HW 2 Problem 2b

The final LP problem is

Maximise

$$300x_{1Y} + 220x_{1B} + 100x_{1M} + 160x_{2Y} + 130x_{1B} + 80x_{2M} + 360x_{3Y} + 280x_{3B} + 140x_{3M}$$

subject to

$x_{1Y} \leq 4$

$x_{1B} \leq 8$

$x_{1M} \leq 22$

$x_{2Y} \leq 8$

$x_{2B} \leq 13$

$x_{2M} \leq 20$

$x_{3Y} \leq 3$

$x_{3B} \leq 10$

$x_{3M} \leq 18$

$x_{1Y} + x_{1B} + x_{1M} + x_{3Y} + x_{3B} + x_{3M} \leq 30$

$x_{2Y} + x_{2B} + x_{2M} + x_{3Y} + x_{3B} + x_{3M} \leq 30$

$x_{1Y}, x_{1B}, x_{1M}, x_{2Y}, x_{1B}, x_{2M}, x_{3Y}, x_{3B}, x_{3M} \geq 0$

and $x_{1Y}, x_{1B}, x_{1M}, x_{2Y}, x_{1B}, x_{2M}, x_{3Y}, x_{3B}, x_{3M}$ are integers.

Steps for installing PuLP

In [1]:
```
import sys
!{sys.executable} -m pip install pulp
```

```
Requirement already satisfied: pulp in /opt/conda/lib/python3.8/site
-packages (2.3)
Requirement already satisfied: amply>=0.1.2 in /opt/conda/lib/python
3.8/site-packages (from pulp) (0.1.2)
Requirement already satisfied: pyparsing in /opt/conda/lib/python3.8
/site-packages (from amply>=0.1.2->pulp) (2.4.7)
Requirement already satisfied: docutils>=0.3 in /opt/conda/lib/pytho
n3.8/site-packages (from amply>=0.1.2->pulp) (0.15.2)
```

In [2]:
```
import pulp
```

In [3]:
```
# Import PuLP modeler functions
from pulp import *     # Here because of * we will not put `pulp' befor
e each pulp command; e.g. instead of pulp.LpVariable, we simply write
LpVariable.
```

Steps for Decision Variables.

In [4]:
```
# Creates a list of the Ingredients
ticket = ['1Y', '1B', '1M', '2Y', '2B', '2M', '3Y', '3B', '3M']   # th
e list of type of tickes.

R13 = ['1Y', '1B', '1M', '3Y', '3B', '3M'] # the tickets for route 1 a
nd route 3  (tickes for passengers in the flights to Ithaca to Newark)

R23 = ['2Y', '2B', '2M', '3Y', '3B', '3M'] # the tickets for route 2 a
nd route 3  (tickes for passengers in the flights to Newark to Boston)

# A dictionary of the costs of each of ticket
price = {'1Y': 300,
         '1B': 220,
         '1M': 100,
         '2Y': 160,
         '2B': 130,
         '2M': 80,
         '3Y': 360,
         '3B': 280,
         '3M':140}
```

In [5]:
```
# Create the 'prob' variable to contain the problem data
prob = LpProblem("Ticketing_Problem", LpMaximize)
```

In [6]:
```
# A dictionary called 'ticket_vars' is created to contain the referenc
ed Variables
ticket_vars = LpVariable.dicts("ticket", ticket, lowBound=0, cat='Inte
ger')    # Here 'lowBound=0' gives the lower bound for the variable.
# Here, cat='Integer' restricts the variable to be integer variable, t
hat is, they take only interger values.
```

Objective function

In [7]:
```
# The objective function is added to 'prob' first
prob += lpSum([price[i]*ticket_vars[i] for i in ticket]), "Total Reven
ue"
```

Constraints

In [8]:
```
# The constraints are added to 'prob'

prob += ticket_vars['1Y'] <= 4
prob += ticket_vars['1B'] <= 8
prob += ticket_vars['1M'] <= 22
prob += ticket_vars['2Y'] <= 8
prob += ticket_vars['2B'] <= 13
prob += ticket_vars['2M'] <= 20
prob += ticket_vars['3Y'] <= 3
prob += ticket_vars['3B'] <= 10
prob += ticket_vars['3M'] <= 18
prob += lpSum([ticket_vars[i]  for i in R13]) <= 30
prob += lpSum([ticket_vars[i]  for i in R23]) <= 30

# Notice that when we defined the decision variable ticket_vars = LpVa
riable.dicts("ticket", ticket, lowbound=0, cat='Integer')
# we already gave the lower bound 0 for the variables, and also we res
tricted those variables to be integer variables.
```

Show the LP problem.

In [9]:
```python
# Or you can directly display the problem here.

print(prob)
```

```
Ticketing_Problem:
MAXIMIZE
220*ticket_1B + 100*ticket_1M + 300*ticket_1Y + 130*ticket_2B + 80*t
icket_2M + 160*ticket_2Y + 280*ticket_3B + 140*ticket_3M + 360*ticke
t_3Y + 0
SUBJECT TO
_C1: ticket_1Y <= 4

_C2: ticket_1B <= 8

_C3: ticket_1M <= 22

_C4: ticket_2Y <= 8

_C5: ticket_2B <= 13

_C6: ticket_2M <= 20

_C7: ticket_3Y <= 3

_C8: ticket_3B <= 10

_C9: ticket_3M <= 18

_C10: ticket_1B + ticket_1M + ticket_1Y + ticket_3B + ticket_3M + ti
cket_3Y
 <= 30

_C11: ticket_2B + ticket_2M + ticket_2Y + ticket_3B + ticket_3M + ti
cket_3Y
 <= 30

VARIABLES
0 <= ticket_1B Integer
0 <= ticket_1M Integer
0 <= ticket_1Y Integer
0 <= ticket_2B Integer
0 <= ticket_2M Integer
0 <= ticket_2Y Integer
0 <= ticket_3B Integer
0 <= ticket_3M Integer
0 <= ticket_3Y Integer
```

Notice that the lower bound >=0 for the variable is not shown, as it is the default condition. If you had changed the lowerbound to something else, then it will show up here.

Solve the LP.

In [10]:
```
# The problem is solved using PuLP's choice of Solver
prob.solve()
# The status of the solution is printed to the screen
print("Status:", LpStatus[prob.status])
```

```
Status: Optimal
```

Show the values for the optimal solution

In [11]:
```
# Each of the variables is printed with it's resolved optimum value
for a in prob.variables():
    print(a.name, "=", a.varValue)
```

```
ticket_1B = 8.0
ticket_1M = 5.0
ticket_1Y = 4.0
ticket_2B = 9.0
ticket_2M = 0.0
ticket_2Y = 8.0
ticket_3B = 10.0
ticket_3M = 0.0
ticket_3Y = 3.0
```

Show the optimal value.

In [12]:
```
print("Total revenue can = ", value(prob.objective))
```

```
Total revenue can =  9790.0
```

Other way to write the final results.

In [13]:
```
print(LpStatus[prob.status])
for i in prob.variables():
    print("Variable {0} = {1}".format(i.name, i.varValue))
print("Objective function z = {0}".format(value(prob.objective)))
```

```
Optimal
Variable ticket_1B = 8.0
Variable ticket_1M = 5.0
Variable ticket_1Y = 4.0
Variable ticket_2B = 9.0
Variable ticket_2M = 0.0
Variable ticket_2Y = 8.0
Variable ticket_3B = 10.0
Variable ticket_3M = 0.0
Variable ticket_3Y = 3.0
Objective function z = 9790.0
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: