SOK 2030 Mappeoppgave 2 - Kode

```
# Laster inn pakker.
import numpy as np
import sympy as sp
from sympy import *
from matplotlib import pyplot as plt
```

```
# Definerer symboler.
qo, qc, q1, q2, q3, c1, c2, c3, a, b, Q = symbols("qo qc q1 q2 q3 c1 c2 c3 a b Q")
```

Oppgave 1.A

```
# Definerer den totale etterspørselen.
def demand_01(q):
    return 990-(1/60)*q

# Definerer profittfunksjonene for Olivita.
def profit_oli(qo, qc):
    return (demand_01(qo+qc)-50)*qo-3000000

# Definerer profittfunksjonen for Dr. Choice.
def profit_cho(qo, qc):
    return (demand_01(qo+qc)-50)*qc-3000000
```

```
# Deriverer profittfunksjonene til Dr. Choice mhp. qc.
der_profit_cho = diff(profit_cho(qo, qc), qc)

# Viser den deriverte til profittfunksjonene til Dr.Choice.
der_profit_cho
```

$-0.0333333333333333qc - 0.0166666666666667qo + 940$

```
# Setter den deriverte lik 0, og løser mhp. qc for å finne reaksjonsfunksjonene til Dr.Cho
sol_profit_cho = solve(der_profit_cho, qc)[0]

# Viser reaksjonsfunksjonen til Dr.Choice.
sol_profit_cho
```

$28200.0 - 0.5qo$

```
# Setter reaksjonsfunksjonen til Dr.Choice inn i profittfunksjonene til Olivita, og derive
der_profit_oli = diff(profit_oli(qo, sol_profit_cho), qo)

# Viser den deriverte til profittfunksjonen til Olivita.
der_profit_oli
```

$470.0 - 0.0166666666666667qo$

```
# Setter den deriverte lik 0, og løser mhp. qo for å finne optimalt kvantum for Olivita.
sol_q_olivita = solve(der_profit_oli, qo)[0]

# Viser optimalt kvantum for Olivita.
sol_q_olivita
```

28200.0

```
# Setter optimalt kvantum for Olivita inn i reaksjonsfunksjonen til Dr. Choice, for å finn
sol_q_drchoice = sol_profit_cho.subs({qo:sol_q_olivita})

# Viser optimalt kvantum for Dr. Choice.
sol_q_drchoice
```

14100.0

```
# Definerer etterspørselen fordelt på de to selskapene.
def demand_split(qo, qc):
    return 990-(1/60)*(qo+qc)
```

```
# Setter optimalt kvantum inn i etterspørselsfunksjonen, og finner markedsprisen.
demand_split(qo, qc).subs({qo:sol_q_olivita, qc:sol_q_drchoice})
```

285.0

```
# Finner profitten til Olivita.
profit_oli(qo, qc).subs({qo:sol_q_olivita, qc:sol_q_drchoice})
```

3627000.0

```
# Finner profitten til Dr. Choice.
profit_cho(qo, qc).subs({qo:sol_q_olivita, qc:sol_q_drchoice})
```

313500.0

Oppgave 2.A - Optimal tilpasning før fusjon

```
# Definerer den totale etterspørsel.
def demand_02(Q, a, b):
    return a-b*Q

# Definerer profittfunksjonen.
def profit_01(q1, q2, q3, c, a, b):
    return(demand_02(q1+q2+q3, a, b)-c)*q1
```

```
# Deriverer profittfunksjonen med hensyn på q1.
der_profit1_Q = diff(profit_01(q1, q2, q3, c1, a, b), q1)

# Deriverer profittfunksjonen med hensyn på q2.
der_profit2_Q = diff(profit_01(q2, q1, q3, c2, a, b), q2)

# Deriverer profittfunksjonen med hensyn på q3.
der_profit3_Q = diff(profit_01(q3, q2, q1, c3, a, b), q3)

# Viser de deriverte funksjonene.
display(der_profit1_Q)
display(der_profit2_Q)
display(der_profit3_Q)
```

$a - bq_1 - b\left(q_1 + q_2 + q_3\right) - c_1$

$a - bq_2 - b\left(q_1 + q_2 + q_3\right) - c_2$

$a - bq_3 - b\left(q_1 + q_2 + q_3\right) - c_3$

```
# Setter lik null og løser med hensyn på q1/q2/q3.
sol_01 = solve([der_profit1_Q, der_profit2_Q, der_profit3_Q],[q1, q2, q3])

# Viser resultatet.
display(sol_01[q1])
display(sol_01[q1])
display(sol_01[q1])
```

$$\frac{a - 3c_1 + c_2 + c_3}{4b}$$
$$\frac{a - 3c_1 + c_2 + c_3}{4b}$$
$$\frac{a - 3c_1 + c_2 + c_3}{4b}$$

```
# Lager en lambdify funksjon.
opt_q = lambdify((a, b, c1, c2, c3), (sol_01[q1], sol_01[q2], sol_01[q3]))

# Definerer verdier.
a_val = 175
b_val = 4
c1_val = 10
c2_val = 10
c3_val = 7

# Bruker lamdify funksjonen til å løse den optimale tilpasnigen.
q1sol, q2sol, q3sol = opt_q(a_val, b_val, c1_val, c2_val, c3_val)

# Bruker den optimale tilpasnigen for å finne profitten.
prof1 = profit_01(q1sol, q2sol, q3sol, c1_val, a_val, b_val)
prof2 = profit_01(q1sol, q2sol, q3sol, c2_val, a_val, b_val)
prof3 = profit_01(q1sol, q2sol, q3sol, c3_val, a_val, b_val)

# Skriver ut svarene.
print(f"Graff Brygghus produserer optimalt", q1sol, f"flasker, med en profitt på", prof1)
print(f"Bryggeri 13 produserer optimalt", q2sol, f"flasker, med en profitt på", prof2)
print(f"Mack Mikrobryggeri produserer optimalt", q3sol, f"flasker, med en profitt på", pro
print(f"Markedsprisen er lik", demand_02(q1sol+q2sol+q3sol, a_val, b_val))
```

```
Graff Brygghus produserer optimalt 10.125 flasker, med en profitt på 410.0625
Bryggeri 13 produserer optimalt 10.125 flasker, med en profitt på 410.0625
```

```
Mack Mikrobryggeri produserer optimalt 10.875 flasker, med en profitt på 440.4375
Markedsprisen er lik 50.5
```

Oppgave 2.A - Optimal tilpasning etter fusjon

```
# Definerer profittfunksjon etter fusjon.
def profit_02(q1, q2, c, a, b):
    return(demand_02(q1+q2, a, b)-c)*q1
```

```
# Deriverer profittfunksjonen med hensyn på q1.
der_profit4_Q = diff(profit_02(q1, q2, c1, a, b), q1)

# Deriverer profittfunksjonen med hensyn på q2.
der_profit5_Q = diff(profit_02(q2, q1, c2, a, b), q2)

# Viser resultatet.
display(der_profit4_Q)
display(der_profit5_Q)
```

$$a - bq_1 - b\left(q_1 + q_2\right) - c_1$$

$$a - bq_2 - b\left(q_1 + q_2\right) - c_2$$

```
# Setter lik null og løser med hensyn på q1/q2.
sol_02 = solve([der_profit4_Q, der_profit5_Q], [q1, q2])

# Viser resultatet.
display(sol_02[q1])
display(sol_02[q2])
```

$$\frac{a - 2c_1 + c_2}{3b}$$

$$\frac{a + c_1 - 2c_2}{3b}$$

```
# Lager en lambdify funksjon.
opt_qf = lambdify((a, b, c1, c2), (sol_02[q1], sol_02[q2]))

# Definerer verdier.
a_val = 175
b_val = 4
```

```
c1_val = 10
c2_val = 7

# Bruker lamdify funksjonen til å løse den optimale tilpasnigen.
q4sol, q5sol = opt_qf(a_val, b_val, c1_val, c2_val)

# Finner profitten til begge bryggeriene.
prof4 = profit_02(q4sol, q5sol, c1_val, a_val, b_val)
prof5 = profit_02(q4sol, q5sol, c2_val, a_val, b_val)

# Skriver ut svarene.
print(f"Graff Brygghus produserer optimalt", q4sol, f"flasker, med en profitt på", prof4)
print(f"En fusjon av Bryggeri 13 og Mack Mikrobrygg produserer optimalt", q5sol, f"flasker
print(f"Markedsprisen blir", demand_02(q4sol+q5sol, a_val, b_val))
```

```
Graff Brygghus produserer optimalt 13.5 flasker, med en profitt på 729.0
En fusjon av Bryggeri 13 og Mack Mikrobrygg produserer optimalt 14.25 flasker, med en profit
Markedsprisen blir 64.0
```

Oppgave 2.B - Optimal tilpasning før fusjon

```
# Definerer den totale etterspørselen.
def demand_03(q1, q2):
    return 175-2*(q1+q2)
```

```
# Definerer kostnadsfunksjonene for begge bryggeriene.
def c_brygg(q1):
    return 7 * q1

def c_mack(q2):
    return 7 * q2
```

```
# Definerer profittfunksjoner.
def prof_brygg(q1, q2):
    return demand_03(q1, q2)*q1-c_brygg(q1)

def prof_mack(q1, q2):
    return demand_03(q1, q2)*q2-c_mack(q2)
```

```python
# Definerer marginalinntekt.
MR1 = sp.diff(demand_03(q1, q2)*q1, q1)

# Setter lik kostnad per flaske og løser for q1.
sol_RF1 = sp.solve(sp.Eq(MR1, 7), q1)[0]

# Definerer reaksjonsfunksjon.
def RF1(q2):
    reaction = eval(str(sol_RF1))
    return reaction
```

```python
# Definerer mariginalinntekt.
MR2 = sp.diff(demand_03(q1, q2)*q2, q2)

# Setter lik kostnad per flaske og løser for q2.
sol_RF2 = sp.solve(sp.Eq(MR2, 7), q2)[0]

# Definerer reaksjonsfunksjon.
def RF2(q1):
    reaction = eval(str(sol_RF2))
    return reaction
```

```python
# Setter reaksjonsfunksjon RF1 inn i reaksjonsfunksjon RF2 for å finne optimalt kvantum q2
q2_R = sp.solve(sp.Eq(RF2(RF1(q2)),q2),q2)[0]

# Finn optimalt kvantum q1.
q1_R = RF1(q2_R)

# Skriver ut svarene.
print(f"Optimal kvantum for Graff Brygghus er lik", q1_R)
print(f"Optimal kvantum for Mack Mikrobryggeri 13 er lik",q2_R)
print(f"Optimal markedspris er lik", demand_03(q1_R, q2_R))
```

```
Optimal kvantum for Graff Brygghus er lik 28
Optimal kvantum for Mack Mikrobryggeri 13 er lik 28
Optimal markedspris er lik 63
```

```python
# Skriver ut profittene.
print(f"Profitten til Graff Brygghus er lik", prof_brygg(q1_R, q2_R))
print(f"Profitten til Mack Mikrobryggeri 13 er lik", prof_mack(q1_R, q2_R))
```

```
Profitten til Graff Brygghus er lik 1568
Profitten til Mack Mikrobryggeri 13 er lik 1568
```

Oppgave 2.B - Optimal tilpasning etter fusjon

```python
# Definerer etterspørsel.
def demand_f(q):
    return demand_03(q, 0)

# Definerer profittfunksjon.
def prof_f(q):
    return (demand_f(q)-7)*q

# Finner marginalinntekt.
MR_M = sp.diff(demand_f(Q)*Q, Q)

# Finner optimal kvantum.
Q_M = sp.solve(sp.Eq(MR_M, 7), Q)[0]

# Finner optimal pris.
opt_P = demand_f(Q_M)

# Skriver ut svarene.
print(f"Optimalt kvantum for monopolisten er lik", Q_M)
print(f"Optimalt pris for monopolisten er lik", opt_P)
print(f"Profitten til monopolisten er lik", prof_f(Q_M))
```

```
Optimalt kvantum for monopolisten er lik 42
Optimalt pris for monopolisten er lik 91
Profitten til monopolisten er lik 3528
```