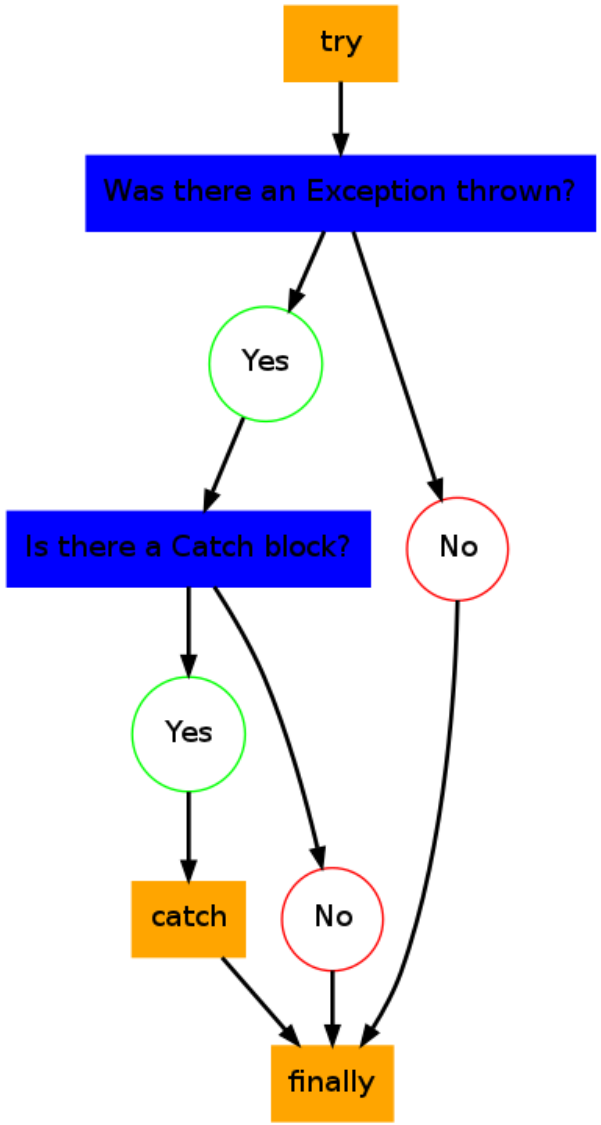
- Kuvat numeroituvat automaattisesti ...
- **`latex`**: kuvat **`ps`**- tai **`eps`**–muodossa
- **`pdflatex`**: kuvat **`jpg`**-, **`png`**- tai **`pdf`**

python ja octave

python: Ohjauskomento `try catch`

(kuva:@sheriframadan.com)

- Ohjauskomennon `try catch` rivi päättyy merkkiin `:`
- Lohkon sisennys
- Käytetään koodin korjaamiseen EnOlKoTa
- Tästä ei laskuharjoitusta
- Esimerkki: **python** ohjelma `Pmalli6.py` seuraavalla sivulla



octave: Ohjauskomento `try catch`

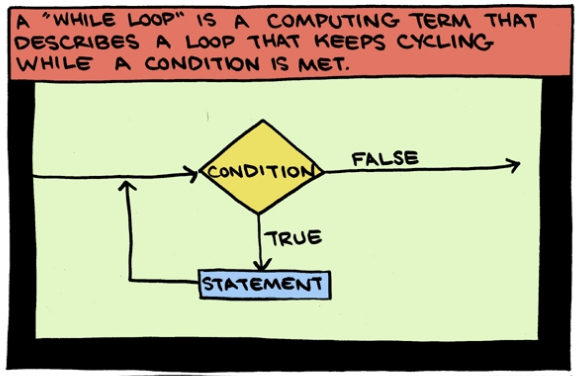
- Ei tarvita sisennystä
- Päättyy komentoon `end_try_catch`
- Tästä ei laskuharjoitusta
- Esimerkki: Sitä seuraavan sivun **octave** ohjelma `Omalli6.m`

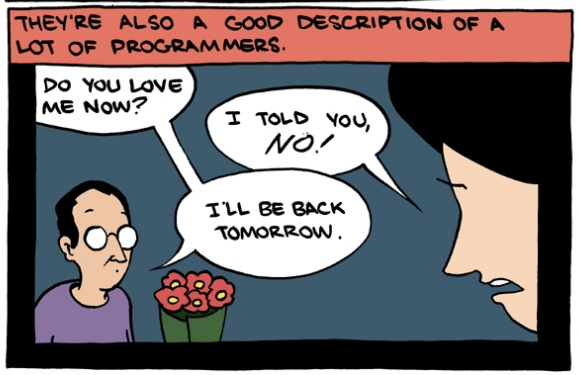
python

python ohjauskomento try catch

(kuva:@www.howtogeek.com)

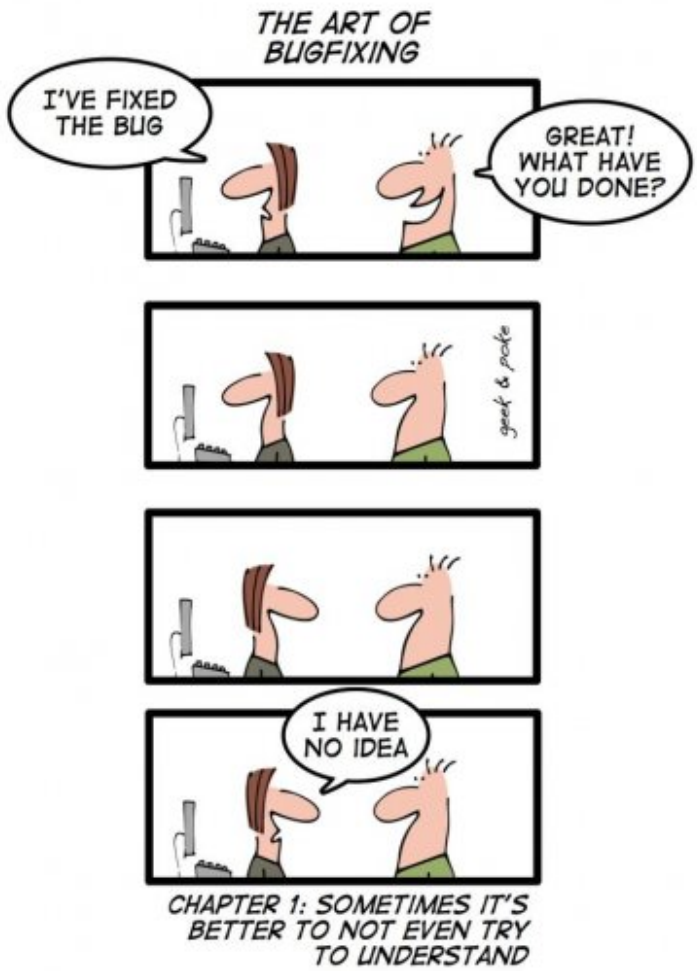
```
# Komenttirivi: Tama on python ohjelmani Pmalli6.py
import os ; os.system('clear')  # Tyhjennetaan naytto
a = [1,2] # a[0]=1,a[1]=2 mutta alkiota a[2] ei ole!
#
#   "print a[2]" komennon virheviesti olisi normaalisti
#   "IndexError: list index out of range"
# Esimerkki 1 ____________________________________________
try:                                # try alkaa
    print(a[2])                     # Komento yritys
except:                             # Jos ei onnistu,
    print("Ei onnistu.")            # printataan tama
# Esimerkki 2 ____________________________________________
try:                                # try alkaa
    print(a[2])                     # Komento yritys
except IndexError:                  # Jos virhetyyppi IndexError,
    print("Virhe tunnistettu")      # niin printataan tama.
except:                             # Jos virhetyyppi ei IndexError,
    print("Virhe tunnistamatta")    # niin printataan tama.
# Esimerkki 3 ____________________________________________
try:                                # Ohjauskomento alkaa
    print(a[2])                     # Sisennys
except TypeError:                   # Jos virhetyyppi TypeError,
    print("Virhe tunnistettu")      # niin printataan tama.
except:                             # Jos virhetyyppi ei TypeError,
    print("Virhe tunnistamatta")    # niin printataan tama.
```





octave: kuva @mattdturner.com

```
# Kommenttirivi: Tama on octave ohjelmani Omalli6.m
clear ; clc                  # Poistetaan ... Tyhjennetaan ...
a = [1 2] ; # a(1)=1,a(2)=2 mutta alkiota a(3) ei ole!
#   "disp(a(3))" virheviesti: "A(I): Index exceeds matrix dim
# Esimerkki 1 ____________________________________
try                          # try alkaa
   disp(a(3))                # Komento yritys
catch                        # Jos ei onnistu,
    disp("Ei onnistu.")      # printataan tama
end_try_catch                # try loppuu
# Esimerkki 2 ____________________________________
try                          # try alkaa
   disp(a(3))                # Komento yritys
catch                        #
msg = lasterror.message;     # Virheviesti talteen
q=strfind(msg,"index out");  # Tarkistetaan msg
  if (q >0)                  # Jos "Inde.." loytyy,
     disp("Virhe tunnistettu")  # niin printataan tama.
  else                       # Jos "Inde.." ei loydy,
     disp("Virhe tunnistamatta")  # niin printataan tama
  endif                      #
end_try_catch                # try loppuu
# Esimerkki 3 ____________________________________
try                          # try alkaa
   disp(a(3))                # Komento yritys
catch                        #
msg = lasterror.message;     # Virheviesti talteen
q=strfind(msg,"Olisikohan tuo");  # Tarkistetaan msg
  if (q>0)                   # Jos "Olis.." loytyy,
     disp("Virhe tunnistettu")  # niin printtaa tama
  else                       # Jos "Olis.." ei loydy,
     disp("Virhe tunnistamatta")  # niin printaan tama
  endif                      #
end                          # try loppuu
```



Aliohjelmat

Funktio/Aliohjelma

- Funktio on aliohjelma.
- Funktio “syö” jotain sisäänsä.
  Tämä `"input"` voi olla muuttujia, tiedostoja, jne..
- Funktio palauttaa jotain syömänsä perusteella.
  Tämä `"output"` voi olla muuttujia, jne..
- Funktio voi myös muokata `"input"` tietoa
- Funktioiden tarkoitus on jakaa ja jäsentää ohjelmaa, sekä vähentää koodin toistoa
- Funktiot on loogisinta määritellä ohjelman alussa, mutta yleensä toimivat muuallakin
- “Muuallakin” = Funktio pitää olla määritetty ennen kuin sitä kutsutaan/käytetään

python

```
def funktio(a,b,..):
```

Sisennys: komennot ...

Sisennys: `return A,B,..`

octave

```
function [A,B,..]=funktio(a,b, ..)
```

Ei sisennystä: komennot

Ei sisennystä: `endfunction`

python & octave

- Funktioiden toiminnoissa on eroja
- Kokeillaan kotisivulta `Psub1.py`
- Kokeillaan kotisivulta `Osub1.m`

python ja octave

python

```
# Ohjelma/Aliohjelma: Psub1.py
def koe(a,b):                    #
    summa=a+b ; erotus=a-b       #
    a=a-9 # Input muutos         #
    return a,b,summa             #
# ____________________________   #
# Kokeilu 1                      #
a=1 ; b=2 ; c=koe(a,b)           #
# ____________________________   #
# Kokeilu 2                      #
#a=1 ; b=2 ; a,b,c=koe(a,b)      #
# ____________________________   #
print('a=',a,'b=',b,'summa=',c)
print(type(c))                   #
# ____________________________   #
# Kokeilu 3                      #
print(erotus) # Ei output!       #
```

- Tehdään kokeilut 1, 2 ja 3
- Mitä tuloksista opitaan?

octave

```
# Ohjelma/Aliohjelma Osub1.m
function [a,b,summa]=koe(a,b)#
summa=a+b ; erotus=a-b ;     #
a=a-9     ; # Input muutos   #
endfunction                  #
# ____________________________ #
# Kokeilu 1                  #
a=1 ; b=2 ; c=koe(a,b);      #
# ____________________________ #
# Kokeilu 2                  #
#a=1 ; b=2 ; [a,b,c]=koe(a,b);#
# ____________________________ #
printf("a=%3.1f b=%3.1f c=%3.1f\n",a,b,c)
printf("class(c) = %3.1f\n",c)
# ____________________________ #
# Kokeilu 3                  #
#disp(erotus) # Ei output!    #
```

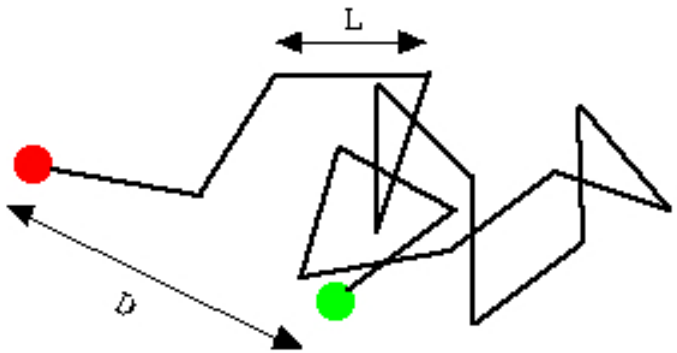
- Tehdään kokeilut 1, 2 ja 3
- Mitä tuloksista opitaan?

Aliohjelmat

python & octave Kuva: @www2.ess.ucla.edu

- Laadi aliohjelma, joka laskee Rayleigh'n testi parametrin

$$z(f) = \frac{1}{n}\left[\left(\sum_{\mathrm{i}=1}^{\mathrm{n}} \cos\theta_i\right)^2 + \left(\sum_{\mathrm{i}=1}^{\mathrm{n}} \sin\theta_i\right)^2\right]$$



aikapisteille $t_1, t_2, ..t_{\mathrm{n}}$ frekvenssillä $f = 1$, missä vaihekulmat ovat $\theta_{\mathrm{i}} = 2\pi f t_i$, sekä antaa aikapisteiden määrän n. Oleta, että aikapisteet ovat jakautuneet satunnaisesti välille $[0, 10]$.

Input: `t` $= t_1, t_2, \ldots, t_n =$ aikapisteet

Input: `f` $= f =$ Frekvenssi

Output: `z` $= z(f) =$ Testi parametri

Output: `n` $=$ Aikapisteiden määrä

- Random Walk: $z = |\bar{R}|^2/n$, missä $\bar{R} = \sum \bar{r}_i$ ja $\bar{r}_i = [\cos\Theta_i, \sin\Theta_i] \equiv$ Kuvassa $L = 1$ ja $D = |\bar{R}|$
- Kotisivulta: `Psub2.py` ja `Osub2.m`

Aliohjelmat

```
# Komenttirivi: Tama on python ohjelmani Psub2.py
import numpy as np                  # numpy importoitu
def rayleightest( t, f):            # Funktio alkaa
    pi=np.pi; x=2.0*pi*f*t ; n=len(t)
    z=(1.0/n)*(sum(np.cos(x))**2+sum(np.sin(x))**2)
    return n,z
# ____________________________________________________________
import os ; os.system('clear')      # Tyhjennetaan naytto
t=10.*np.random.sample(10)          # Satunnaisajat 0 < t_i < 10 (n=10)
f=1.                                # Frekvenssi on f=1.
n,z=rayleightest(t,f)               # Rayleigh testin n ja z(f)
print('n=',n)                       # Tarkistus
print('z=',z)                       # Tarkistus
m=rayleightest(t,f)                 # Koe: Kaytetaan vain yhta muuttujaa
print('m=',m)                       # Tulos: Antaa MOLEMPIEN arvot
print('type(m)= ',type(m))          # Tuple, s.o arvoja ei voi muuttaa
```

- Tulos on aina erilainen, koska aina arvotaan uudet aikapisteet

```
n= 10
z= 0.310150653761
m= (10, 0.31015065376068091)
type(m)=  <class 'tuple'>
```

- Mitä tuloksista opitaan?

Aliohjelmat

```
# Kommenttirivi: Tama on octave ohjelmani Osub2.m
function [n, z] = rayleightest (t, f)
    n=length(t) ;
    x=2.*pi*f*t ;
z=(1./n)*(sum(cos(x))^2 + sum(sin(x))^2);
endfunction
# ____________________________________________________________
clear ; clc                          # Poistetaan ... Tyhjennetaan ...
t=rand(10,1)*10.D0 ;                 # Satunnaisajat 0 < t_i < 10 (n=10)
f=1. ;                               # Frekvenssi on f=1.
[n, z]=rayleightest(t,f) ;           # Rayleigh testin n ja z(f)
printf("n = %f\n",n)                 # Tarkistus
printf("z = %f\n",z)                 # Tarkitus
m=rayleightest(t,f);                 # Koe: Kaytetaan vain yhta muuttujaa
printf("m = %f\n",m)                 # Tulos: antaa VAIN ensimmaisen arvon
printf("class(m) = %s\n",class(m)) # String
```

- Tulos on aina erilainen, koska aina arvotaan uudet aikapisteet

```
n = 10.000000
z = 0.283295
m = 10.000000
class(m) = double
```

- Mitä tuloksista opitaan?

Laskuharjoitusten aliaohjelmat, kuva: @www.pinterest.com

- **Laskuharjoitus:** Tee ensin **python** aliohjelma `tau=aliohjelma1(t,f)` joka laskee

$$\tau = \frac{1}{4\pi f} \mathrm{atan}\left[\frac{\sum_{i=1}^{n} \sin(4\pi f t_i)}{\sum_{i=1}^{n} \cos(4\pi f t_i)}\right]$$

Input (`aliohjelma1`)
– Havaintoajat `t`, jotka ovat
$t_1 = 1$, $t_2 = 2$ ja $t_3 = 3$ (`n` = 3)
– Frekvenssi on `f` = $f = 1.41$

Output (`aliohjelma1`)
– Arvo muuttujalle `tau` = $\tau \approx 0.0496$

- Tee sitten **python** aliohjelma `z1,z2=aliohjelma2(t,ydot,f,tau)` joka laskee

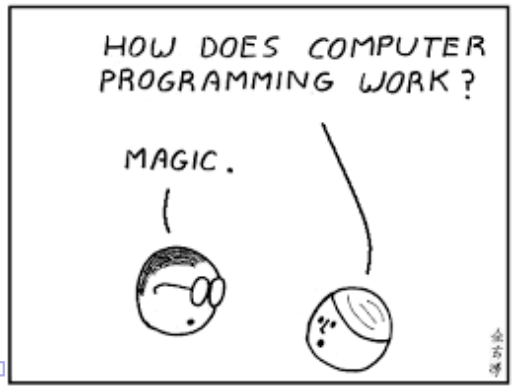
$$z_1(f) = \{\sum_{i=1}^{n} y_i' \cos[2\pi f(t_i - \tau)]\}^2$$

$$z_2(f) = \{\sum_{i=1}^{n} y_i' \sin[2\pi f(t_i - \tau)]\}^2$$

Input (`aliohjelma2`)
– Havaintoajat `t`, jotka ovat
$t_1 = 1$, $t_2 = 2$ ja $t_3 = 3$ (`n` = 3)
– Havaintoarvot `ydot`, jotka ovat
$y_1' = 4$, $y_2' = 5$ ja $y_3' = 6$ (`n` = 3)
– Frekvenssi on `f` = $f = 1.41$
– Muuttuja `tau` = $\tau \approx 0.0496$

Output (`aliohjelma2`)
– Arvo muuttujalle `z1` = $z_1(f) \approx 1.148$
– Arvo muuttujalle `z2` = $z_2(f) \approx 11.856$

- On siis laadittava aliohjelma rakenne, jossa on oikeassa järjestyksessä `def`, sisennys, `return`, Output
- Käytä mallina pari sivua aiemmin näytettyä `Psub2.py`



Laskuharjoitusten aliaohjelmat, kuva: @www.pinterest.com

- **Laskuharjoitus:** Tee ensin **octave** aliohjelma `[tau]=aliohjelma1(t,f)` joka laskee

$$\tau = \frac{1}{4\pi f} \operatorname{atan}\left[\frac{\sum_{i=1}^{n} \sin(4\pi f t_i)}{\sum_{i=1}^{n} \cos(4\pi f t_i)}\right]$$

Input (`aliohjelma1`)
– Havaintoajat `t`, jotka ovat
$t_1 = 1$, $t_2 = 2$ ja $t_3 = 3$ (`n` = 3)
– Frekvenssi on `f` = $f = 1.41$

Output (`aliohjelma1`)
– Arvo muuttujalle `tau` = $\tau \approx 0.0496$

- Tee sitten **python** aliohjelma `[z1,z2]=aliohjelma2(t,ydot,f,tau)` joka laskee

$$z_1(f) = \{\sum_{i=1}^{n} y_i' \cos[2\pi f(t_i - \tau)]\}^2$$

$$z_2(f) = \{\sum_{i=1}^{n} y_i' \sin[2\pi f(t_i - \tau)]\}^2$$

Input (`aliohjelma2`)
– Havaintoajat `t`, jotka ovat
$t_1 = 1$, $t_2 = 2$ ja $t_3 = 3$ (`n` = 3)
– Havaintoarvot `ydot`, jotka ovat
$y_1' = 4$, $y_2' = 5$ ja $y_3' = 6$ (`n` = 3)
– Frekvenssi on `f` = $f = 1.41$
– Muuttuja `tau` = $\tau \approx 0.0496$

Output (`aliohjelma2`)
– Arvo muuttujalle `z1` = $z_1(f) \approx 1.148$
– Arvo muuttujalle `z2` = $z_2(f) \approx 11.856$

- On siis laadittava aliohjelma rakenne, jossa on oikeassa järjestyksessä `function`, `endfunction`, Output
- Käytä mallina pari sivua aiemmin näytettyä `Osub2.m`