# ***HOMEWORK 2 ANSWERS***

## ***Answer 1***

First, lets start with naming courses.

## ***#Decision Variables***

C1 -> Calculus

C2 -> Operations Research

C3 -> Data Structures

C4 -> Business Statistics

C5 -> Computer Simulation

C6 -> Introduction to Computer Programming

C7 -> Forecasting

***What we want is minimize the following formula;***

## ***#Objective Function***

MinZ = C1 + C2 + C3 + C4 + C5 + C6 + C7

A condition in the question is => "***a student must complete at least two math courses, at least two OR courses, and at least two computer courses"***

## ***#Constraints***

We are going to build our conditions around this request.

-For Math Course requirements;

C1+C2+C3+C4+C7 >= 2

-For OR Course requirements;

C2+C4+C5+C7 >= 2

-For Computer Course Requirements;

C3+C5+C6 >= 2

Also there are prerequisites conditions;

***"Calculus is a prerequisite for business statistics"***

C4 <= C1

***"introduction to computer programming is a prerequisite for computer simulation and for data structures"***

C3 <= C6

C5 <= C6

***"business statistics is a prerequisite for forecasting"***

C7 <= C4

Also any course can be 0 or 1, 0 <= Ci <= 1

When we combining all these constraints we got the appropiate IP;

MinZ = C1 + C2 + C3 + C4 + C5 + C6 + C7

C1+C2+C3+C4+C7 >= 2

C2+C4+C5+C7 >= 2

C3+C5+C6 >= 2

C4 <= C1

C3 <= C6

C5 <= C6

C7 <= C4

0 <= Ci <= 1

## ***PULP SOLVE CODE AND OUTPUT***

```
from pulp import LpVariable, LpMinimize, LpStatus, LpProblem


prob = LpProblem("Minimize_Course_Selection",LpMinimize)


#Decision Variables
c1 = LpVariable("Calculus", lowBound=0, upBound=1, cat="Integer")
c2 = LpVariable("Operations Research", lowBound=0, upBound=1, cat="Integer")
c3 = LpVariable("Data Structures", lowBound=0, upBound=1, cat="Integer")
c4 = LpVariable("Business Statistics", lowBound=0, upBound=1, cat="Integer")
c5 = LpVariable("Computer Simulation", lowBound=0, upBound=1, cat="Integer")
c6 = LpVariable("Introduction to Computer Programming", lowBound=0, upBound=1, cat="Integer")
```

```
c7 = LpVariable("Forecasting", lowBound=0, upBound=1, cat="Integer")

#Objective Function for Minimize Z
prob += c1 + c2 + c3 + c4 + c5 + c6 + c7

#Constraints of Problem
prob += c1 + c2 + c3 + c4 + c7 >= 2, "Math Course Requirements"
prob += c2 + c4 + c5 + c7 >= 2, "Or Course Requirements"
prob += c3 + c5 + c6 >= 2, "Computer Course Requirements"

prob += c4 <= c1, "Calculus is a prerequisite for business statistics"
prob += c3 <= c6, "introduction to computer programming is a prerequisite for computer simulation"
prob += c5 <= c6, "introduction to computer programming is a prerequisite for data structures"
prob += c7 <= c4, "business statistics is a prerequisite for forecasting"

prob.solve()

print("Status:", LpStatus[prob.status])

for v in prob.variables():
    print(v.name, "=", v.varValue)
print("Total Course of the Process = ", prob.objective.value())
```

## ***Output;***

```
Business_Statistics = 0.0
Calculus = 0.0
Computer_Simulation = 1.0
Data_Structures = 1.0
Forecasting = 0.0
Introduction_to_Computer_Programming = 1.0
Operations_Research = 1.0
Total Course of the Process =  4.0
```

## ***Answer 2***

## ***#Decision Variables***

P1W -> Production Line 1 Worker

P2W -> Production Line 2 Worker

P1S -> Production Line 1 Set up

P2S -> Production Line 2 Set up

## ***#Objective Function***

MinZ = 1000*P1S + 2000*P2S + 500*P1W + 900*P2W

## ***#Constraints***

Each week, at least 120 units of glue 1, at least 150 units of glue 2, and at least 200 units of glue 3 must be produced.

20*P1W + 50*P2W >= 120

30*P1W + 35*P2W >= 150

40*P1W + 45*P2W >= 200

Each line can be utilized by up to seven workers at a time

P1W <= 7*P1S

P2W <= 7*P2S

Also

P1W, P2W >= 0

1 >= P1S, P2S >= 0

## ***Formulate an IP to minimize the total cost of meeting weekly demands.***

MinZ = 1000*P1S + 2000*P2S + 500*P1W + 900*P2W

20*P1W + 50*P2W >= 120

30*P1W + 35*P2W >= 150

40*P1W + 45*P2W >= 150

P1W <= 7*P1S

P2W <= 7*P2S

P1W, P2W >= 0

1 >= P1S, P2S >= 0

## ***PULP SOLVE CODE AND OUTPUT***

```
from pulp import LpVariable, LpMinimize, LpStatus, LpProblem


prob = LpProblem("Minimize_Total_Cost_Of_Weekly_Demands_Selection",LpMinimize)


#Decision Variables
p1w = LpVariable("Production Line 1 Worker", lowBound=0, cat="Integer")
p2w = LpVariable("Production Line 2 Worker", lowBound=0, cat="Integer")
p1s = LpVariable("Production Line 1 Set up", lowBound=0, upBound=1, cat="Integer")
p2s = LpVariable("Production Line 2 Set up", lowBound=0, upBound=1, cat="Integer")


#Objective Function for Minimize Z
prob += 1000*p1s + 2000*p2s + 500*p1w + 900*p2w


#Constraints of Problem
prob += 20*p1w + 50*p2w >= 120, "Glue 1 Unit"
prob += 30*p1w + 35*p2w >= 150, "Glue 2 Unit"
prob += 40*p1w + 45*p2w >= 200, "Glue 3 Unit"


prob += p1w <= 7*p1s, "Line 1 Utilization"
prob += p2w <= 7*p2s, "Line 2 Utilization"


prob.solve()
```

```
print("Status:", LpStatus[prob.status])


for v in prob.variables():
    print(v.name, "=", v.varValue)
print("Total Cost of the Process = ", prob.objective.value())
```

***Output;***

```
Production_Line_1_Set_up = 1.0
Production_Line_1_Worker = 6.0
Production_Line_2_Set_up = 0.0
Production_Line_2_Worker = 0.0
Total Cost of the Process =  4000.0
```

## ***Answer 3***

## ***#Decision Variables***

P1 -> Product 1 Ounce

P2 -> Product 2 Ounce

C -> Chemical Ounce

## ***#Objective Function***

MaxZ = P1*(30-P1) + P2*(50-2*P2) - 3*P1 – 5*P2 – 10*C

## ***#Constraints***

-P1 - P2 + C >= 0

17.5 - C >= 0

P1 >= 0

P2 >= 0

C >= 0

## ***#SciPy Solve Code and Output***

```
import numpy as np

from scipy.optimize import minimize


def obj_funx (_list, sign=-1.0):
  p1=_list[0]
  p2=_list[1]
  c=_list[2]


  p1_price=30-p1
  p2_price=50-2*p2


  p1_cost=-3
  p2_cost=-5
  c_cost=-10
```

```
    obj_func = p1*(p1_price) + p2*p2_price + p1*p1_cost + p2*p2_cost + c*c_cost
    return sign*obj_func

def const_1 (_list):
    p1=_list[0]
    p2=_list[1]
    c=_list[2]
    return c - p1 - p2

def const_2 (_list):
    c=_list[2]
    return 17.5 - c

variable_list = np.ones(3)

all_const = ([{'type':'ineq', 'fun':const_1}, {'type':'ineq', 'fun':const_2}])


minimize(obj_funx, variable_list, constraints=all_const, method='SLSQP')
#minimize(obj_funx, variable_list, constraints=all_const, method='trust-constr')
#minimize(obj_funx, variable_list, constraints=all_const, method='COBYLA')
```

## ***#SciPy Solve Code and Output***

```
     fun: -225.3749999995488
     jac: array([-10.00002289, -10.00004578,  10.        ])
 message: 'Optimization terminated successfully.'
    nfev: 26
     nit: 5
    njev: 5
  status: 0
 success: True
       x: array([ 8.4999866 ,  8.74998869, 17.24997529])
```