In [2]:
```
from pulp import LpMaximize, LpMinimize, LpProblem, LpStatus, lpSum, LpVariable
import pandas as pd
import numpy as np
```

A movie production company will be shooting on location in Atlanta, GA for five months. During this time period, the production company will require storage space for its sets, props and costumes, which it can lease from a local warehouse. The warehouse leases storage space by the square foot, at prices shown in Table 1 below, depending on the duration of the lease. The warehouse will allow the production company to have multiple leases for space in any particular month and to lease space starting in any month for any duration. For example, the production company could sign a 3 month lease for 30,000 sq ft in month 1 and then add 1 month lease for an additional 20,000 sq ft in month 3.

Table 1: Lease Cost

Cost per sq. ft.	Lease Duration
$ 65	1 month
$ 100	2 months
$ 135	3 months
$ 160	4 months
$ 190	5 months

The production company has varying needs for storage space over the course of the five month shoot. The space they estimate they will need in each month is shown in Table 2.

Table 2: Space Required

Month	Sq. Ft.
1	30,000
2	20,000
3	40,000
4	10,000
5	50,000

Formulate and solve a LP model to determine the optimal leasing plan for the production company, in order to have all the storage space it needs in each month at the minimum cost.

In [106]:
```
durations = np.arange(1, 6)
months = np.arange(1, 6)

start_month = 1
end_month = 5

space_required = dict(zip([m for m in months], [30000, 20000, 40000, 10000, 50000

cost_per_sqft = dict(zip([d for d in durations], [65, 100, 135, 160, 190]))
cost_per_sqft
```

Out[106]:
```
{1: 65, 2: 100, 3: 135, 4: 160, 5: 190}
```

In [50]:
```
leases = []

for m in months :
    for d in durations :
        if m + d - 1 <= end_month :
            leases.append((m, d))

leases
```

Out[50]:
```
[(1, 1),
 (1, 2),
 (1, 3),
 (1, 4),
 (1, 5),
 (2, 1),
 (2, 2),
 (2, 3),
 (2, 4),
 (3, 1),
 (3, 2),
 (3, 3),
 (4, 1),
 (4, 2),
 (5, 1)]
```

In [72]:
```
model = LpProblem("Lease_Plan", LpMinimize)
```

In [73]:
```
# Construct Decision Variables

plan = LpVariable.dicts("sqft", ((l) for l in leases),
                                lowBound = 0, cat = 'Integer')
plan
```

Out[73]:
```
{(1, 1): sqft_(1,_1),
 (1, 2): sqft_(1,_2),
 (1, 3): sqft_(1,_3),
 (1, 4): sqft_(1,_4),
 (1, 5): sqft_(1,_5),
 (2, 1): sqft_(2,_1),
 (2, 2): sqft_(2,_2),
 (2, 3): sqft_(2,_3),
 (2, 4): sqft_(2,_4),
 (3, 1): sqft_(3,_1),
 (3, 2): sqft_(3,_2),
 (3, 3): sqft_(3,_3),
 (4, 1): sqft_(4,_1),
 (4, 2): sqft_(4,_2),
 (5, 1): sqft_(5,_1)}
```

In [74]:
```
#note on indexing

cost_per_sqft[leases[3][1]]
```

Out[74]:
```
160
```

In [75]:
```
# Add the objective function to the model
model += lpSum(cost_per_sqft[ls[1]] * plan[ls] for ls in leases)
model
```

Out[75]:
```
Lease_Plan:
MINIMIZE
65*sqft_(1,_1) + 100*sqft_(1,_2) + 135*sqft_(1,_3) + 160*sqft_(1,_4) + 190*sqft
_(1,_5) + 65*sqft_(2,_1) + 100*sqft_(2,_2) + 135*sqft_(2,_3) + 160*sqft_(2,_4)
+ 65*sqft_(3,_1) + 100*sqft_(3,_2) + 135*sqft_(3,_3) + 65*sqft_(4,_1) + 100*sqf
t_(4,_2) + 65*sqft_(5,_1) + 0
VARIABLES
0 <= sqft_(1,_1) Integer
0 <= sqft_(1,_2) Integer
0 <= sqft_(1,_3) Integer
0 <= sqft_(1,_4) Integer
0 <= sqft_(1,_5) Integer
0 <= sqft_(2,_1) Integer
0 <= sqft_(2,_2) Integer
0 <= sqft_(2,_3) Integer
0 <= sqft_(2,_4) Integer
0 <= sqft_(3,_1) Integer
0 <= sqft_(3,_2) Integer
0 <= sqft_(3,_3) Integer
0 <= sqft_(4,_1) Integer
0 <= sqft_(4,_2) Integer
0 <= sqft_(5,_1) Integer
```

In [88]:
```
## note on creating the columns
m = 2
d = 2
i = 1

[1 if m + d - 1 >= i and i >= m else 0 for i in range(1,len(durations) + 1)]
```

Out[88]:
```
[0, 1, 1, 0, 0]
```

In [105]:
```
matrix = pd.DataFrame()

for i in range(len(leases)) :
    m = leases[i][0]
    d = leases[i][1]
    matrix[i] = [1 if m + d - 1 >= i and i >= m else 0 for i in range(1,len(durat

matrix.columns = leases
matrix['month'] = months
matrix = matrix.set_index('month')
matrix
```

Out[105]:

month	(1, 1)	(1, 2)	(1, 3)	(1, 4)	(1, 5)	(2, 1)	(2, 2)	(2, 3)	(2, 4)	(3, 1)	(3, 2)	(3, 3)	(4, 1)	(4, 2)	(5, 1)
1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
2	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0
3	0	0	1	1	1	0	1	1	1	1	1	1	0	0	0
4	0	0	0	1	1	0	0	1	1	0	1	1	1	1	0
5	0	0	0	0	1	0	0	0	1	0	0	1	0	1	1

In [120]:
```
# Add the constraints to the model
for m in months:
    model += (lpSum(plan[ls] * matrix.loc[m][ls] for ls in leases) >= space_requi

model
```

Out[120]:
```
Lease_Plan:
MINIMIZE
65*sqft_(1,_1) + 100*sqft_(1,_2) + 135*sqft_(1,_3) + 160*sqft_(1,_4) + 190*sqft
_(1,_5) + 65*sqft_(2,_1) + 100*sqft_(2,_2) + 135*sqft_(2,_3) + 160*sqft_(2,_4)
+ 65*sqft_(3,_1) + 100*sqft_(3,_2) + 135*sqft_(3,_3) + 65*sqft_(4,_1) + 100*sqf
t_(4,_2) + 65*sqft_(5,_1) + 0
SUBJECT TO
_C1: sqft_(1,_1) + sqft_(1,_2) + sqft_(1,_3) + sqft_(1,_4) + sqft_(1,_5)
 >= 30000

_C2: sqft_(1,_2) + sqft_(1,_3) + sqft_(1,_4) + sqft_(1,_5) + sqft_(2,_1)
 + sqft_(2,_2) + sqft_(2,_3) + sqft_(2,_4) >= 20000

_C3: sqft_(1,_3) + sqft_(1,_4) + sqft_(1,_5) + sqft_(2,_2) + sqft_(2,_3)
 + sqft_(2,_4) + sqft_(3,_1) + sqft_(3,_2) + sqft_(3,_3) >= 40000

_C4: sqft_(1,_4) + sqft_(1,_5) + sqft_(2,_3) + sqft_(2,_4) + sqft_(3,_2)
 + sqft_(3,_3) + sqft_(4,_1) + sqft_(4,_2) >= 10000

_C5: sqft_(1,_5) + sqft_(2,_4) + sqft_(3,_3) + sqft_(4,_2) + sqft_(5,_1)
 >= 50000

VARIABLES
0 <= sqft_(1,_1) Integer
0 <= sqft_(1,_2) Integer
0 <= sqft_(1,_3) Integer
0 <= sqft_(1,_4) Integer
0 <= sqft_(1,_5) Integer
0 <= sqft_(2,_1) Integer
0 <= sqft_(2,_2) Integer
0 <= sqft_(2,_3) Integer
0 <= sqft_(2,_4) Integer
0 <= sqft_(3,_1) Integer
0 <= sqft_(3,_2) Integer
0 <= sqft_(3,_3) Integer
0 <= sqft_(4,_1) Integer
0 <= sqft_(4,_2) Integer
0 <= sqft_(5,_1) Integer
```

In [121]:
```
model.solve()
```

Out[121]:
```
1
```

In [122]:
```
model.objective.value()
```

Out[122]:
```
7650000.0
```

In [123]:
```
for v in model.variables(): print(f"{v.name}: {v.varValue}")
```

```
sqft_(1,_1): 0.0
sqft_(1,_2): 0.0
sqft_(1,_3): 0.0
sqft_(1,_4): 0.0
sqft_(1,_5): 30000.0
sqft_(2,_1): 0.0
sqft_(2,_2): 0.0
sqft_(2,_3): 0.0
sqft_(2,_4): 0.0
sqft_(3,_1): 10000.0
sqft_(3,_2): 0.0
sqft_(3,_3): 0.0
sqft_(4,_1): 0.0
sqft_(4,_2): 0.0
sqft_(5,_1): 20000.0
```

In [ ]:

In [ ]:

In [ ]: