Assignment 1 : Question 2 (Optimization method)

In [1]:

```
from gurobipy import *#--------------importing required libraries
import numpy as np
```

In [2]:

```
U = [1,2,3,4,5]#------------------ As there are total 5 tools, we are numberng each tool he

S = [
    [1,0,1,0,0], #--(1,3)#-------------each row tell each toolbox
    [0,1,1,1,0], #---(2,3,4)#---------- if there is tool present in partiucalr toolbox then
    [1,1,0,0,0], #--(1,2)#----------1 means yes, 0 means NO
    [0,0,1,1,1] #---(3,4,5)

]

boxes_cost = [100,120,280,200]#----------------cost of each box
```

In [3]:

```
m = Model("Question 2 Toolbox problem")#---creating the model
```

```
Using license file C:\Users\Aloukik Aditya\gurobi.lic
Academic license - for non-commercial use only
```

In [4]:

```
C = []#------------------ creating the varibale list here

for i in range(len(S)):
    v = m.addVar(lb=0, ub=1,vtype=GRB.BINARY, name=str(i))#------------using for loop to a

    C.append(v)
```

In [5]:

```
#-- objective function
obj = 0
for v in C:
    obj += v
#m.setObjective(obj, GRB.MINIMIZE)


# obj2 = 0
# for p in p_l:
#     obj2 += p


obj2 = 0#-------------------------- multiplying each box with, C variable(which tells wethe
for i in range(len(boxes_cost)):
    obj2 += C[i]*boxes_cost[i]#



m.setObjective(obj2+obj, GRB.MINIMIZE)#------------- this will help minimizing cost and num
```

In [6]:

```
#--------------------------------setting up constrains
boxes= np.asarray(boxes_cost)

const = 0




for j in range(len(U)):
    const = 0

    for i in range(len(C)):#------------------for values
        const += S[i][j] * C[i]
    m.addConstr(const >= 1)



# price = 0


# for i in range(len(S)):#-------------------for price
#     price += boxes[i] * C[i]
#     #print(const)
#     print(boxes[i])
#     print(C[i])
# m.addConstr(price >= 1)
```

In [7]:

```
m.optimize()#-------------using optimize function
```

```
Gurobi Optimizer version 9.0.2 build v9.0.2rc0 (win64)
Optimize a model with 5 rows, 4 columns and 10 nonzeros
Model fingerprint: 0x9011abf4
Variable types: 0 continuous, 4 integer (4 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [1e+02, 3e+02]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 1e+00]
Found heuristic solution: objective 482.0000000
Presolve removed 5 rows and 4 columns
Presolve time: 0.00s
Presolve: All rows and columns removed

Explored 0 nodes (0 simplex iterations) in 0.01 seconds
Thread count was 1 (of 8 available processors)

Solution count 2: 423

Optimal solution found (tolerance 1.00e-04)
Best objective 4.230000000000e+02, best bound 4.230000000000e+02, gap 0.000
0%
```

In [8]:

```
C#--------------the final values shows that 1 means toolbox selected, 0 means not selected
```

Out[8]:

```
[<gurobi.Var 0 (value 1.0)>,
 <gurobi.Var 1 (value 1.0)>,
 <gurobi.Var 2 (value 0.0)>,
 <gurobi.Var 3 (value 1.0)>]
```

In [9]:

```
total_cost = 0
for i in range(len(boxes_cost)):#--------------------------------------getting the total co

    total_cost += round(C[i].X) * boxes_cost[i]

print("Total cost is", + total_cost)
```

```
Total cost is 420
```

In [10]:

```
const
```

Out[10]:

```
<gurobi.LinExpr: 0.0 0 + 0.0 1 + 0.0 2 + 3>
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: