Import and Build Functions

In [1]:
```
import gurobipy as gp
from gurobipy import GRB
import numpy as np
import pandas as pd


#Primal Function
def myf(x,m,v):
    s=x.X
    rc=x.RC
    ll=x.SAObjLow
    ul=x.SAObjUp
    mydf=pd.DataFrame({'X':s, 'Reduced_Cost':rc, 'Lower_Limits:':ll, 'Upper_
    return(mydf)

#Dual Function
def myf2(m,const):
    duals=m.pi
    slack=np.round(m.slack,3)
    RHSLL=np.round(m.SARHSLow,3)
    RHSUL=np.round(m.SARHSUp,3)
    mydf2=pd.DataFrame({'Pi':duals, 'Slack':slack, 'RHS_Lower_Limit:':RHSLL,
    return(mydf2)
```

Problem 1

1A.

Write the general dual problem associated with the given LP. (Do not transform or rewrite the primal problem before writing the general dual)

$$
\begin{array}{cccc}
Maximize & & & \\
-4x_1 & +2x_2 & +0x_3 & \\
\hline
4x_1 & +1x_2 & +1x_3 & = 20 \\
2x_1 & -1x_2 & +0x_3 & \geq 6 \\
1x_1 & -1x_2 & +5x_3 & \geq -5 \\
-3x_1 & +2x_2 & +1x_3 & \leq 4 \\
\hline
x_1 \leq 0 & x_2 \geq 0 & x_3 = free &
\end{array}
$$

Solution

Table 6.13 from the textbook provides the directionality of the dual variables. It shows these constraint directions as 'sensible,' 'odd,' or 'bizarre.' When the constraint is ≥ in the primal, the variable is ≤ in the dual. When the constraint is ≥ in the primal, the variable is ≤ in the dual. When the constraint is ≤ in the primal. the variable is ≥ in the dual.

Step 1. We first transpose the constrain matrix without the RHS.

Step 2. The profit / cost vector then becomes the RHS.

Step 3. The RHS becomes the cost / profit vector.

$$
\begin{array}{cccc|c}
Maximize & & & & \\
-20\pi_1 & 6\pi_2 & -5\pi_3 & +4\pi_4 & \\
\hline
4\pi_1 & 2\pi_2 & 1\pi_3 & -3\pi_4 & \leq -4 \\
1\pi_1 & -1\pi_2 & -1\pi_3 & 2\pi_4 & \geq 2 \\
1\pi_1 & & 5\pi_3 & 1\pi_4 & = 0 \\
\hline
\pi_1 = free & \pi_2 \leq 0 & \pi_3 \leq 0 & \pi_4 \geq 0 &
\end{array}
$$

1B

1B. Given the following information for a product-mix problem with three products and three resources.
Primal Decision Variables: x_1 = number of unit 1 produced; x_2 = # of unit 2 produced; x_3 = # of unit 3 produced

Primal Formulation:

Max Z (Rev.) =	$25x_1$	$+ 30x_2$	$+ 20x_3$		
Subject To	$8x_1$	$+ 6x_2$	$+ x_3$	≤ 50	(Res. 1 constraint)
	$4x_1$	$+ 2x_2$	$+ 3x_3$	≤ 20	(Res. 2 constraint)
	$2x_1$	$+ x_2$	$+ 2x_3$	≤ 25	(Res. 3 constraint)
		x_1, x_2, x_3		≥ 0	(Nonnegativity)

Dual Formulation:

Min W =	$50\pi_1$	$+ 20\pi_2$	$+25\pi_3$	
Subject To	$8\pi_1$	$+ 4\pi_2$	$+2\pi_3$	≥ 25
	$6\pi_1$	$+ 2\pi_2$	$+\pi_3$	≥ 30
	π_1	$+ 3\pi_2$	$+2\pi_3$	≥ 20
	π_1, π_2, π_3			≥ 0

Optimal Solution:
Optimal Z = Revenue = $268.75

$x_1 = 0$ (Number of unit 1)	Dual Var. Optimal Value = 22.5 (Surplus variable in 1st dual constraint)
$x_2 = 8.125$ (Number of unit 2)	Dual Var. Optimal Value = 0 (Surplus variable in 2nd dual constraint)
$x_3 = 1.25$ (Number of unit 3)	Dual Var. Optimal Value = 0 (Surplus variable in 3rd dual constraint)
Resource Constraints:	
Resource 1 = 0 leftover units	Dual Var. Optimal Value = 3.125 = π_1
Resource 2 = 0 leftover units	Dual Var. Optimal Value = 5.625 = π_2
Resource 3 = 14.375 leftover units	Dual Var. Optimal Value = 0 = π_3

1Bi

What is the fair-market price for one unit of Resource 3?

$\pi_3 = 0$ so the constraint is not tight, and our shadow price is ZERO. We can also see surplus for Resource 3. We would pay nothing!

1Bii

What is the meaning of the surplus variable value of 22.5 in the 1st dual constraint with respect to the primal problem?

22.5 is the reduced cost or the amount that the objective value associated with the 1st primal variable would have to improve for it to enter the basis. In this case, the profit would need to go from $25 + 22.5 = 47.5$ to enter the basis.

Problem 2

In [2]:

```
# Create a new model
m = gp.Model("matrix1")
# Set objective
obj = np.asarray([260,350]) #Profit=Revenue-Expenses
v=['Cars', 'Trucks']
const=['Machine1','Machine2','Steel', 'Cars','Trucks']
# Create variables
x = m.addMVar(shape=len(obj), vtype=GRB.CONTINUOUS, name="x")
m.setObjective(obj @ x, GRB.MAXIMIZE)
A= np.asarray(
     [[.8,1],   #Machine 1
      [.6,.7],  #Machine 2
      [2,3],     #Steel
      [-1,0],    #Min Cars
      [0,-1]     #Min Trucks
     ])
# Build rhs vector
b=np.asarray([98,73,260,-88,-26])
# Add constraints
m.addConstr(A @ x <= b)
# Optimize model
m.optimize()
```

```
Restricted license - for non-production use only - expires 2022-01-13
Gurobi Optimizer version 9.1.2 build v9.1.2rc0 (win64)
Thread count: 10 physical cores, 20 logical processors, using up to 20 threads
Optimize a model with 5 rows, 2 columns and 8 nonzeros
Model fingerprint: 0xecb10890
Coefficient statistics:
  Matrix range     [6e-01, 3e+00]
  Objective range  [3e+02, 4e+02]
  Bounds range     [0e+00, 0e+00]
  RHS range        [3e+01, 3e+02]
Presolve removed 2 rows and 0 columns
Presolve time: 0.00s
Presolved: 3 rows, 2 columns, 6 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    3.2540000e+04   0.000000e+00   0.000000e+00      0s
       0    3.2540000e+04   0.000000e+00   0.000000e+00      0s

Solved in 0 iterations and 0.00 seconds
Optimal objective  3.254000000e+04
```

Objective Function

In [3]:

```
print('Objective Value:', m.objVal)
```

```
Objective Value: 32540.0
```

Primal

In [4]:
```
1 myf(x,m,v)
```

Out[4]:

	X	Reduced_Cost	Lower_Limits:	Upper_Limits
Cars	88.0	0.0	-inf	280.0
Trucks	27.6	0.0	325.0	inf

2C1

If cars contributed $310 to profit, what would be the new optimal solution to the problem?

Adjusting for the cost, that profit is actually 270 (a 10 dollar increase) which is within the upper limits of the reduced cost. So there is no change in the X_i variables; however, there is an increase in the objective value by 10 dollars x 88 or 880 dollars. The new objective value is then 33420 dollars.

Dual

In [5]:
```
1 myf2(m,const)
```

Out[5]:

	Pi	Slack	RHS_Lower_Limit:	RHS_Upper_Limit
Machine1	350.0	0.00	96.40	98.4
Machine2	0.0	0.88	72.12	inf
Steel	0.0	1.20	258.80	inf
Cars	20.0	0.00	-90.00	-85.0
Trucks	0.0	1.60	-27.60	inf

2C2 through 2C5

2C2

What is the most that Carco should be willing to pay to rent an additional Type 1 machine for 1 day?

If you ran this with machine 1 decision variable, you would see a negative allowable increase, meaning we cannot be certain. Without that variable, we simply re-run and increase the RHS constraint by +1.

Original OF Value: 32540. New OF Value: 32760. Increase: 220. So we know that at most, we should pay under 220 for an extra day. But if you evaluate the DV's, cars go up 3 and trucks go down -1.6. We are then only using 3 x .8 - 1.6 x 1 = 0.8 of the day. So we should be willing to pay just under 220 for .8 days and nothing for the additional .2 days. The only way to get this solution is to re-run.

2C3

What is the most that Carco should be willing to pay for an extra ton of steel? The slack is 1.20 meaning we have excess. We should pay zero dollars.

2C4

If Carco were required to produce at least 86 cars, what would Carco's profit become? This is an increase in 2 cars to 86 (-86 in our formulation), and the allowable increase is between 85 and 90 cars. We can then just use the objective function value + the shadow price for cars x the number of additional cars since we are within this range.

32540 + 20 (shadow price for cars) x 2 (number of additional cars produced) = 32580

2C5

Carco is considering producing jeeps. A jeep contributes 600 to profit and requires 1.2 days on machine 1, 2 days on machine 2, and 4 tons of steel. Should Carco produce any jeeps?

First, that profit is actually z3=600-1.2 x 50 = 540 in our formulatioon. The question then is if the production vector (x3) of [1.2,2,4,0,0] is in the basis. You could simply re-solve. Or you could compare the contribution of the jeeps to cars and trucks using shadow prices (duals). If we take the dot product of the the new production vector (x3) and the shadow price vector (pi), we are calculating the expected profit, which is 420. If the actual profit of 540 exceeds that, then we know to produce jeeps, as it is more profitable than we would expect. Formally, we calculate the reduced cost for the new variable as x3 * pi - z3.

In [6]:
```
x3=[1.2,2,4,0,0]
pi=[350,0,0,20,0]
z3=540
np.dot(x3,pi)-z3
```

Out[6]:
```
-120.0
```

Problem 3

In [7]:

```
# Create a new model
m = gp.Model("matrix1")
# Set objective
obj = np.asarray([20,10,6,20,10,6,20,10,20])
v=['Day 1 Buy', 'Day 1 FC', 'Day 1 SC', 'Day 2 Buy', 'Day 2 FC', 'Day 2 SC',
const=['Day 1 Demand', 'Day 2 Demand', 'Day 3 Demand','Day 4 Demand',
         'On-hand >= clean Day 1', 'On-hand >= clean Day 2', 'On-hand >= cle
# Create variables
x = m.addMVar(shape=len(obj), vtype=GRB.CONTINUOUS, name="x")
m.setObjective(obj @ x, GRB.MINIMIZE)
A= np.asarray(
     [[1,0,0,0,0,0,0,0,0],    #Day 1
      [0,1,0,1,0,0,0,0,0],    #Day 2
      [0,0,1,0,1,0,1,0,0],    #Day 3
      [0,0,0,0,0,1,0,1,1],    #Day 4
      [1,-1,-1,0,0,0,0,0,0],  #OH>=Clean Day 1
      [1,0,-1,1,-1,-1,0,0,0], #OH>=Clean Day 2
      [1,0,0,1,0,-1,1,-1,0]   #OH>=Clean Day 3
     ])
# Build rhs vector
b =np.asarray([15,12,18,6,0,0,0])
# Add constraints
m.addConstr(A @ x >= b)
# Optimize model
m.optimize()
```

```
Gurobi Optimizer version 9.1.2 build v9.1.2rc0 (win64)
Thread count: 10 physical cores, 20 logical processors, using up to 20 threads
Optimize a model with 7 rows, 9 columns and 22 nonzeros
Model fingerprint: 0x3c17c256
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [6e+00, 2e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [6e+00, 2e+01]
Presolve removed 1 rows and 0 columns
Presolve time: 0.00s
Presolved: 6 rows, 9 columns, 21 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    3.0000000e+02   3.600000e+01   0.000000e+00      0s
       6    6.6600000e+02   0.000000e+00   0.000000e+00      0s

Solved in 6 iterations and 0.01 seconds
Optimal objective  6.660000000e+02
```

In [8]:

```
print('Objective Value:', m.objVal)
```

```
Objective Value: 666.0
```

Primal

In [9]:

```
myf(x,m,v)
```

Out[9]:

	X	Reduced_Cost	Lower_Limits:	Upper_Limits
Day 1 Buy	15.0	0.0	10.0	inf
Day 1 FC	9.0	0.0	6.0	12.0
Day 1 SC	6.0	0.0	4.0	10.0
Day 2 Buy	3.0	0.0	18.0	24.0
Day 2 FC	12.0	0.0	6.0	12.0
Day 2 SC	0.0	2.0	4.0	inf
Day 3 Buy	0.0	4.0	16.0	inf
Day 3 FC	6.0	0.0	0.0	12.0
Day 4 Buy	0.0	10.0	10.0	inf

Dual

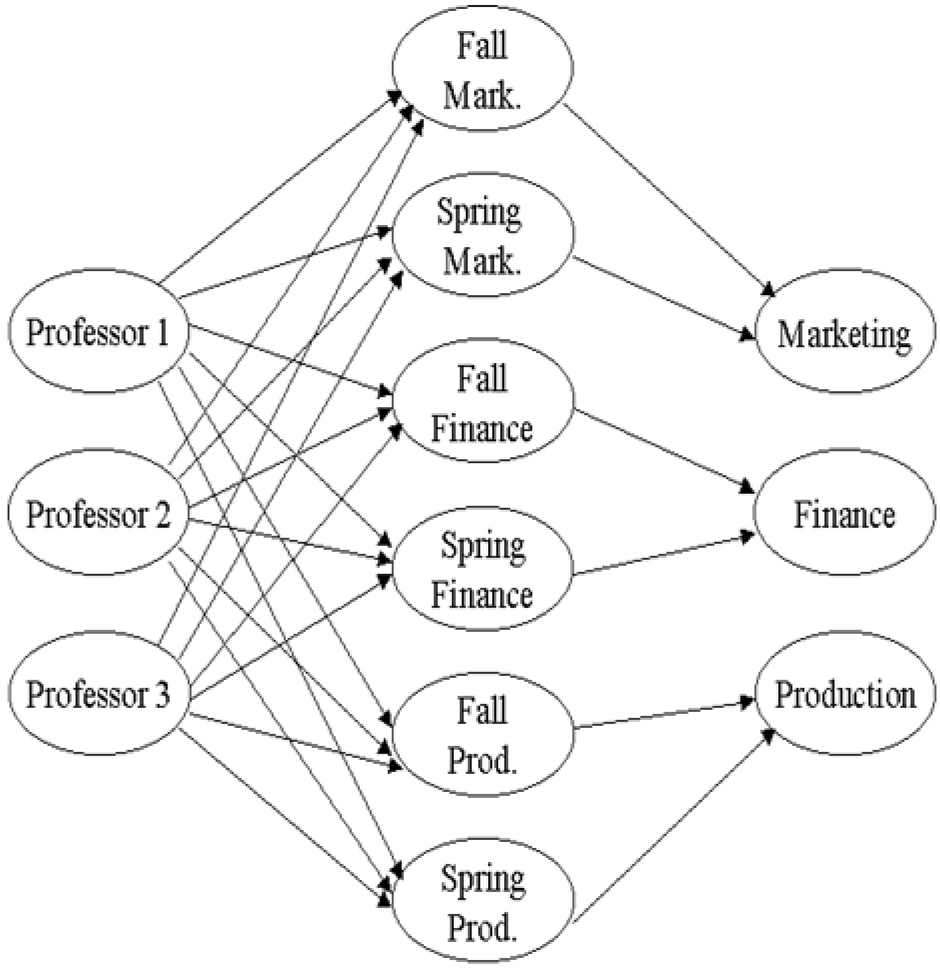
In [10]:

```
myf2(m,const)
```

Out[10]:

	Pi	Slack	RHS_Lower_Limit:	RHS_Upper_Limit
Day 1 Demand	10.0	0.0	6.0	18.0
Day 2 Demand	14.0	0.0	3.0	18.0
Day 3 Demand	16.0	0.0	15.0	27.0
Day 4 Demand	10.0	0.0	-0.0	18.0
On-hand >= clean Day 1	4.0	0.0	-12.0	6.0
On-hand >= clean Day 2	6.0	0.0	-3.0	9.0
On-hand >= clean Day 3	0.0	-12.0	-inf	12.0

Problem 4


Fall Mark.
Spring Mark.
Fall Finance
Spring Finance
Fall Prod.
Spring Prod.
Professor 1
Professor 2
Professor 3
Marketing
Finance
Production

In [11]:

```
# Create a new model
m = gp.Model("matrix1")
# Set objective
obj = np.asarray([9,8,7,10,9,8,9,11,10,7,9,8,9,8,10,9,8,10])
# Create variables
x = m.addMVar(shape=len(obj), vtype=GRB.CONTINUOUS, name="x")
v=['X111','X112','X113','X121','X122','X123',
       'X211','X212','X213','X221','X222','X223',
       'X311','X312','X313','X321','X322','X323']
const=['P1 4 Courses','P2 4 Courses', 'P3 4 Courses',
        'Fall Marketing','Fall Finance', 'Fall Production',
        'Spring Marketing','Spring Finance', 'Spring Production',
        'Total Marketing', 'Total Finance', 'Total Production']
m.setObjective(obj @ x, GRB.MAXIMIZE)
A= np.asarray(
     [[-1,-1,-1,-1,-1,-1,0,0,0,0,0,0,0,0,0,0,0,0], #Prof1
      [0,0,0,0,0,0,-1,-1,-1,-1,-1,-1,0,0,0,0,0,0], #Prof2
      [0,0,0,0,0,0,0,0,0,0,0,0,-1,-1,-1,-1,-1,-1], #Prof3
      [1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0], #Marketing Fall
      [0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0], #Finance Fall
      [0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0], #Production Fall
      [0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,0], #Marketing Spring
      [0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,0], #Finance Spring
      [0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1], #Production Spring
      [1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0],#Marketing Year
      [0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0],#Finance Year
      [0,0,1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,1] #Production Year
     ])
# Build rhs vector
b =np.asarray([-4,-4,-4,1,1,1,1,1,1,4,4,4])
# Add constraints
m.addConstr(A@x>=b)
# Optimize model
m.optimize()
```

```
Gurobi Optimizer version 9.1.2 build v9.1.2rc0 (win64)
Thread count: 10 physical cores, 20 logical processors, using up to 20 threads
Optimize a model with 12 rows, 18 columns and 54 nonzeros
Model fingerprint: 0xddfc0075
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [7e+00, 1e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+00, 4e+00]
Presolve time: 0.00s
Presolved: 12 rows, 18 columns, 54 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    1.5900000e+32   1.800000e+31   1.590000e+02      0s
       8    1.2100000e+02   0.000000e+00   0.000000e+00      0s

Solved in 8 iterations and 0.00 seconds
Optimal objective  1.210000000e+02
```

In [12]:
```
print('Objective Value:', m.objVal)
```

```
Objective Value: 121.0
```

Primal

In [13]:
```
myf(x,m,v)
```

Out[13]:

	X	Reduced_Cost	Lower_Limits:	Upper_Limits
X111	0.0	0.0	8.0	9.0
X112	0.0	-3.0	-inf	11.0
X113	0.0	-3.0	-inf	10.0
X121	3.0	0.0	9.0	11.0
X122	1.0	0.0	9.0	11.0
X123	0.0	-2.0	-inf	10.0
X211	1.0	0.0	9.0	10.0
X212	3.0	0.0	10.0	inf
X213	0.0	-0.0	-inf	10.0
X221	0.0	-3.0	-inf	10.0
X222	0.0	-0.0	-inf	9.0
X223	0.0	-2.0	-inf	10.0
X311	0.0	0.0	8.0	9.0
X312	0.0	-3.0	-inf	11.0
X313	1.0	0.0	10.0	10.0
X321	0.0	-1.0	-inf	10.0
X322	0.0	-1.0	-inf	9.0
X323	3.0	0.0	10.0	11.0

Duals

In [14]:
```
myf2(m,const)
```

Out[14]:

	Pi	Slack	RHS_Lower_Limit:	RHS_Upper_Limit
P1 4 Courses	-11.0	0.0	-5.0	-4.0
P2 4 Courses	-11.0	0.0	-inf	-4.0
P3 4 Courses	-11.0	0.0	-5.0	-4.0
Fall Marketing	-1.0	0.0	1.0	3.0
Fall Finance	0.0	-2.0	-inf	3.0
Fall Production	-0.0	0.0	-0.0	3.0
Spring Marketing	0.0	-2.0	-inf	3.0
Spring Finance	-2.0	0.0	-0.0	1.0
Spring Production	0.0	-2.0	-inf	3.0
Total Marketing	-1.0	0.0	3.0	4.0
Total Finance	0.0	-0.0	-inf	4.0
Total Production	-1.0	0.0	3.0	4.0