Optimization Homework 1

Problem 1

code:

```
from gurobipy import *


# create a model
m = Model("problem 1")

# create variables
# x1 := 1000 barrels of oil
# x2 := 1000 barrels of aviation fuel
# x3 := 1000 barrels of heating oil
# x4 := 1000 barrels of processed aviation fuel
# x5 := 1000 barrels of processed heating oil
x1 = m.addVar(vtype=GRB.CONTINUOUS, name="x1", lb=0)
x2 = m.addVar(vtype=GRB.CONTINUOUS, name="x2", lb=0)
x3 = m.addVar(vtype=GRB.CONTINUOUS, name="x3", lb=0)
x4 = m.addVar(vtype=GRB.CONTINUOUS, name="x4", lb=0)
x5 = m.addVar(vtype=GRB.CONTINUOUS, name="x5", lb=0)

# integrate new variables
m.update()

# set objective
m.setObjective(
    -40*x1 + 60*(x2-x4) + 40*(x3-x5) + 130*x4 + 90*x5,
    GRB.MAXIMIZE
)

# add constraints
# we can only buy 20,000 barrels of oil a day
m.addConstr(x1 <= 20, "c0")
# 1000 barrels of oil can yields 500 barrels of aviation fuel
# and 500 barrels of heating oil
m.addConstr(0.5*x1 - x2 >= 0, "c1")
m.addConstr(0.5*x1 - x3 >= 0, "c2")
# we only have 8hrs cracker time per day
m.addConstr(60*x4 + 45*x5 <= 60*8, "c3")
# cracked aviation fuel < aviation fuel
m.addConstr(x2 - x4 >= 0, "c4")
# cracked heating oil < heating oil
m.addConstr(x3 - x5 >= 0, "c5")

# optimize
m.optimize()
print("Model status: ", m.status)
```

```
# print out decision variables
for v in m.getVars():
    print(v.varName, v.x, "\n")

print("-"*15)
print("Obj Value: ", m.objVal)
```

results:

```
Academic license - for non-commercial use only
Optimize a model with 6 rows, 5 columns and 11 nonzeros
Coefficient statistics:
  Matrix range     [5e-01, 6e+01]
  Objective range  [4e+01, 7e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [2e+01, 5e+02]
Presolve removed 6 rows and 5 columns
Presolve time: 0.01s
Presolve: All rows and columns removed
Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    7.6000000e+02   0.000000e+00   0.000000e+00      0s

Solved in 0 iterations and 0.01 seconds
Optimal objective  7.600000000e+02
('Model status: ', 2)
('x1', 20.0, '\n')
('x2', 10.0, '\n')
('x3', 10.0, '\n')
('x4', 8.0, '\n')
('x5', 0.0, '\n')
---------------
('Obj Value: ', 760.0)
```

Problem 2

code:

```
from gurobipy import *


# create a model
m = Model("problem 2")

# create variables
# x1 := produce times of process 1
# x2 := produce times of process 2
```

```
# x3 := hiring hours
x1 = m.addVar(vtype=GRB.CONTINUOUS, name="x1", lb=0)
x2 = m.addVar(vtype=GRB.CONTINUOUS, name="x2", lb=0)
x3 = m.addVar(vtype=GRB.CONTINUOUS, name="x3", lb=0)

# integrate new variables
m.update()

# set objective
m.setObjective(
    15*x1 + 25*x2 - 100*x3 - 3*1*x1 - 2*2*x1 - 3*2*x2 - 2*3*x2,
    GRB.MAXIMIZE
)

# add constraints
# max labors are 20000
# max chemicals are 35000
m.addConstr(x1 + 2*x2 <= 20000, "c1")
m.addConstr(2*x1 + 3*x2 <= 35000, "c2")
# the number of product should larger than
# the number of sells
m.addConstr(3*x1 + 5*x2 - 200*x3 - 1000 == 0, "c3")

# optimize
m.optimize()
print("Model status: ", m.status)

# print out decision variables
for v in m.getVars():
    print(v.varName, v.x, "\n")

print("-"*15)
print("Obj Value: ", m.objVal)
```

result:

```
Optimize a model with 3 rows, 3 columns and 7 nonzeros
Coefficient statistics:
  Matrix range     [1e+00, 2e+02]
  Objective range  [8e+00, 1e+02]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+03, 4e+04]
Presolve time: 0.01s
Presolved: 3 rows, 3 columns, 7 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    2.9000000e+31   4.437500e+30   2.900000e+01      0s
       3    1.1800000e+05   0.000000e+00   0.000000e+00      0s

Solved in 3 iterations and 0.01 seconds
Optimal objective  1.180000000e+05
```

```
('Model status: ', 2)
('x1', 10000.0, '\n')
('x2', 5000.0, '\n')
('x3', 270.0, '\n')
---------------
('Obj Value: ', 118000.0)
```

Problem 4

code:

```
from gurobipy import *


# create a model
m = Model("problem 4")

# create variables
# ci := crude oil used in method i
# gi := barrels of i grade gas
# hi := barrels of i grade heating oil
# 628 := barrels of grade 6 products cracked into grade 8 products
# 8210 := barrels of grade 8 products cracked into grade 10 products
xc1 = m.addVar(vtype=GRB.CONTINUOUS, name="xc1", lb=0)
xc2 = m.addVar(vtype=GRB.CONTINUOUS, name="xc2", lb=0)
xc3 = m.addVar(vtype=GRB.CONTINUOUS, name="xc3", lb=0)
xg6 = m.addVar(vtype=GRB.CONTINUOUS, name="xg6", lb=0)
xg8 = m.addVar(vtype=GRB.CONTINUOUS, name="xg8", lb=0)
xg10 = m.addVar(vtype=GRB.CONTINUOUS, name="xg10", lb=0)
xh6 = m.addVar(vtype=GRB.CONTINUOUS, name="xh6", lb=0)
xh8 = m.addVar(vtype=GRB.CONTINUOUS, name="xh8", lb=0)
xh10 = m.addVar(vtype=GRB.CONTINUOUS, name="xh10", lb=0)
x628 = m.addVar(vtype=GRB.CONTINUOUS, name="x628", lb=0)
x8210 = m.addVar(vtype=GRB.CONTINUOUS, name="x8210", lb=0)

# integrate new variables
m.update()

# set objective
m.setObjective(
    12*(xg6 + xg8 + xg10) + 5*(xh6 + xh8 + xh10) - 3.4*xc1 - 3*xc2 -
2.6*xc3 - 1*x628 - 1.5*x8210,
    GRB.MAXIMIZE
)

# add constraints
m.addConstr(xg6 + xg8 + xg10 <= 2000)
m.addConstr(xh6 + xh8 + xh10 <= 600)
m.addConstr(6*xg6 + 8*xg8 + 10*xg10 >= 9*(xg6 + xg8 + xg10))
m.addConstr(6*xh6 + 8*xh8 + 10*xh10 >= 7*(xh6 + xh8 + xh10))
m.addConstr(0.3*xc1 + 0.4*xc2 + 0.3*xc3 == xg6 + xh6 + x628)
```

```
m.addConstr(0.5*xc1 + 0.2*xc2 + 0.3*xc3 == xg8 + xh8 + x8210 - x628)
m.addConstr(0.8*xc1 + 0.4*xc2 + 0.2*xc3 == xg10 + xh10 - x8210)

# optimize
m.optimize()
print("Model status: ", m.status)

# print out decision variables
for v in m.getVars():
    print(v.varName, v.x, "\n")

print("-"*15)
print("Obj Value: ", m.objVal)
```

result:

```
Optimize a model with 7 rows, 11 columns and 31 nonzeros
Coefficient statistics:
  Matrix range     [2e-01, 3e+00]
  Objective range  [1e+00, 1e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [6e+02, 2e+03]
Presolve removed 1 rows and 1 columns
Presolve time: 0.01s
Presolved: 6 rows, 10 columns, 33 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    1.5320000e+32   4.950000e+30   1.532000e+02      0s
       7    2.1475000e+04   0.000000e+00   0.000000e+00      0s

Solved in 7 iterations and 0.01 seconds
Optimal objective  2.147500000e+04
('Model status: ', 2)
('xc1', 1625.0, '\n')
('xc2', 0.0, '\n')
('xc3', 0.0, '\n')
('xg6', 300.0, '\n')
('xg8', 400.0, '\n')
('xg10', 1300.0, '\n')
('xh6', 187.5, '\n')
('xh8', 412.5, '\n')
('xh10', 0.0, '\n')
('x628', 0.0, '\n')
('x8210', 0.0, '\n')
---------------
('Obj Value: ', 21475.0)
```

Problem 5

code:

```
from gurobipy import *


# create a model
m = Model("problem 5")

# create variables
# xi := represent i stock
x1 = m.addVar(vtype=GRB.CONTINUOUS, name="x1", lb=0)
x2 = m.addVar(vtype=GRB.CONTINUOUS, name="x2", lb=0)
x3 = m.addVar(vtype=GRB.CONTINUOUS, name="x3", lb=0)
x4 = m.addVar(vtype=GRB.CONTINUOUS, name="x4", lb=0)
x5 = m.addVar(vtype=GRB.CONTINUOUS, name="x5", lb=0)
x6 = m.addVar(vtype=GRB.CONTINUOUS, name="x6", lb=0)
x7 = m.addVar(vtype=GRB.CONTINUOUS, name="x7", lb=0)
x8 = m.addVar(vtype=GRB.CONTINUOUS, name="x8", lb=0)
x9 = m.addVar(vtype=GRB.CONTINUOUS, name="x9", lb=0)
x10 = m.addVar(vtype=GRB.CONTINUOUS, name="x10", lb=0)

# integrate new variables
m.update()

# set objective
m.setObjective(
    36*x1 + 39*x2 + 42*x3 + 45*x4 + 51*x5 + 55*x6 + 63*x7 + 64*x8 + 66*x9
+ 70*x10,
    GRB.MAXIMIZE
)

# add constraints
m.addConstr((100 - x1)*(0.99*30 - 0.3*10) + (100 - x2)*(0.99*34 - 0.3*9) +
(100 - x3)*(0.99*43 - 0.3*13) + (100 - x4)*(0.99*47 - 0.3*12) + (100 -
x5)*(0.99*49 - 0.3*9) + (100 - x6)*(0.99*53 - 0.3*8) + (100 - x7)*(0.99*60
- 0.3*10) + (100 - x8)*(0.99*62 - 0.3*7) + (100 - x9)*(0.99*64 - 0.3*4) +
(100 - x10)*(0.99*66 - 0.3*1) == 30000, "c1")
m.addConstr(x1 <= 100)
m.addConstr(x2 <= 100)
m.addConstr(x3 <= 100)
m.addConstr(x4 <= 100)
m.addConstr(x5 <= 100)
m.addConstr(x6 <= 100)
m.addConstr(x7 <= 100)
m.addConstr(x8 <= 100)
m.addConstr(x9 <= 100)
m.addConstr(x10 <= 100)

# optimize
m.optimize()
print("Model status: ", m.status)

# print out decision variables
for v in m.getVars():
```

```
    print(v.varName, v.x, "\n")

print("-"*15)
print("Obj Value: ", m.objVal)
```

result:

```
Optimize a model with 11 rows, 10 columns and 20 nonzeros
Coefficient statistics:
  Matrix range     [1e+00, 7e+01]
  Objective range  [4e+01, 7e+01]
  Bounds range     [0e+00, 0e+00]
  RHS range        [1e+02, 2e+04]
Presolve removed 10 rows and 0 columns
Presolve time: 0.01s
Presolved: 1 rows, 10 columns, 10 nonzeros

Iteration    Objective       Primal Inf.    Dual Inf.      Time
       0    5.3100000e+04   4.687500e+02   0.000000e+00      0s
       1    2.0893709e+04   0.000000e+00   0.000000e+00      0s

Solved in 1 iterations and 0.01 seconds
Optimal objective  2.089370881e+04
('Model status: ', 2)
('x1', 100.0, '\n')
('x2', 100.0, '\n')
('x3', 0.0, '\n')
('x4', 0.0, '\n')
('x5', 100.0, '\n')
('x6', 36.24925104853209, '\n')
('x7', 100.0, '\n')
('x8', 0.0, '\n')
('x9', 0.0, '\n')
('x10', 0.0, '\n')
---------------
('Obj Value: ', 20893.708807669267)
```