OR Q6(b)

In [1]:

```
from gurobipy import *
import numpy as np
import pandas as pd

#########Parameters Set-up############
# Read the data from the csv file and use the first column as the index of rows
investment_data = pd.read_csv('investment.csv', index_col = 0)#specify the first column as index
 otherwise treated as data
print(investment_data)
print(investment_data.shape)

# Record the number of rows and columns in the data
N = investment_data.shape[0]
```

```
       Buying Price  Current Share  Current Price  Expected Future Price
Stock
S1              1.2           1000            2.1                    2.0
S2              2.1           1000            3.2                    3.7
S3              3.2           1000            4.1                    5.2
S4              4.1           1000            5.1                    7.1
S5              4.5           1000            6.7                    9.1
(5, 4)
```

In [2]:

```
#Extracting the values by ignoring the index and header of the dataframe

p = investment_data.iloc[0:N,0].values
s = investment_data.iloc[0:N,1].values
q = investment_data.iloc[0:N,2].values
r = investment_data.iloc[0:N,3].values
K = 9000

print("Buying Price",p)
print("Current Share",s)
print("Current Price:",q)
print("Expected Future Price:",r)
```

```
Buying Price [1.2 2.1 3.2 4.1 4.5]
Current Share [1000 1000 1000 1000 1000]
Current Price: [2.1 3.2 4.1 5.1 6.7]
Expected Future Price: [2.  3.7 5.2 7.1 9.1]
```

In [3]:

```
#########Model Set-up###############

m = Model("investment")

# Creat variables
# addVars ( *indices, lb=0.0, ub=GRB.INFINITY, obj=0.0, vtype=GRB.CONTINUOUS, name="" )
x = m.addVars(N)

# Set objective
m.setObjective(quicksum(r[i]*(s[i]-x[i]) for i in range(N)), GRB.MAXIMIZE)

# Add constraints:
m.addConstr(quicksum((0.99*q[i]*x[i]-0.3*(q[i]-p[i])*x[i]) for i in range(N)) >= K)
m.addConstrs(x[i] <= s[i] for i in range(N))

# Solving the model
m.optimize()

#  Print optimal solutions and optimal value
for v in m.getVars():
    print(v.VarName, v.x)

print('Obj:', m.objVal)
```

```
Academic license - for non-commercial use only
Optimize a model with 6 rows, 5 columns and 10 nonzeros
Coefficient statistics:
  Matrix range     [1e+00, 6e+00]
  Objective range  [2e+00, 9e+00]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+03, 9e+03]
Presolve removed 5 rows and 0 columns
Presolve time: 0.15s
Presolved: 1 rows, 5 columns, 5 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    2.7100000e+04   5.625000e+02   0.000000e+00      0s
       1    1.5356791e+04   0.000000e+00   0.000000e+00      0s

Solved in 1 iterations and 0.25 seconds
Optimal objective  1.535679090e+04
C0 1000.0
C1 1000.0
C2 1000.0
C3 118.76184459886306
C4 0.0
Obj: 15356.790903348072
```