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/`

   Oppgi gruppenavn m/ medlemmer på epost ole.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 [11]:
```
antall_boller = 1000
boller_vekt = 80.0
boller_er_runde = True
frakt_selskap = "Ole Kristians trucking service"
```

```
print(f"Kåre skal frakte bollene sine fra Tromsø til Nordkjosbotn. Dette var ingen
f"Det er nemlig {boller_er_runde} at boller er runde, og da må det litt planlegging
f"Han har {antall_boller} boller og hver bolle veier {boller_vekt}g. \n"
f"Vekten på alle bollene sine blir da {(antall_boller*boller_vekt)/1000} kg. \n"
f"Dette ble for mye vekt ettersom {frakt_selskap} kun kan frakte {(antall_boller*b
```

```
Kåre skal frakte bollene sine fra Tromsø til Nordkjosbotn. Dette var ingen lett sa
k.
Det er nemlig True at boller er runde, og da må det litt planlegging til.
Han har 1000 boller og hver bolle veier 80.0g.
Vekten på alle bollene sine blir da 80.0 kg.
Dette ble for mye vekt ettersom Ole Kristians trucking service kun kan frakte 79.9
99 kg.
```

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

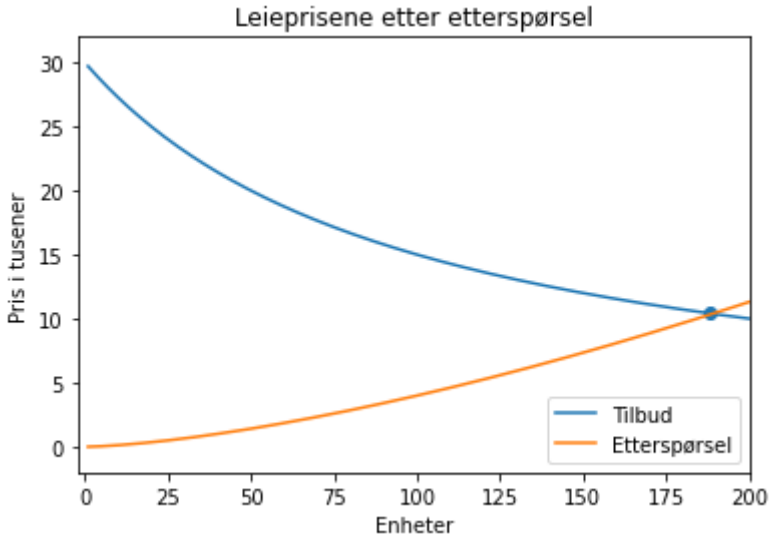
In [82]:
```
def demand(x):
    return (x**1.5)*(1/250)

def supply(x):
    return 3000/(100+x)
```

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

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

In [78]:
```
from matplotlib import pyplot as plt
import numpy as np

x = np.linspace(1,1000,1000)

plt.plot(x,supply(x),label="Tilbud")

plt.plot(x,demand(x),label="Etterspørsel")

plt.legend(loc="lower right")

plt.ylabel("Pris i tusener")
plt.xlabel("Enheter")
plt.title("Leieprisene etter etterspørsel")

plt.ylim((-2,32))
plt.xlim((-2,200))

plt.scatter(188, 10.5)

plt.show

print(supply(np.arange(0,10)))
print(demand(np.arange(0,10)))
```

```
[30.         29.7029703  29.41176471 29.12621359 28.84615385 28.57142857
 28.30188679 28.03738318 27.77777778 27.52293578]
[0.         0.004      0.01131371 0.02078461 0.032      0.04472136
 0.05878775 0.07408104 0.09050967 0.108     ]
```



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

Her har du likevekten på 188 enheter og 10,5 tusen i pris.Funksjonen viser oss at det er få tilgjengelige boliger til en lav pris, men samtidig høy etterspørsel på dem. Samtidig ser vi at de boligene med høy leiepris har lavere etterspørsel.

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 [53]:
```
year = ["year", 2012, 2013, 2014, 2015, 2015, 2017, 2018, 2019, 2020, 2021]
pop = ["pop", 4985870, 5051275, 5109056, 5165802, 5213985, 5258317, 5295619, 53282
dead = ["dead", 41992, 41282, 40394, 40727, 40726, 40774, 40840, 40684, 40611, 420
movein = ["movein", 78570, 75789, 70030, 67276, 66800, 58192, 52485, 52153, 38071,
moveout = ["moveout", 31227, 35716, 31875, 37474, 40724, 36843, 34382, 26826, 2674
ssb = [year, pop, dead, movein, moveout]
```

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

In [54]:
```
np_ssb = np.array(ssb)
```

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

In [55]:
```
ssb_numbers = np.array(np_ssb[:,1:], dtype=int)
```

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

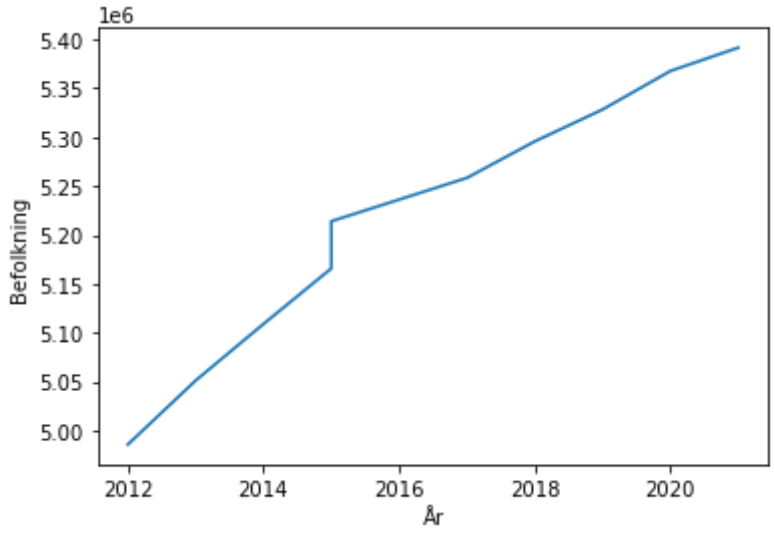
In [56]:
```
f, ax = plt.subplots()
ax.plot(year[1:], pop[1:])
ax.set_ylabel("Befolkning i milioner")
ax.set_xlabel("År")
ax.set_title("Befolkningsutvikling i perioden 2012-2021")
```

Out[56]:
```
Text(0.5, 1.0, 'Befolkningsutvikling i perioden 2012-2021')
```

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

In [57]:
```
dictionary = {"year":year[1:], "pop":pop[1:]}

plt.ylabel("Befolkning")
plt.xlabel("År")
plt.plot(dictionary["year"], dictionary["pop"])

#Fordelen med et oppslag er at det er mer lesbart og raskere.
```

Out[57]:
```
[<matplotlib.lines.Line2D at 0x7f10ec2737c0>]
```

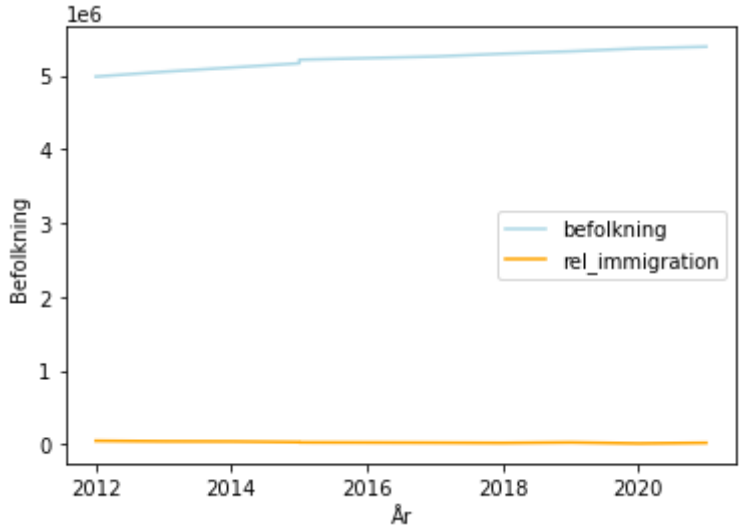


f) Hva er den relative befolkningstilveksten utenom fødsler (dvs. innvandring/utvandring)?

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).

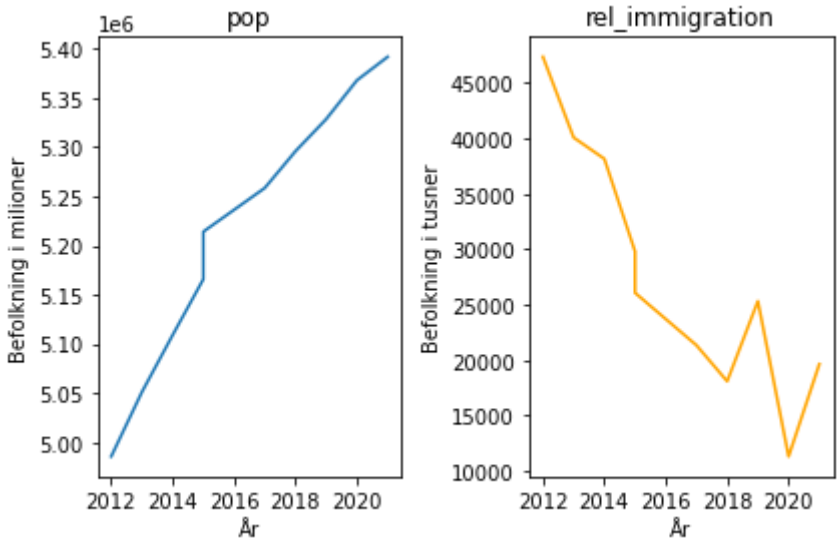
In [58]:
```
np_movein = np.array(movein[1:])
np_moveout = np.array(moveout[1:])

rel_immigration = np_movein-np_moveout

dictionary["rel_immigration"] = rel_immigration

plt.ylabel("Befolkning")
plt.xlabel("År")
plt.plot(dictionary["year"], dictionary["pop"], label="befolkning", color="lightblu
plt.plot(dictionary["year"], dictionary["rel_immigration"], label="rel_immigration
plt.legend(loc="best")
```

Out[58]:
```
<matplotlib.legend.Legend at 0x7f10ec2cc6a0>
```



g) ekstrapoeng. Kan plotte 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 [11]:
```
fig, (ax1, ax2) = plt.subplots(1,2)
ax1.plot(dictionary["year"], dictionary["pop"])
ax1.set_ylabel("Befolkning i milioner")
ax1.set_xlabel("År")
ax1.set_title("pop")

ax2.plot(dictionary["year"], dictionary["rel_immigration"], color="orange")
ax2.set_ylabel("Befolkning i tusner")
ax2.set_xlabel("År")
ax2.set_title("rel_immigration")

fig.tight_layout()
```



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.
$renteutgifter_{t} = (x-a*(t-1))*{r/f}$

Det vil si at renteutgiftene første termin er
$renteutgifter_{1} = (x-a*0)*{r/f}$, og andre termin er
$renteutgifter_{2} = (x-a*1)*{r/f}$ osv..

Siden vi er ute etter den totale kostnaden i svaret, må du summere renteutgiftene over alle terminer, det vil si $\sum_{t=1}^N(x-a*(t-1))*{r/f}$. Dette betyr egentlig bare
$renteutgifter_{1} + renteutgifter_{2},...,+renteutgifter_{t}$

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. Mao, det er ikke nødvendig å bruke sigma ($\sum_{t=1}^N$) i formelen til dette

In [80]:
```
def f(x, a, r, t, f):
    return (x-a*np.arange(0, t))*r/f


np.sum(f(1000000,8000,0.03,12, 12))
```

Out[80]:
```
28680.0
```

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 [77]:
```
def f(x, a, r, t, f):
    return (x-a*np.arange(0, t))*r/f


print("Kostnader etter 10 år: " + str(np.sum(f(1000000,8000,0.03,120, 120))))
print("Kostnader etter 20 år: " + str(np.sum(f(1000000,8000,0.03,240, 240))))
print("Kostnader etter 30 år: " + str(np.sum(f(1000000,8000,0.03,360, 360))))
```

```
Kostnader etter 10 år: 15720.0
Kostnader etter 20 år: 1320.0
Kostnader etter 30 år: -13080.0
```

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: $T = x*\frac{r/f}{(1-(1+(r/f))^{-t})}$, hvor x=lånebeløp, r = årlig rente, t = terminer, f= antall forrentinger

In [48]:
```
def annuitetslan(x, r, t, f):
    return x*((r/f)/(1-(1+(r/f))**-t))

print(annuitetslan(1000000, 0.03, 12, 12))
```

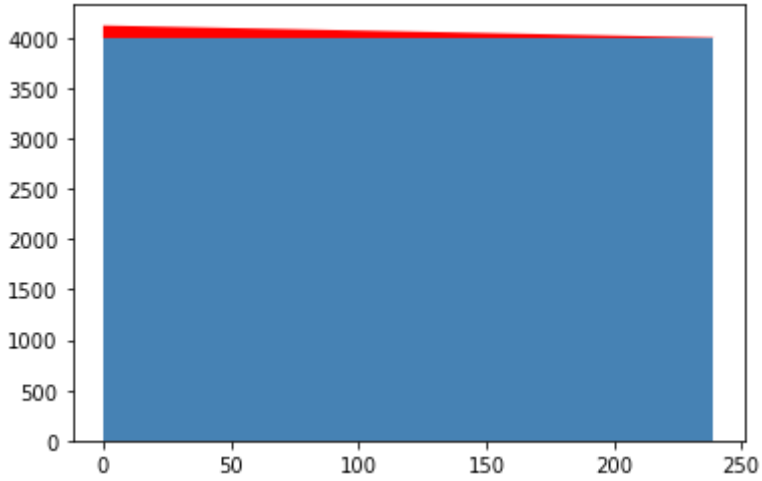
```
84693.69875849057
```

c) 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 20 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 formler 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 [81]:
```
renter = f(1000000,4000,0.03,240, 240)
terminer = range(0, 240)
avdrag = [4000]*240

plt.stackplot(terminer, avdrag, renter, colors =["steelblue", "r"])
```

Out[81]:
```
[<matplotlib.collections.PolyCollection at 0x7f10e9aeb700>,
 <matplotlib.collections.PolyCollection at 0x7f10e9aebaf0>]
```


4000
3500
3000
2500
2000
1500
1000
500
0
0
50
100
150
200
250