Preben Jensen ejer en mindre restaurant i Sønder Bråbjerg, som er ret godt besøgt. Preben har gennem en årrække oparbejdet et vist ry for at skabe store kulinariske oplevelser ud af årstidens lokale råvarer, og dette ry har gjort, at han som oftest har fulde huse både torsdag, fredag, lørdag og søndag.

Men, så ramte coronakrisen. Preben måtte holde lukket i en periode, hvor en stor del af omsætningen forsvandt. Da der igen blev åbnet op i løbet af sommeren 2020, trak Preben Jensen derfor sig selv op ved nakkehårene og bedyrede "Nu skal vi lave restaurant igen. Vi skal lave mad, vi skal have glade gæster og vi skal gøre det forsvarligt!".

Prebens restaurant i Sønder Bråbjerg er indrettet efter traditionerne fra amerikanske dinere, hvor bordene er boltet fast til gulvet, og det er derfor desværre ikke muligt, at flytte rundt på dem. For at overholde myndighedernes afstandskrav i restauranten, har Preben derfor bestemt sig for at spærre nogle af bordene af så de ikke kan bruges. Nu er Prebens problem bare, at han skal vælge hvilke borde, der skal forblive i brug og hvilke der skal spærres af, således at Preben får plads til flest muligt i restauranten, uden at folk sidder for tæt. Preben har været rundt med en tommestok og har målt afstanden mellem bordene på det sted hvor de står tættest og har på den baggrund lavet en oversigt over hvilke borde, der ikke kan være i brug på samme tid uden at bryde det såkaldte afstandskrav. Prebens indsamlede data er tilgængeligt i det vedlagte Excel-ark.

Din opgave er nu at hjælpe Preben med at etablere en plan for den første åbningsaften ved at besvare de følgende opgaver. I de følgende tre opgaver kan du benytte binære variabler y_i for hvert bord defineret ved

$$y_i = \begin{cases} 1, hvis\ bord\ nummer\ i\ benyttes \\ 0, hvis\ bord\ nummer\ i\ spærres \end{cases}$$

Opgave 1: (10 point)

Formuler en lineær objektfunktion, som maksimerer antallet af pladser i restauranten.

Svar: Lad $B = \{bord\ 1, Bord\ 2, \dots, bord\ 20\}$ være mængden af alle borde og lad b_i være antallet af pladser ved bord $i \in B$. Da kan objektfunktionen, som maksimerer det totale antal pladser i restauranten, formuleres som følger:

$$\max \sum_{i \in B} b_i y_i$$

Opgave 2: (10 point)

Formuler en mængde af lineære begrænsninger, der sørger for, at der ikke benyttes borde som er for tætte på hinanden (*Hint: Hvis to borde er i konflikt, kan man ikke benytte begge to*). Formuler et lineært heltalsprogram ved at kombinere begrænsningerne med objektfunktion fra opgave 1.

Svar: Lad her $(a_{ij})_{i,j \in B}$ være en matrix hvor $a_{ij} = 1$ hvis bord i og bord j er i konflikt med hinanden (de kan ikke begge være i brug) og ellers er $a_{ij} = 0$. Da skal der gælde, at vi maksimalt kan vælge 1 af de to borde i og j hvis $a_{ij} = 1$. Det vil sige, at vi skal have

$$y_i + y_j \leq 1, \quad \forall i, j \in B: a_{ij} = 1$$

Alternativt kan disse begrænsninger også formuleres på én af følgende to måder

$$a_{ij}(y_i + y_j) \leq 1, \quad \forall i,j \in B$$

$$y_i + y_j + a_{ij} \leq 2, \quad \forall i,j \in B$$

Alle tre måder er ækvivalente og skrives de ud, er det nøjagtigt de samme uligheder som udtrykkes ved ovenstående tre formuleringer.

Objektfunktionen og begrænsningerne leder sammen til følgende lineære heltalsmodel:

$$\max \sum_{i \in B} b_i y_i$$

$$\text{s.t.}: y_i + y_j \leq 1, \quad \forall i,j \in B: a_{ij} = 1$$

$$\left(a_{ij}(y_i + y_j) \leq 1, \quad \forall i,j \in B\right)$$

$$(y_i + y_j + a_{ij} \leq 2, \quad \forall i,j \in B)$$

$$y_i \in \{0,1\}, \quad \forall i \in B$$

Opgave 3: (20 Point)

Implementer din model fra opgave 1 og 2 i OPL og løs den ved hjælp af CPLEX. Kommenter på løsningen: hvilke borde er valgt og hvor mange pladser er bevaret i restauranten.

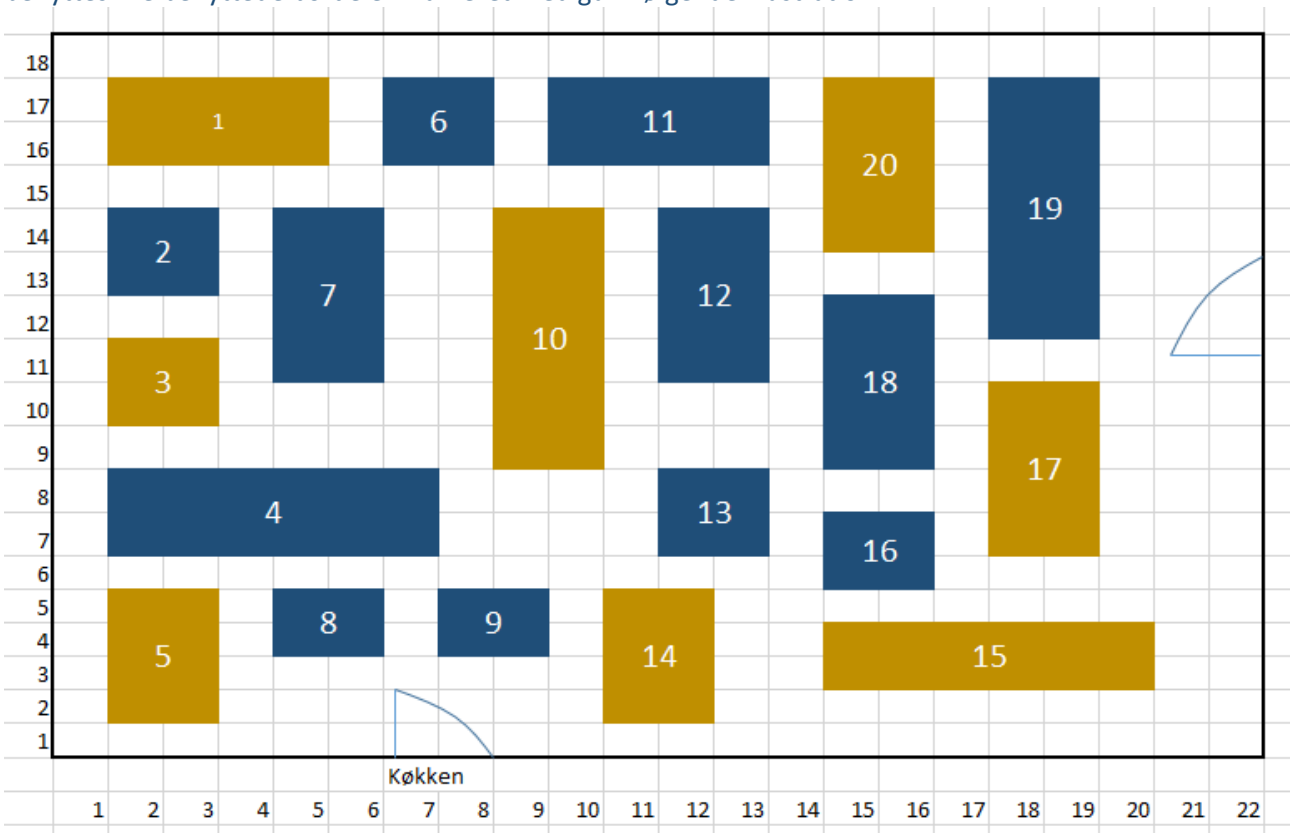
Svar: En implementering i OPL af modellen fra opgave 2 kan se ud som følger

```
{string} borde = ...;
int antalPladser[borde] = ...;
int a[borde][borde] = ...;

dvar boolean y[borde];

maximize sum ( i in borde ) antalPladser[i] * y[i];
subject to
{
  forall ( i, j in borde : a[i][j] ==1 )
  {
    y[i] + y[j] <= 1;
  }
  //Alternativt
  //forall ( i,j in borde )
  //{
  //  a[i][j] * ( y[i] + y[j] ) <= 1;
  //  y[i] + y[j] + a[i][j] <= 2;
  //}
}
```

Med følgende data-fil

```
borde = {"Bord 1""Bord 2""Bord 3""Bord 4""Bord 5""Bord 6""Bord 7""Bord 8""Bord 9""Bor
antalPladser = [4 2 2 6 4 2 4 2 2 6 4 4 2 4 6 2 4 4 6 4];
a =[[0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0],
    [1 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0],
    [0 1 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0],
    [0 0 1 0 1 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0],
    [0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0],
    [1 0 0 0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 0 0],
    [1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0],
    [0 0 0 1 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0],
    [0 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0],
    [0 0 0 1 0 1 0 0 0 0 1 1 1 0 0 0 0 0 0 0],
    [0 0 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0 0 0 1],
    [0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 1],
    [0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 0 1 0 0],
    [0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0],
    [0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0],
    [0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 1 0 0],
    [0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 1 0 0],
    [0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1],
    [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1],
    [0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 0]];
```

En optimal løsning leder til 34 pladser i restauranten og bordene Bord 1, Bord 3, Bord 5, Bord 10, Bord 14, Bord 15, Bord 17, Bord 20 benyttes mens resten tages ud af brug. Bord 3 er det eneste 2-mandsbord som benyttes. De benyttede borde er markeret med gul i følgende illustration



Ved gennemsyn af løsningen fra Opgave 3, bliver Preben i tvivl om, om det i virkeligheden er den løsning, der løser hans problem. Der er kun valgt ét tomandsbord i løsningen, og Prebens erfaring siger ham, at der ofte er en del par, som ønsker at komme ud og have en romantisk aften i Sønder Bråbjerg. Derfor vil Preben gerne have en analyse af de trade-offs der måtte være mellem antallet af pladser i restauranten og antallet af tomandsborde, som er i brug.

Opgave 4: (20 point)

Beskriv ε-constraint metoden og brug denne til at beregne alle rationelle kompromisser mellem antallet af pladser i restauranten og antallet af tomandsborde. Beskriv din fremgangsmetode og illustrer og kommenter på løsningerne.

Svar: ε-constraint metoden benyttes til at løse multikriterie optimeringsproblemer når man ønsker at finde alle rationelle kompromisser mellem objektfunktionerne som er spil. Metoden fungerer ved, at man initialt vælger én objektfunktion som alt fokus skal være på. Problemet optimeres nu uden at skele til de andre objektfunktioner. Når en løsning er fundet, kræves en forbedring i en af de andre objektfunktioner som før blev ignoreret (en begrænsning tilføjes som fordrer forbedring af den ignorerede objektfunktion) og problemet genoptimeres. Metoden fortsætter indtil problemet er *infeasible* hvorved det vides, at de objektfunktioner som er placeret blandt begrænsningerne ikke kan forbedres mere.

I dette tilfælde, vil ε-constraint metoden forløbe som følger:

1. Først løses problemet

$$\max \sum_{i \in B} b_i y_i$$
$$\text{s.t.}: y_i + y_j \leq 1, \quad \forall i,j \in B: a_{ij} = 1$$
$$y_i \in \{0,1\}, \quad \forall i \in B$$

2. Det registreres, at antallet af tomandsborde som benyttes i løsningen er lig med 1, hvorfor en begrænsning som kræver at mindst to tomandsborde benyttes bliver tilføjet modellen:

$$\max \sum_{i \in B} b_i y_i$$
$$\text{s.t.}: y_i + y_j \leq 1, \quad \forall i,j \in B: a_{ij} = 1$$
$$y_2 + y_3 + y_6 + y_8 + y_9 + y_{13} + y_{16} \geq 2$$
$$y_i \in \{0,1\}, \quad \forall i \in B$$

3. Denne model løses nu, og det registreres, at der nu er 2 tomandsborde og 32 pladser i restauranten. Derfor hæves den nedre grænse på antallet af tomandsborde nu til 3:

$$\max \sum_{i \in B} b_i y_i$$
$$\text{s.t.}: y_i + y_j \leq 1, \quad \forall i,j \in B: a_{ij} = 1$$
$$y_2 + y_3 + y_6 + y_8 + y_9 + y_{13} + y_{16} \geq 3$$
$$y_i \in \{0,1\}, \quad \forall i \in B$$

4. Løses denne model, fås en løsning med 3 tomandsborde og 30 siddepladser i restauranten. Derfor hæves den nedre grænse på antallet af tomandsborde nu til 4:

$$\max \sum_{i \in B} b_i y_i$$
$$\text{s.t.}: y_i + y_j \leq 1, \quad \forall i,j \in B: a_{ij} = 1$$

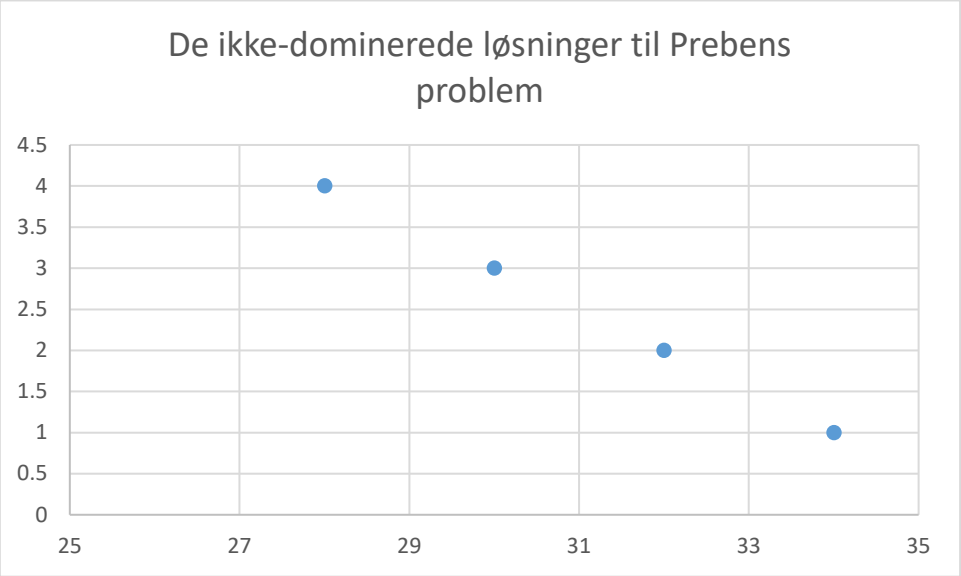
$$y_2 + y_3 + y_6 + y_8 + y_9 + y_{13} + y_{16} \geq 4$$
$$y_i \in \{0,1\}, \quad \forall i \in B$$

5. Løses denne model, fås en løsning med 4 tomandsborde og 28 siddepladser i restauranten. Derfor hæves den nedre grænse på antallet af tomandsborde nu til 5:

$$\max \sum_{i \in B} b_i y_i$$
$$\text{s.t.}: y_i + y_j \leq 1, \quad \forall i,j \in B: a_{ij} = 1$$
$$y_2 + y_3 + y_6 + y_8 + y_9 + y_{13} + y_{16} \geq 5$$
$$y_i \in \{0,1\}, \quad \forall i \in B$$

6. Løses denne model, fås at modellen er "*infeasible*" hvorved vi ved, at det ikke kan lade sig gøre at forbedre objektfunktionen som maksimerer antallet af tomandsborde mere. Derfor stopper ε-constraint metoden her med de 4 fundne løsninger

De ikke-dominerede løsningsvektorer ser således ud



Det ses, at metoden stopper ved 4 tomandsborde selvom der faktisk er 7 tomandsborde i restauranten. Men det er ikke muligt at benytte mere end 4 tomandsborde grundet afstandskravet.

I det følgende antages det, at Preben vælger en løsning hvor bordene 1, 3, 5, 10, 14, 15,17 og 20 er i brug.

Hver ret, som serveres i Prebens restaurant, bliver serveret samtidig for alle gæster. Restaurantens 3 tjenere marcherer ud fra køkkenet med en bakke hver, som kan holde 12 serveringer, og så servicerer de bordene i restauranten. Det er vigtigt, at hvert bord kun serviceres af én tjener, da det giver den bedste oplevelse for gæsterne. Men, det er naturligvis vigtigt for Preben, at hans tjenere ikke bliver udsat for unødvendig smitterisiko, hvorfor tjenernes "ruter" rundt blandt bordene skal optimeres. I det vedlagte Excel-ark (Smittescore-matrix) er angivet en matrix med en smitte-score som angiver hvor høj smitterisikoen er ved at bevæge sig mellem alle par af borde. Matricens indgange inkluderer risikoen ved at servicere bordene.

Opgave 5: (20 point)

Det er nu din opgave, at formulere en matematisk optimeringsmodel, der beregner de bedst-mulige ruter for restaurantens tjenere således, at hvert bord bliver serviceret af præcis én tjener, hver tjener maksimalt bærer 12 anretninger på sin bakke og således, at alle tjenere starter og slutter deres rute i køkkenet. Med bedst muligt menes ruter, der samlet set har lavest mulig smitte-score. Beskriv din models delelementer (data, variabler, objektfunktion og begrænsninger).

Svar: Jeg vil her benytte mig af modeller for Capacitated Vehicle Routing Problem (CVRP) og bruger MTZ formuleringen. En Gavish og Graves one-commodity flow formulering kunne også sagtens bruges her, men problemets størrelse taget i betragtning, er det ikke strengt nødvendigt. Som input-data og variabler til modellen benyttes de i tabellen viste parametre og variabler:

Parametre	**Beskrivelse**
V	Mængde af borde som skal besøges: $V = \{Bord\ 1, Bord\ 3, Bord\ 5, Bord\ 10, Bord\ 14, Bord\ 15, Bord\ 17, Bord\ 20\}$
V_0	Mængden af alle "punkter" som skal besøges i restauranten. Det vil sige V inklusiv køkkenet. Vi vil referere til køkkenet som "punkt 0": $V_0 = \{0\} \cup V$
b_i	Antal serveringer ved hvert bord. Det vil sige, antallet af pladser ved hvert bord.
Q	Antallet af serveringer en tjener kan have på en bakke. Her er det 12, serveringer.
s_{ij}	Smitte-score som indtræffer hvis en tjener bevæger sig direkte fra punkt $i \in V_0$ til $j \in V_0$
m	Antallet af tjenere der er til rådighed. I dette tilfælde er $m = 3$
Variabler	**Beskrivelse**
x_{ij}	Binær variabel for hvert par af punkter, $i, j \in V_0$, som skal besøges i restauranten. Hvis $x_{ij} = 1$ betyder det, at en tjener går direkte fra punkt i til punkt j i restauranten.
τ_i	Kontinuert variabel for hvert bord $i \in V$, som angiver hvor mange serveringer en tjener har leveret når denne *forlader* bord $i \in V$.

Objektfunktionen er nu at minimere den samlede smitte eksponering er givet ved

$$\min \sum_{i \in V_0} \sum_{\substack{j \in V_0 \\ i \neq j}} s_{ij} x_{ij}$$

Følgende begrænsninger skal overholdes:

1. Der skal gå præcis tre tjenere ud af og ind i køkkenet:

$$\sum_{j \in V} x_{0j} = m$$

$$\sum_{i \in V} x_{i0} = m$$

2. Der skal ankomme præcis én tjener til hvert bord og der skal være præcis én tjener som forlader hvert bord:

$$\sum_{\substack{i \in V_0 \\ i \neq j}} x_{ij} = 1, \quad \forall j \in V$$

$$\sum_{\substack{j \in V_0 \\ i \neq j}} x_{ij} = 1, \quad \forall i \in V$$

3. En tjener kan ikke servicere mere en Q serveringer grundet bakkens beskaffenhed og der kan ikke være leveret færre end antallet af pladser ved bordet når tjeneren forlader borde:
$$b_i \leq \tau_i \leq Q, \quad \forall i \in V$$
4. Antallet af leverede serveringer skal akkumuleres korrekt for tjenerne:
$$\tau_i - \tau_j + Qx_{ij} \leq Q - b_j, \quad \forall i,j \in V$$

Dette leder samlet frem til følgende MILP:

$$\min \sum_{i \in V_0} \sum_{\substack{j \in V_0 \\ i \neq j}} s_{ij} x_{ij}$$

$$\begin{aligned}
\text{s.t.}: & \sum_{j \in V} x_{0j} = m \\
& \sum_{i \in V} x_{i0} = m \\
& \sum_{\substack{i \in V_0 \\ i \neq j}} x_{ij} = 1, \quad \forall j \in V \\
& \sum_{\substack{j \in V_0 \\ i \neq j}} x_{ij} = 1, \quad \forall i \in V \\
& b_i \leq \tau_i \leq Q, \quad \forall i \in V \\
& \tau_i - \tau_j + Qx_{ij} \leq Q - b_j, \quad \forall i,j \in V \\
& x_{ij} \in \{0,1\}, \quad \forall i,j \in V_0
\end{aligned}$$

Opgave 6 (20 point)

Implementer modellen fra opgave 5 ved hjælp af OPL og løs den ved hjælp af CPLEX. Illustrer og kommenter på din løsning. Argumenter for om løsningen er *fair* eller ej.

Svar: En implementering af modellen fra Opgave 5 kan se ud som følger:

```
{string} V_0 = ...;
int s[V_0][V_0] = ...;
int Q = ...;
int b[V_0] = ...;
int m = ...;

{string} V = { i | i in V_0 : i!= "køkken"};
dvar boolean x[V_0][V_0];
dvar float+ tau[i in V] in b[i]..Q;

minimize sum (i,j in V_0 ) s[i][j] * x[i][j];
subject to
{
  sum ( j in V ) x["køkken"][j] == m;
  sum ( i in V ) x[i]["køkken"] == m;

  forall ( i in V )
  {
    sum ( j in V_0 ) x[i][j] == 1;
    sum ( j in V_0 ) x[j][i] == 1;
  }
  forall ( i,j in V )
  {
    tau[i] - tau[j] + Q*x[i][j] <= Q - b[j];
  }
}
```

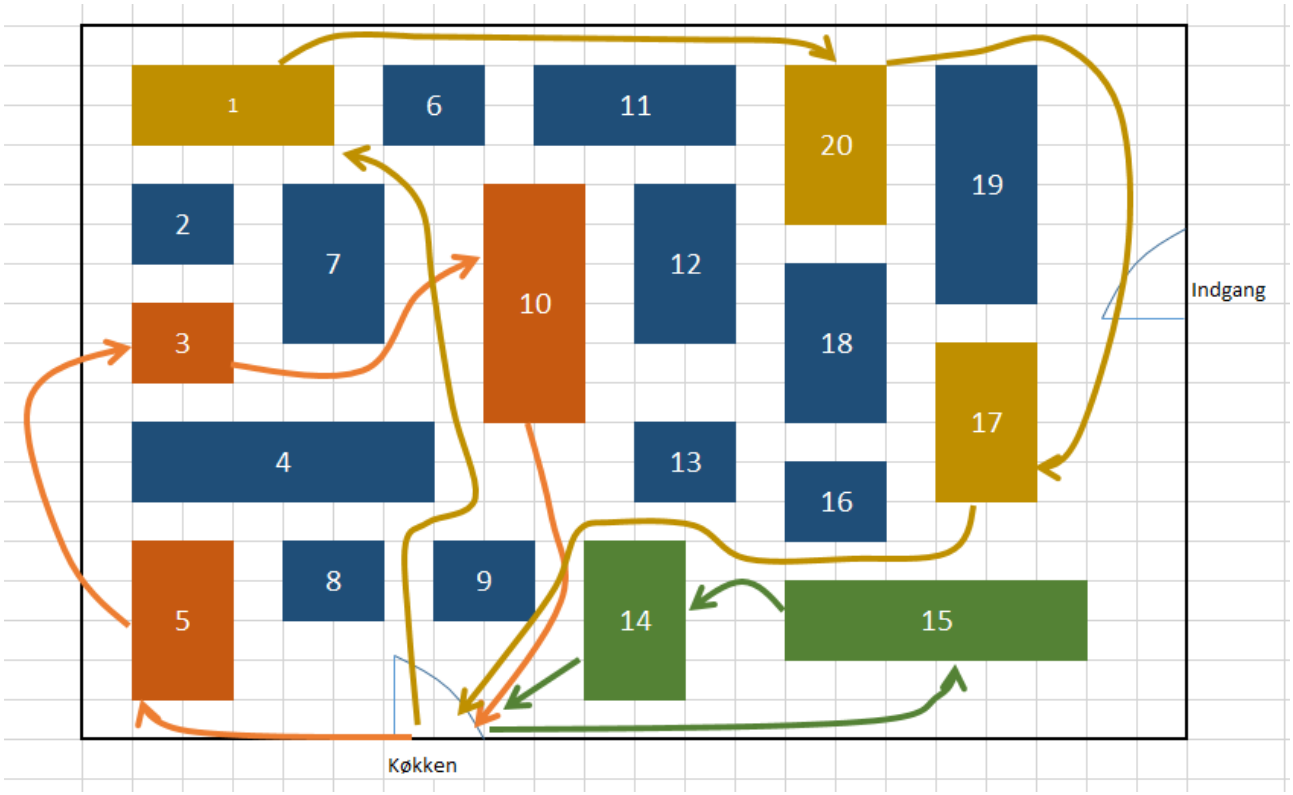
Hvor data-filen er givet ved:

```
V_0 = {"køkken" "Bord 1" "Bord 3" "Bord 5" "Bord 10" "Bord 14" "Bord 15" "Bord 17" "Bord 20"};
s=[ [0  12  5   6   13  15  21  19  23],
    [16 0   11  18  15  25  31  27  17],
    [7  9   0   9   10  18  24  20  20],
    [10 18  11  0   19  13  19  25  29],
    [19 17  14  21  0   16  22  18  16],
    [19 25  20  13  14  0   10  16  20],
    [27 33  28  21  22  12  0   12  20],
    [23 27  22  25  16  16  10  0   14],
    [27 17  22  29  14  20  18  14  0]];
Q = 12;
b = [0 4 2 4 6 4 6 4 4 ];
m = 3;
```

En optimal løsning leder til en samlet smitte-score på 164 med følgende ruter for de tre tjenere

Tjener	Rute	Smitte-score
1	Køkken – 1 – 20 – 17 – Køkken	66
2	Køkken – 5 – 10 – 3 – Køkken	46
3	Køkken – 15 – 14 – Køkken	52

Nedenfor er løsningen illustreret med ruter gennem restauranten.



Umiddelbart virker løsningen ikke fair idet der er relativt stor forskel på hvor meget den enkelte tjener er udsat for smitte. Fx har tjener 1 en smitte-score som er $\frac{66-46}{46} \approx 43\%$ højere end tjener 2's smitte-score. Man kunne derfor, måske med rette, argumentere for, at det ikke er den rette model, men at man i stedet for at minimere den samlede-smitte score burde minimere smitte-scoren for den tjener som har den højeste smitte-score. Det vil sige, at objektfunktionen i stedet burde have været af "min-max" typen.

OBS: Der er alternative optimale løsninger til modellen. Derfor, sørg for at sammenlign med objektfunktionsværdien. Følgende er en liste over alle optimale løsninger

```
Tjener nummer 1:
køkken -> Bord 1-> Bord 20-> Bord 17-> køkken      SmitteScore : 66
Tjener nummer 2:
køkken -> Bord 5-> Bord 10-> Bord 3-> køkken       SmitteScore : 46
Tjener nummer 3:
køkken -> Bord 15-> Bord 14-> køkken                SmitteScore : 52
```

```
Tjener nummer 1:
køkken -> Bord 1-> Bord 20-> Bord 17-> køkken      SmitteScore : 66
Tjener nummer 2:
køkken -> Bord 10-> Bord 3-> Bord 5-> køkken       SmitteScore : 46
Tjener nummer 3:
køkken -> Bord 15-> Bord 14-> køkken               SmitteScore : 52
```

```
Tjener nummer 1:
køkken -> Bord 1-> Bord 20-> Bord 17-> køkken      SmitteScore : 66
Tjener nummer 2:
```

køkken -> Bord 5-> Bord 3-> Bord 10-> køkken Smitte Score : 46 Tjener nummer 3: køkken -> Bord 14-> Bord 15-> køkken Smitte Score : 52
Tjener nummer 1: køkken -> Bord 1-> Bord 20-> Bord 17-> køkken Smitte Score : 66 Tjener nummer 2: køkken -> Bord 10-> Bord 3-> Bord 5-> køkken Smitte Score : 46 Tjener nummer 3: køkken -> Bord 14-> Bord 15-> køkken Smitte Score : 52
Tjener nummer 1: køkken -> Bord 1-> Bord 20-> Bord 17-> køkken Smitte Score : 66 Tjener nummer 2: køkken -> Bord 10-> Bord 3-> Bord 5-> køkken Smitte Score : 46 Tjener nummer 3: køkken -> Bord 14-> Bord 15-> køkken Smitte Score : 52
Tjener nummer 1: køkken -> Bord 5-> Bord 3-> Bord 10-> køkken Smitte Score : 46 Tjener nummer 2: køkken -> Bord 15-> Bord 14-> køkken Smitte Score : 52 Tjener nummer 3: køkken -> Bord 17-> Bord 20-> Bord 1-> køkken Smitte Score : 66
Tjener nummer 1: køkken -> Bord 1-> Bord 20-> Bord 17-> køkken Smitte Score : 66 Tjener nummer 2: køkken -> Bord 3-> Bord 10-> Bord 5-> køkken Smitte Score : 46 Tjener nummer 3: køkken -> Bord 14-> Bord 15-> køkken Smitte Score : 52
Tjener nummer 1: køkken -> Bord 5-> Bord 3-> Bord 10-> køkken Smitte Score : 46 Tjener nummer 2: køkken -> Bord 14-> Bord 15-> køkken Smitte Score : 52 Tjener nummer 3: køkken -> Bord 17-> Bord 20-> Bord 1-> køkken Smitte Score : 66
Tjener nummer 1: køkken -> Bord 1-> Bord 20-> Bord 17-> køkken Smitte Score : 66 Tjener nummer 2: køkken -> Bord 3-> Bord 10-> Bord 5-> køkken Smitte Score : 46 Tjener nummer 3: køkken -> Bord 15-> Bord 14-> køkken Smitte Score : 52
Tjener nummer 1: køkken -> Bord 10-> Bord 3-> Bord 5-> køkken Smitte Score : 46 Tjener nummer 2: køkken -> Bord 15-> Bord 14-> køkken Smitte Score : 52 Tjener nummer 3: køkken -> Bord 17-> Bord 20-> Bord 1-> køkken Smitte Score : 66
Tjener nummer 1: køkken -> Bord 1-> Bord 20-> Bord 17-> køkken Smitte Score : 66

```
Tjener nummer 2:
køkken -> Bord 5-> Bord 3-> Bord 10-> køkken      Smitte Score : 46
Tjener nummer 3:
køkken -> Bord 15-> Bord 14-> køkken              Smitte Score : 52
```

```
Tjener nummer 1:
køkken -> Bord 5-> Bord 10-> Bord 3-> køkken      Smitte Score : 46
Tjener nummer 2:
køkken -> Bord 15-> Bord 14-> køkken              Smitte Score : 52
Tjener nummer 3:
køkken -> Bord 17-> Bord 20-> Bord 1-> køkken     Smitte Score : 66
```

```
Tjener nummer 1:
køkken -> Bord 3-> Bord 10-> Bord 5-> køkken      Smitte Score : 46
Tjener nummer 2:
køkken -> Bord 15-> Bord 14-> køkken              Smitte Score : 52
Tjener nummer 3:
køkken -> Bord 17-> Bord 20-> Bord 1-> køkken     Smitte Score : 66
```

```
Tjener nummer 1:
køkken -> Bord 1-> Bord 20-> Bord 17-> køkken     Smitte Score : 66
Tjener nummer 2:
køkken -> Bord 5-> Bord 10-> Bord 3-> køkken      Smitte Score : 46
Tjener nummer 3:
køkken -> Bord 14-> Bord 15-> køkken              Smitte Score : 52
```

```
Tjener nummer 1:
køkken -> Bord 5-> Bord 10-> Bord 3-> køkken      Smitte Score : 46
Tjener nummer 2:
køkken -> Bord 14-> Bord 15-> køkken              Smitte Score : 52
Tjener nummer 3:
køkken -> Bord 17-> Bord 20-> Bord 1-> køkken     Smitte Score : 66
```

```
Tjener nummer 1:
køkken -> Bord 3-> Bord 10-> Bord 5-> køkken      Smitte Score : 46
Tjener nummer 2:
køkken -> Bord 14-> Bord 15-> køkken              Smitte Score : 52
Tjener nummer 3:
køkken -> Bord 17-> Bord 20-> Bord 1-> køkken     Smitte Score : 66
```

```
Tjener nummer 1:
køkken -> Bord 10-> Bord 3-> Bord 5-> køkken      Smitte Score : 46
Tjener nummer 2:
køkken -> Bord 14-> Bord 15-> køkken              Smitte Score : 52
Tjener nummer 3:
køkken -> Bord 17-> Bord 20-> Bord 1-> køkken     Smitte Score : 66
```