

Mappeoppgave 1

Beskrivelse

Les oppgaveteksten nøye. Se hvordan man leverer oppgaven [her](#) og [her](#). Husk at den skal leveres både som jupyter-fil og som PDF. Kommenter kodene du skriver i alle oppgaver og vær nøye på å definere aksene mm i figurer. I noen av oppgavetekstene står det hint, men det betyr ikke at de ikke kan løses på andre måter

For å hente denne filen til Jupyter gjør du slik:

1. Åpne et "terminalvindu"

2. Gå til hjemmeområdet ditt

```
[user@jupty02 ~]$ cd
```

3. Lag en ny mappe på ditt hjemmeområde ved å skrive inn i terminalvinduet

```
[user@jupty02 ~]$ mkdir SOK-1003-eksamen-2022-mappe1
```

4. Gå så inn i den mappen du har laget ved å skrive

```
[user@jupty02 ~]$ cd SOK-1003-eksamen-2022-mappe1
```

5. Last ned kursmateriellet ved å kopiere inn følgende kommando i kommandovinduet:

```
[user@jupty02 sok-1003]$ git clone https://github.com/uit-sok-1003-h22/mappe/  
</ol>
```

Oppgi gruppenavn m/ medlemmer på epost o.k.aars@uit.no innen 7/10, så blir dere satt opp til tidspunkt for presentasjon 19/10.

Bruk så denne filen til å gjøre besvarelsen din. Ved behov; legg til flere celler ved å trykke "b"

Oppgavene

Oppgave 1 (5 poeng).

a) Lag en kort fortelling i en python kode som inkluderer alle de fire typer variabler vi har lært om i kurset. Koden skal kunne kjøres med print(). Koden burde inneholde utregninger av elementer du har definert

```
In [41]: #Lager de 4 ulike variablene
Nordmenn_liker_ferie=True #bool (true/false)
Antall_nordmenn_på_ferie=8 #int (heltall)
Pris_for_middag_eur=9.8 #float (desimaltall)
Nordmenn_string = "De koser seg på ferie og skal spise ute." #str (skrift)

#Printer ut hvilken type variabel de er for å fremvise dette.
print(type(Nordmenn_liker_ferie))
print(type(Antall_nordmenn_på_ferie))
print(type(Pris_for_middag_eur))
print(type(Nordmenn_string))

#Printer historien, må gjøre om til str for å kunne printe sammen med and
print("At nordmenn liker ferie er "+str(Nordmenn_liker_ferie)+" "+"Total
print("Middagen kostet "+str(Pris_for_middag_eur)+" EUR per person. "+"To

<class 'bool'>
<class 'int'>
<class 'float'>
<class 'str'>
At nordmenn liker ferie er True. Totalt dro det 8 nordmenn på ferie. De k
oser seg på ferie og skal spise ute.
Middagen kostet 9.8 EUR per person. Totalt betalte de 78.4 EUR.
```

Oppgave 2 (10 poeng).

Leieprisene i landet har steget de siste månedene. Ved å bruke realistiske tall

a) Lag tilbuds og etterspørselsfunksjoner for leie av bolig (Bruk av ikke-lineære funksjoner belønnes).

Definer funksjonene slik at det er mulig å finne en likevekt

```
In [5]: # Funksjonen viser hvordan leieprisen varierer utifra hvor mange boliger
# når utleier ønsker å leie ut til ca. 8000 kr.
def tilbud2(x):
    a = 800000/(100+x)
    return a

# Funksjonen viser hvordan prisen man er villig til å betale utifra etter
# når man i ugangspunktet ønsker å leie for ca. 7000 kr.

def etterspørsel2(x):
    a = 700000/(100-x)
    return a
```

b) Vis at disse er henholdsvis fallende og stigende, ved bruk av

- Regning
- figurativt (matplotlib) Husk å markere aksene tydelig og at funksjonene er definert slik at linjene krysser

```
In [35]: # Tilbudsprisen er avtagende, da leieprisen synker utifra hvor mange boli
# er tilgjengelig for utleie.
print("Prisen synker utifra hvor mange som leieboliger som er tilgjengeli
print(tilbud2(1))
print(tilbud2(3))
print(tilbud2(13))

#Følgende lager et mellomrom for å skille tilbud og etterspørsel.
print(" ").

# Prisen man er villig til å betale øker desto flere som ønsker å leie.,
print("Prisen man er villig til å betale øker utifra hvor mange som ønske
print(etterspørsel2(1))
print(etterspørsel2(3))
print(etterspørsel2(13))

#Følgende lager et mellomrom for å skille tilbud og etterspørsel.
print(" ").
print(" ").

#pakker
from matplotlib import pyplot as plt
import numpy as np

#plot
fig, ax = plt.subplots() #navngir plot
x = np.arange(0,11,1) #x verdier vi ønsker å undersøke
ax.tilbud = plt.plot(x, tilbud2(x), label="Tilbud") #tilbudsfunksjon
ax.etterspørsel = plt.plot(x, etterspørsel2(x), label="Etterspørsel") #et
ax = plt.scatter(6.65, 7502, color="b") #scatter for likevekt
ax = plt.legend(loc="lower right") #legendeposisjon
ax = plt.ylabel("Pris") #y label
```

Prisen synker utifra hvor mange som leieboliger som er tilgjengelig. (tilbud).

7920.792079207921

7766.9902912621355

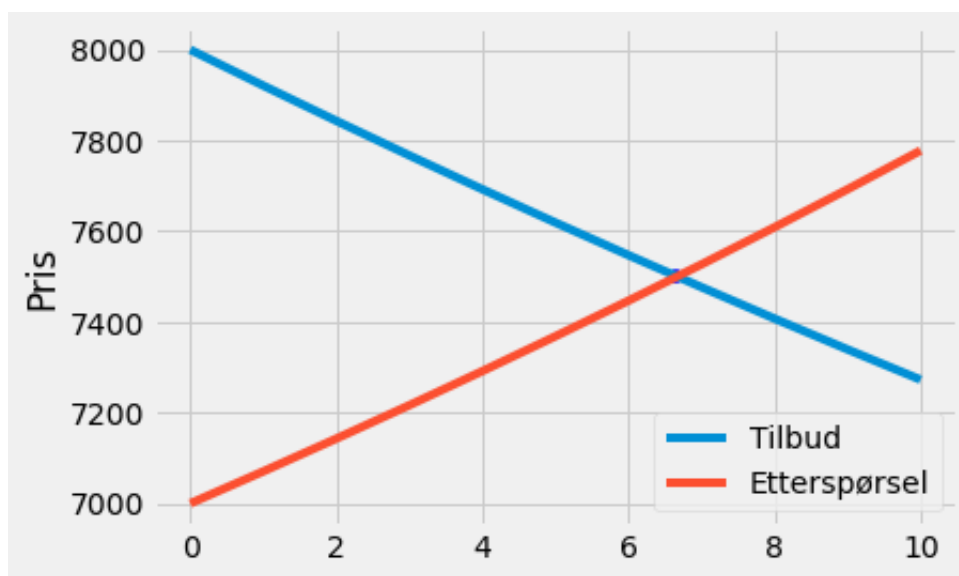
7079.646017699115

Prisen man er villig til å betale øker utifra hvor mange som ønsker bolig. (etterspørsel).

7070.707070707071

7216.494845360825

8045.977011494253



c) Kommenter funksjonene og likevekten. Vis gjerne figurativt hvor likevekten er ved bruk av scatter

```
In [7]: print("""Funksjonen viser hvordan prisen endrer seg utifra tilbud (tilgjenge og etterspørsel (hvor mange som ønsker bolig)). Scatterpointet (likevekten) er på 6.65, 7502. Det vil si at tilbudet møte pris 7502 i månedsleie, når 6.65 personer ønsker å leie bolig, og 6.65 personer ønsker å leie ut."
```

Funksjonen viser hvordan prisen endrer seg utifra tilbud (tilgjengelighet) og etterspørsel (hvor mange som ønsker bolig). Scatterpointet (likevekten) er på 6.65, 7502. Det vil si at tilbudet møter etterspørselen på pris 7502 i månedsleie, når 6.65 personer ønsker å leie bolig, og 6.65 personer ønsker å leie ut.

Oppgave 3 (15 poeng)

SSB har omfattende data på befolkningsutvikling

(<https://www.ssb.no/statbank/table/05803/tableViewLayout1/>). Disse dataene skal du bruke i de neste deloppgavene.

a) lag lister av følgende variabler: "Befolkning 1. januar", "Døde i alt", "Innflyttinger" og "Utflyttinger". Velg selv variabelnavn når du definerer dem i python. Første element i hver liste skal være variabelnavnet. Bruk tall for perioden 2012-2021. Lag så en liste av disse listene. Du kan kalle den "ssb".

Hint: når du skal velge variabler på SSB sin nettside må du holde inne ctrl for å velge flere variabler.

```
In [11]: #lager lister
år = ["år", 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019, 2020, 2021]
befolkning_1_januar = ["befolkning_1_januar", 4985870, 5051275, 5109056,
døde_i_alt = ["døde_i_alt", 41992, 41282, 40394, 40727, 40726, 40774, 408
innflyttinger = ["innflyttinger", 78570, 75789, 70030, 67276, 66800, 58192
utflyttinger = ["utflyttinger", 31227, 35716, 31875, 37474, 40724, 36843,

ssb = [år, befolkning_1_januar, døde_i_alt, innflyttinger, utflyttinger]
```

b) konverter "ssb" til en numpy matrise og gi den et nytt navn

```
In [12]: import numpy as np
#gjør listen av lister ssb til np matrise.
ssb_np = np.array(ssb)

#printer denne matrisen
print(ssb_np)

[['år' '2012' '2013' '2014' '2015' '2016' '2017' '2018' '2019' '2020'
 '2021']
 ['befolkning_1_januar' '4985870' '5051275' '5109056' '5165802' '5213985'
 '5258317' '5295619' '5328212' '5367580' '5391369']
 ['døde_i_alt' '41992' '41282' '40394' '40727' '40726' '40774' '40840'
 '40684' '40611' '42002']
 ['innflyttinger' '78570' '75789' '70030' '67276' '66800' '58192' '52485'
 '52153' '38071' '53947']
 ['utflyttinger' '31227' '35716' '31875' '37474' '40724' '36843' '34382'
 '26826' '26744' '34297']]
```

c) Putt alle tallene inn i en egen matrise og konverter disse til int

```
In [21]: #velger alle verdier etter rad en for å fjerne str verdiene.
ssb_np_2 = ssb_np[:, 1:].
#gjør matrisen om til int.
ssb_np_int = ssb_np_2.astype("int").
#printer denne ut
print(ssb_np_int)
```

```
[ [ 2012      2013      2014      2015      2016      2017      2018      2019      2020
    2021]
  [4985870 5051275 5109056 5165802 5213985 5258317 5295619 5328212 5367580
    5391369]
  [ 41992   41282   40394   40727   40726   40774   40840   40684   40611
    42002]
  [ 78570   75789   70030   67276   66800   58192   52485   52153   38071
    53947]
  [ 31227   35716   31875   37474   40724   36843   34382   26826   26744
    34297]]
```

d) vis befolkningsutviklingen grafisk for de gjeldene årene ved bruk av matplotlib, og mer spesifikt "fig, ax = plt.subplots()". Vis befolkning på y-aksen i millioner

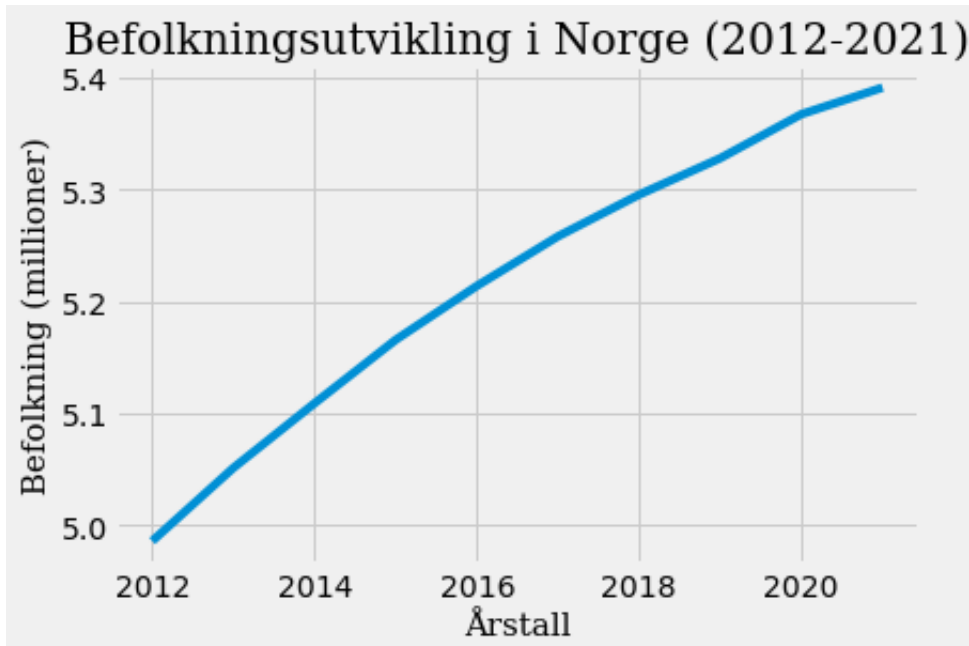
```
In [22]: #pakker
from matplotlib import pyplot as plt #pakke pyplot
import matplotlib.style as style #pakke til tema

#font oppslag
font1 = {'family':'serif','color':'black','size':20} #definerer font
font2 = {'family':'serif','color':'black','size':15} #definerer font

#definerer aksene for å lettere kunne gjøre endringer senere.
x = ssb_np_int[0, :] #årstall
y = ssb_np_int[1, :]/1000000 #Befolkning #gjør befolkning til millioner

#plotter figur
fig, ax = plt.subplots() #definerer figur

ax = plt.style.use('fivethirtyeight') #tema
ax = plt.plot(x, y) #definerer hva som skal plottes
ax = plt.title("Befolkningsutvikling i Norge (2012-2021)", fontdict = font1)
ax = plt.ylabel("Befolkning (millioner)", fontdict = font2) #y akse tittel
ax = plt.xlabel("Årstall", fontdict = font2) #x akse tittel
```



e) Lag det samme plottet ved bruk av oppslag. Hva er fordelen med dette?

```
In [23]: #lager oppslag "norge"
norge = dict()
norge["år"] = ssb_np_int[0, :] #lager oppslagsord til kolonneverdiene
norge["befolkning_i_mill"] = ssb_np_int[1, :]/1000000 #lager oppslagsord

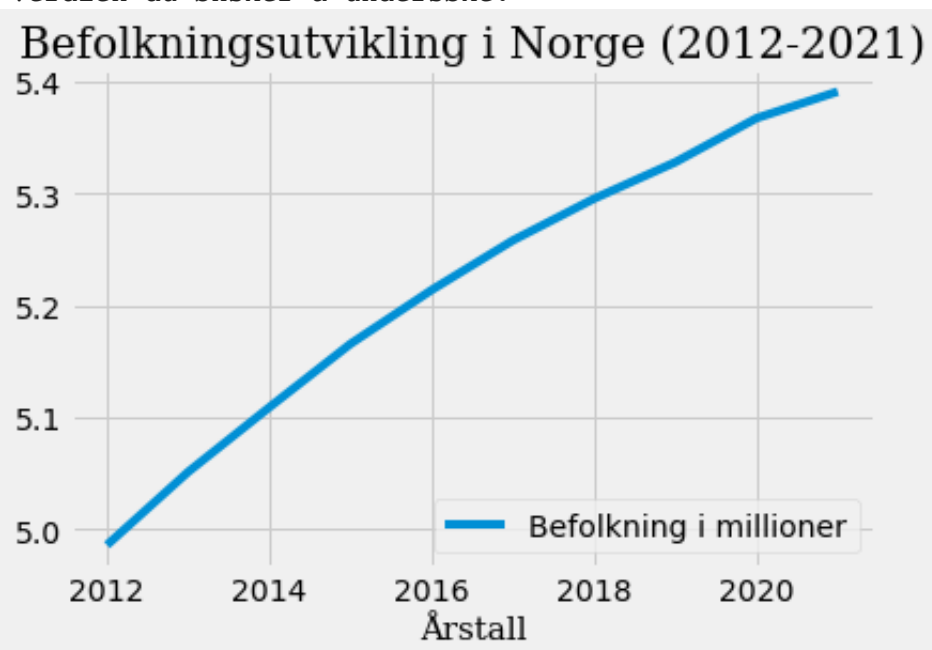
#definerer x/y verdi for å lett kunne foreta endringer
x2 = norge["år"] #x
y2 = norge["befolkning_i_mill"] #y

#plotter figur
fig, ax2 = plt.subplots() #definerer figur

ax2 = plt.style.use('fivethirtyeight') #tema
ax2 = plt.plot(x2, y2, label = "Befolkning i millioner") #definerer hva s
ax2 = plt.title("Befolkningsutvikling i Norge (2012-2021)", fontdict = fo
ax2 = plt.xlabel("Årstall", fontdict = font2) #x akse titel
ax2 = plt.legend(loc="lower right") #legende posisjon

print("""Fordelen med å bruke oppslag er at du slipper å huske hvilket ra
verdien du ønsker å undersøke.""").
```

Fordelen med å bruke oppslag er at du slipper å huske hvilket rad eller kolonne som tilhører verdien du ønsker å undersøke.



f) Hva er den relative befolkningstilveksten utenom fødsler (dvs. innvandring/utvandring)? Definer en ny array og legg den til i oppslaget du laget i oppgaven tidligere. Kall den "rel_immigration". Plot denne sammen med grafen du laget i (d).

```

In [42]: #lager ny var
rel_immigration = ssb_np_int[3, :] / ssb_np_int[4, :] #ny var

#legger til rel immigration til oppslag norge
norge["rel_immigration"] = rel_immigration #legger til i oppslaget

#definerer y verdi
y3 = norge["rel_immigration"] #definerer ny y verdi

#plotter figur
fig, ax2 = plt.subplots() #definerer figur

ax2 = plt.style.use('fivethirtyeight') #tema
ax2 = plt.plot(x2, y2, label = "Befolkning i millioner") #befolkning i mi
ax2 = plt.title("Befolkningsutvikling i Norge (2012-2021)", fontdict = fo
ax2 = plt.xlabel("Årstall", fontdict = font2) #x akse titel
ax2 = plt.plot(x2, y3, label = ""Relativ befolkningstilvekst utenom føds
(innvandring/utvandring)"" ) #rel immigrasjon plot
ax2 = plt.legend(loc="right") #legendeposisjon
ax2 = plt.show #vise plot

```



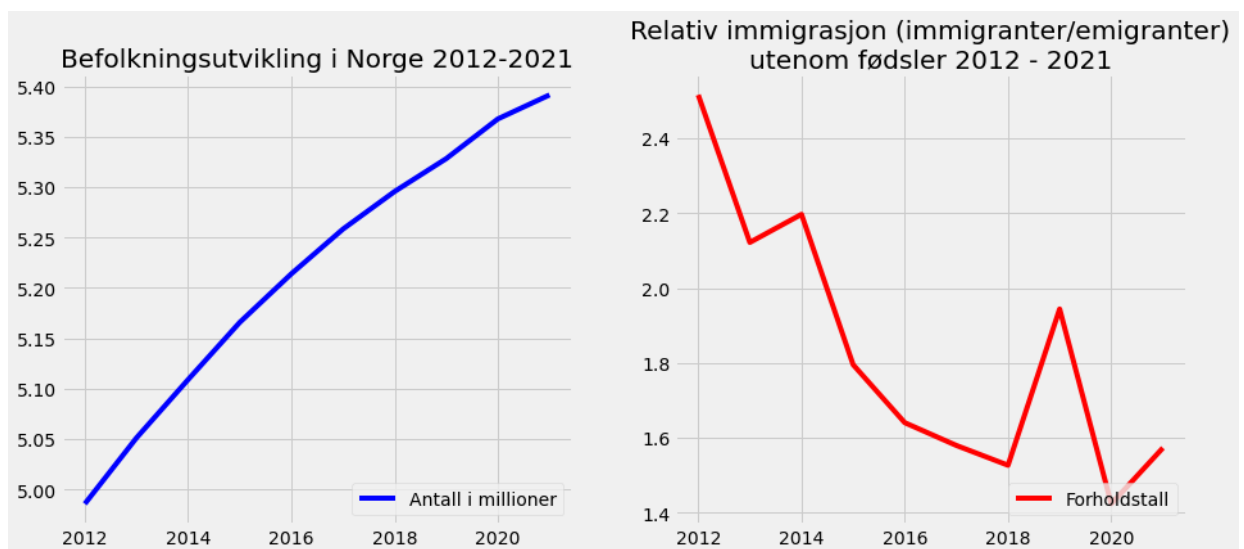
g) ekstrapoeng. Kan plote de samme tallene (dvs "rel_immigration" og "befolkning" sammen med år) i to figurer ved siden av hverandre ved bruk av "fig, (ax1, ax2) = plt.subplots(1, 2)". Gi grafene ulik farge


```
In [43]: #lager panel
fig2,(p1, p2)=plt.subplots(1,2, figsize=(14, 6))

#definerer x og y for å lett kunne foreta endringer.
xv = norge["år"]
yv1 = norge["befolkning_i_mill"]
yv2 = norge["rel_immigration"]

#plot 1
p1.plot(xv, yv1, color="blue", label = "Antall i millioner") #plot
p1.title.set_text("Befolkningsutvikling i Norge 2012-2021") #tittel
p1.legend(loc='lower right', frameon=True) #legend posisjon
#plot 2
p2.plot(xv, yv2, color="red", label="Forholdstall") #plot
p2.title.set_text("Relativ immigrasjon (immigranter/emigranter)
utenom fødsler 2012 - 2021") #tittel
p2.legend(loc='lower right', frameon=True) #legend posisjon
```

```
Out[43]: <matplotlib.legend.Legend at 0x7f5f05a5b520>
```



Oppgave 4 (20 poeng).

Et lån består som regel av et månedlig terminbeløp. Dette beløpet er summen av avdrag (nedbetalingen på lånet) og renter. Vi antar månedlig forrenting i alle oppgavene. Dvs. at det er 12 terminer i hvert år.

a) Lag en funksjon som regner ut hvor mye lånet "x" koster deg i renteutgifter for "t" terminer med årlig rente "r" for et serielån.

Siden dette er et serielån, så vil avdragene være like hver måned men renteutgiftene reduseres i takt med avdragene. Renteutgiftene for en gitt termin "t" vil derfor være den årlige renten "r" (delt på antall forrentinger "f") på gjenværende beløp på det tidspunktet. $\text{renteutgifter}_{\{t\}} = (x - a \cdot t) \cdot \{r/f\}$

Siden vi er ute etter den totale kostnaden i svaret, må du summere renteutgiftene over alle terminer, det vil si $\sum_{t=1}^N (x - a \cdot t) \cdot \{r/f\}$

Hint: siden terminbeløpet varierer for hver måned (pga at rentene endres), må alle enkeltperioder summeres. Det kan være nyttige å bruke funksjonen `np.arange()` til dette.

```
In [1]: import numpy as np

#Inputs: x = lånesum, T = terminer, r = årlig rente, f = antall forentni

def renteutgift(x, T, f, r):
    a = x / T
    renteutgifter = []

    for t in range(T): #For hver termin t opp til totale antall terminer
        rente_t = (x - a*t)*r/f
        renteutgifter.append(rente_t) #append legger sammen listene.
    return sum(renteutgifter)
```

b) regn ut hvor mye lånet koster deg med henholdsvis 10, 20 og 30 års tilbakebetaling. Anta 1 000 000 kr lånebeløp med 3% rente

```
In [3]: print("Et lån på 1 000 000 kr med en rente på 3%, koster følgende med 10
print(renteutgift(1000000, 120, 12, 0.03))
print(" ")
print("Et lån på 1 000 000 kr med en rente på 3%, koster følgende med 20
print(renteutgift(1000000, 240, 12, 0.03))
print(" ")
print("Et lån på 1 000 000 kr med en rente på 3%, koster følgende med 30
print(renteutgift(1000000, 360, 12, 0.03))
```

Et lån på 1 000 000 kr med en rente på 3%, koster følgende med 10 års til
bakebetaling:
151250.0

Et lån på 1 000 000 kr med en rente på 3%, koster følgende med 20 års til
bakebetaling:
301250.0

Et lån på 1 000 000 kr med en rente på 3%, koster følgende med 30 års til
bakebetaling:
451250.0

c) Vis hva det samme lånet koster som annuitetslån, dvs differansen mellom alle
terminbeløp og lånebeløp.

Annuitetslån gir like terminbeløp hver måned, men renten utgjør en større del av
dette beløpet i starten. Terminbeløpet for et annuitetslån er definert ved formelen: $T = x \cdot \frac{r/f}{1 - (1 + (r/f))^{-t}}$, hvor x =lånebeløp, r = årlig rente, t = terminer, f =
antall forrentinger

```
In [53]: def terminbeløp(x, r, t, f):
          return x*((r/f)/(1-(1+(r/f))**(-t)))

print("Et lån på 1 000 000 kr med en rente på 3%, koster følgende med 10
print(str(int(12*10*terminbeløp(1000000, 0.03, 120, 12)-1000000)) + " kr"
print(" ")
print("Et lån på 1 000 000 kr med en rente på 3%, koster følgende med 20
print(str(int(12*20*terminbeløp(1000000, 0.03, 240, 12)-1000000)) + " kr"
print(" ")
print("Et lån på 1 000 000 kr med en rente på 3%, koster følgende med 30
print(str(int(12*30*terminbeløp(1000000, 0.03, 360, 12)-1000000)) + " kr"
```

Et lån på 1 000 000 kr med en rente på 3%, koster følgende med 10 års til
bakebetaling:
158728 kr

Et lån på 1 000 000 kr med en rente på 3%, koster følgende med 20 års til
bakebetaling:
331034 kr

Et lån på 1 000 000 kr med en rente på 3%, koster følgende med 30 års til
bakebetaling:
517774 kr

d) Vis hvordan utviklingen i rentekostnader og avdrag på terminer for serielån grafisk ved hjelp av stackplot funksjonen i matplotlib. Anta et bankinnskudd $x = 1\,000\,000$ kr, årlig rente $r=3\%$ og antall terminer $t = 240$ (det vil si 2 år). Siden vi må vise utviklingen per termin, husk at "t" også definerer hvilken måned vi er i. Dvs, hvis $t=15$, har det gått 1 år og 3 mnd med terminer. Se forøvrig relevante formel i oppgave (a).

Hint1: Siden avdragene er like for alle måneder, kan det være lurt å definere det månedlige avdraget som en liste og gange det med antall perioder. **Hint2:**

Siden vi er ute etter både rentekostnader og avdrag hver for seg, kan det være lurt å definere en funksjon for hver av dem.

```
In [45]: #Lager funksjoner.

def renteutgift2(x, T, f, r): #liste over nedbetalingene
    a = x / T
    renteutgifter = []

    for t in range(T): #For hver termin t opp til totale antall terminer
        rente_t = (x - a*t)*r/f
        renteutgifter.append(rente_t)
    return renteutgifter

def avdrag(x, T): #avdrag
    a = x/T
    return a

def avdrag2(x, T): #i liste form
    b = []
    a = x/T
    b.append(a)
    return b
```

```
In [48]: import matplotlib.pyplot as plt #pakke pyplot

nedbetaling_av_renter = np.array(renteutgift2(1000000, 240, 12, 0.03)) #ar
avdrag240 = avdrag2(1000000, 240) * 240 #lager en liste på med 240 tall (
avdragsbetaling = np.array(avdrag240) #gjør liste til array.

x = np.arange(0, 240, 1) #definerer ønskede x verdier, i vårt tilfelle 24
y1 = nedbetaling_av_renter #definerer y1
y2 = avdragsbetaling #definerer y2
plt.stackplot(x, y2, y1, colors=['green', 'mediumseagreen'], labels = ["
plt.style.use('fivethirtyeight') #tema
plt.xlabel("Terminer (Måneder)") #x akse label
plt.ylabel("Månedsbetaling") #y akse label
plt.title("Månedsbetaling for tilbakebetaling
av lån på 1 million med 3% rente") #tittel
plt.legend(loc='lower left', frameon=True) #legend posisjon
```

```
Out[48]: <matplotlib.legend.Legend at 0x7f5f0413ef40>
```

