FINAL COURSE WORK

MWESIGWA JATIUS

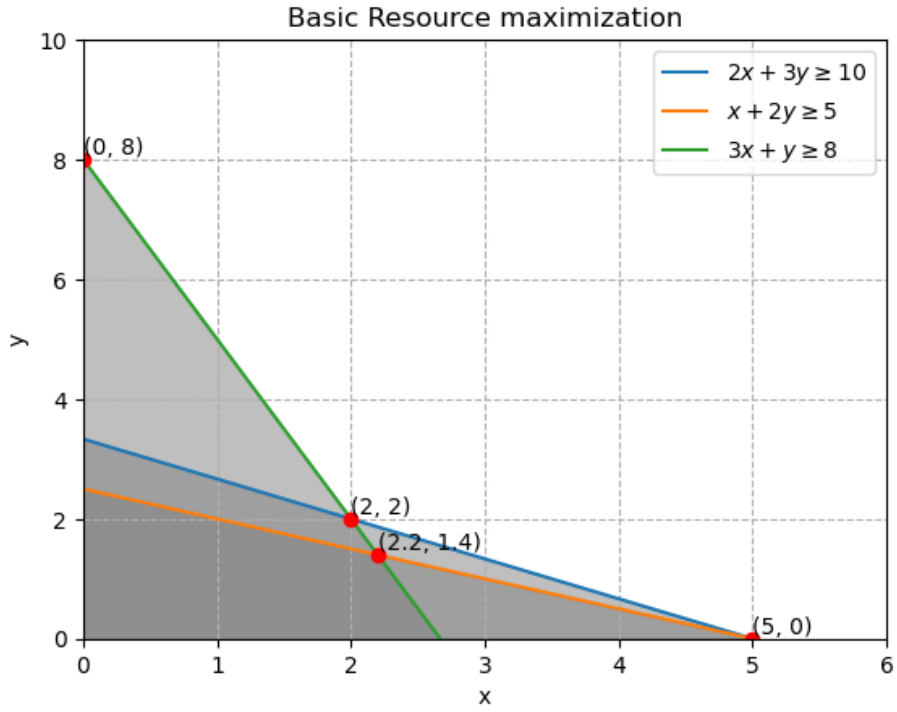
February 2024

1 my whole work

2 number one

```
    #Importing necessary libraries
from pulp import LpProblem, LpMaximize, LpVariable

# Create a linear programming problem
model = LpProblem(name="Basic_resource_allocation", sense=LpMaximize)

#Define Decision Variables
x = LpVariable(name="x", lowBound=0) # Quantity of product A
y = LpVariable(name="y", lowBound=0) # Quantity of product B

# Define the objective function
model += 4 * x + 5 * y, "Objective"

# Define constraints
model += 2 * x + 3 * y >= 10, "CPU_Constraint"
model += x + 2 * y >= 5, "Memory_Constraint"
model += 3 * x + y >= 8, "Storage_Constraint"

# Solve the linear programming problem
model.solve()

# Display the results
print("Optimal Solution:")
print(f"Quantity of Product A (x): {x.varValue}")
print(f"Quantity of product B (y): {y.varValue}")
print(f"Maximum Profit (Z): {model.objective.value()}")
Optimal Solution:

Quantity of Product A (x): 2.0
Quantity of product B (y): 2.0
```

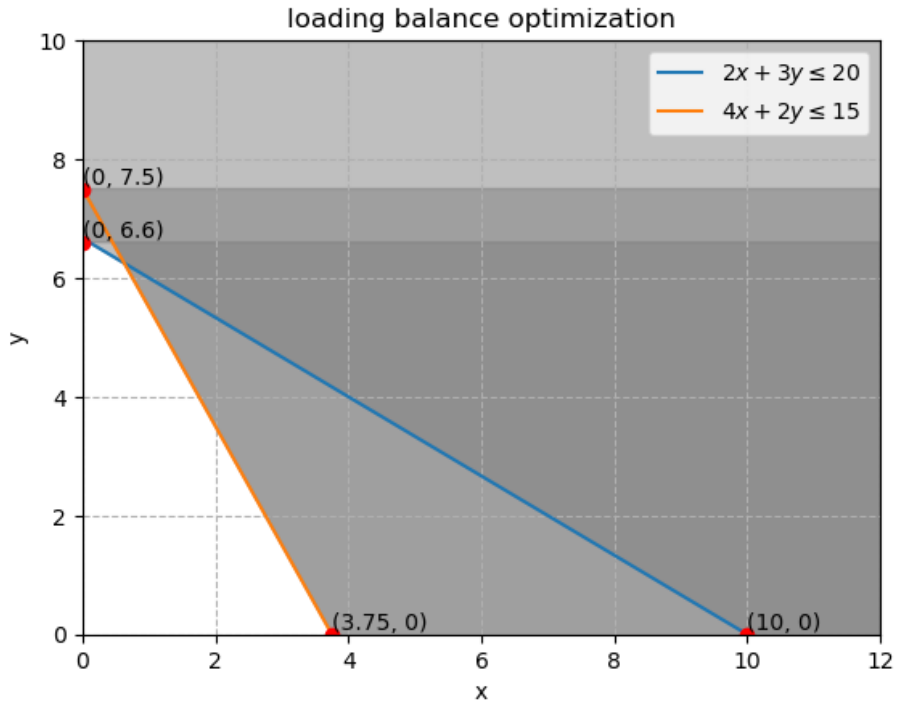
```
Maximum Profit (Z): 18
```



3 number two

```
  #Importing necessary libraries
from pulp import LpProblem, LpMaximize, LpVariable

# Create a linear programming problem
model = LpProblem(name="Loading_Balance_Optmization", sense=LpMaximize)

#Define Decision Variables
x = LpVariable(name="x", lowBound=0) # Quantity of product A
y = LpVariable(name="y", lowBound=0) # Quantity of product B

# Define the objective function
model += 5 * x + 4 * y, "Objective"

# Define constraints
model += 2 * x + 3 * y <= 20, "Server 1_Constraint"
model += 4 * x + 2 * y <= 15, "Server_Constraint"
```

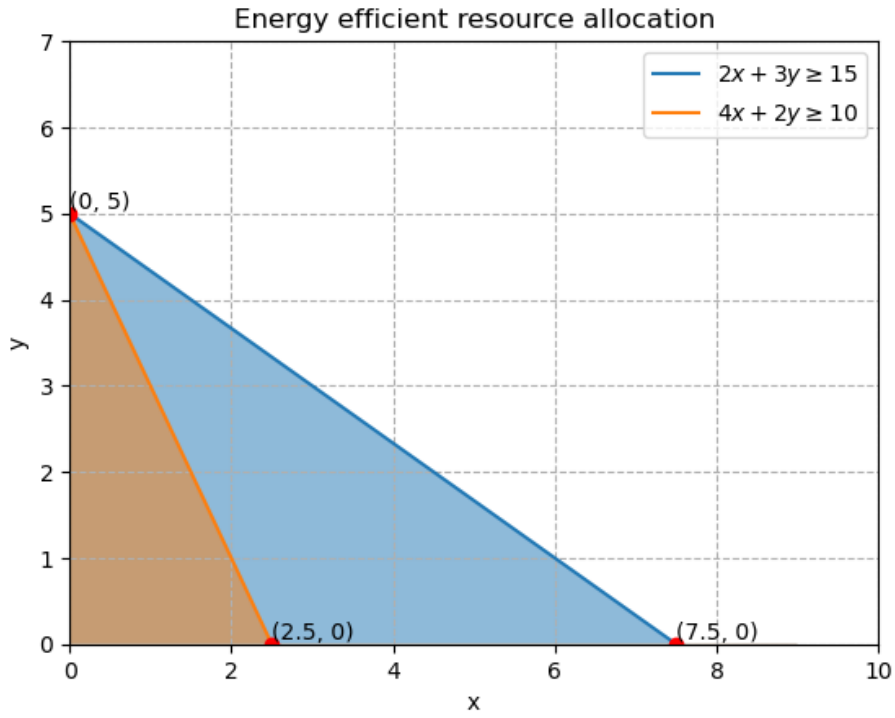
```
# Solve the linear programming problem
model.solve()

# Display the results
print("Optimal Solution:")
print(f"Quantity of Product A (x): {x.varValue}")
print(f"Quantity of product B (y): {y.varValue}")
print(f"Maximum Profit (Z): {model.objective.value()}")

Optimal Solution:
Quantity of Product A (x): 0
Quantity of product B (y) : 0
Maximum Profit (Z): 28.125
```



4 number three

```
    #Importing necessary libraries
from pulp import LpProblem, LpMaximize, LpVariable

# Create a linear programming problem
model = LpProblem(name="Energy_efficient_resource_allocation", sense=LpMaximize)
```

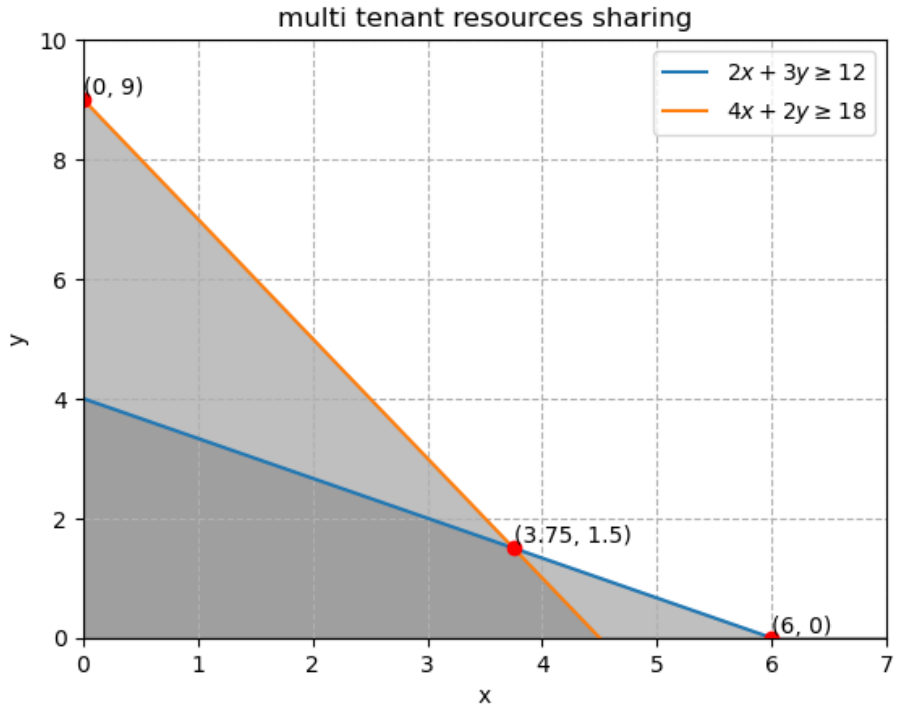
```
#Define Decision Variables
x = LpVariable(name="x", lowBound=0) # Quantity of product A
y = LpVariable(name="y", lowBound=0) # Quantity of product B

# Define the objective function
model += 3 * x + 2 * y, "Objective"

# Define constraints
model += 2 * x + 3 * y >= 15, "CPU allocation_Constraint"
model += 4 * x + 2 * y >= 10, "Memory_Constraint"

# Solve the linear programming problem
model.solve()

# Display the results
print("Optimal Solution:")
print(f"Quantity of Product A (x): {x.varValue}")
print(f"Quantity of product B (y): {y.varValue}")
print(f"Maximum Profit (Z): {model.objective.value()}")

Optimal Solution:
Quantity of Product A (x): 0.0
Quantity of product B (y): 5.0
Manimum Profit (Z): 10.0
```



5 number four

```
    #Importing necessary libraries
from pulp import LpProblem, LpMaximize, LpVariable

# Create a linear programming problem
model = LpProblem(name="Multi_Tenant_Resource_Planning", sense=LpMaximize)

#Define Decision Variables
x = LpVariable(name="x", lowBound=0) # Quantity of product A
y = LpVariable(name="y", lowBound=0) # Quantity of product B

# Define the objective function
model += 5 * x + 4 * y, "Objective"

# Define constraints
model += 2 * x + 3 * y >= 12, "Tenant 1_Constraint"
model += 4 * x + 2 * y >= 18, "Tenant 2_Constraint"

# Solve the linear programming problem
model.solve()
```

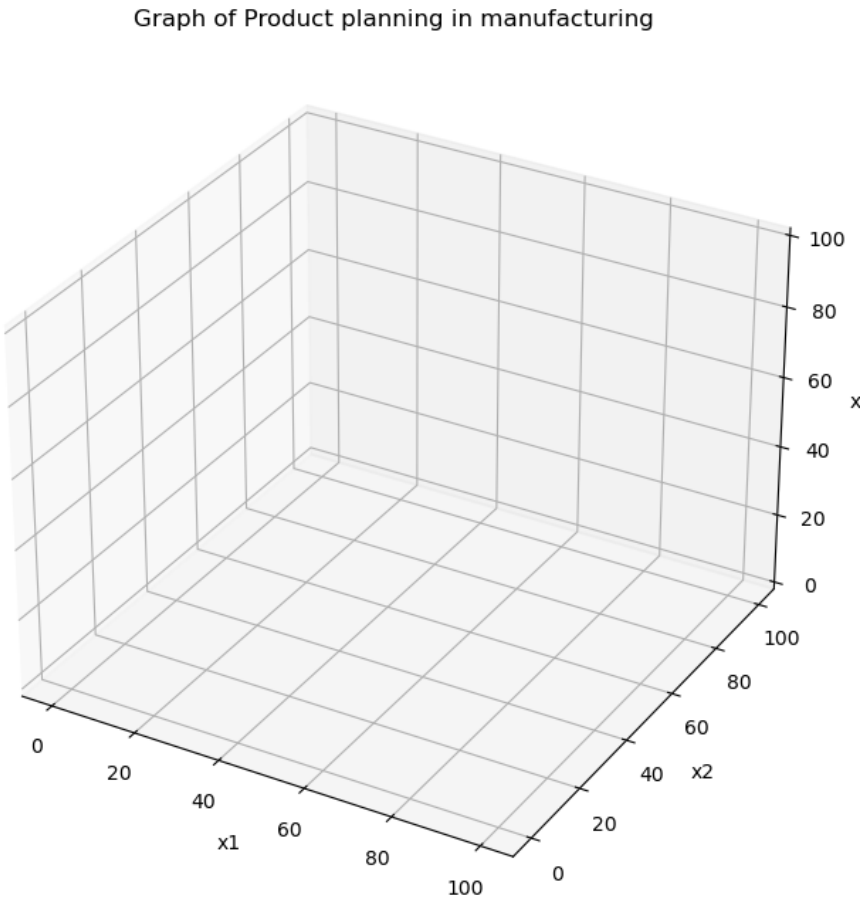
```
# Display the results
print("Optimal Solution:")
print(f"Quantity of Product A (x): {x.varValue}")
print(f"Quantity of product B (y): {y.varValue}")

Optimal Solution:
Quantity of Product A (x): 3.75
Quantity of product B (y): 1.5
Minimum profit 24.5
```



6 number five

```
    #Importing necessary libraries
from pulp import LpProblem, LpMaximize, LpVariable

# Create a linear programming problem
model = LpProblem(name="Product_Planning_in_Manufracturing", sense=LpMaximize)

#Define Decision Variables
x1 = LpVariable(name="x1", lowBound=0) # Quantity of product A
```

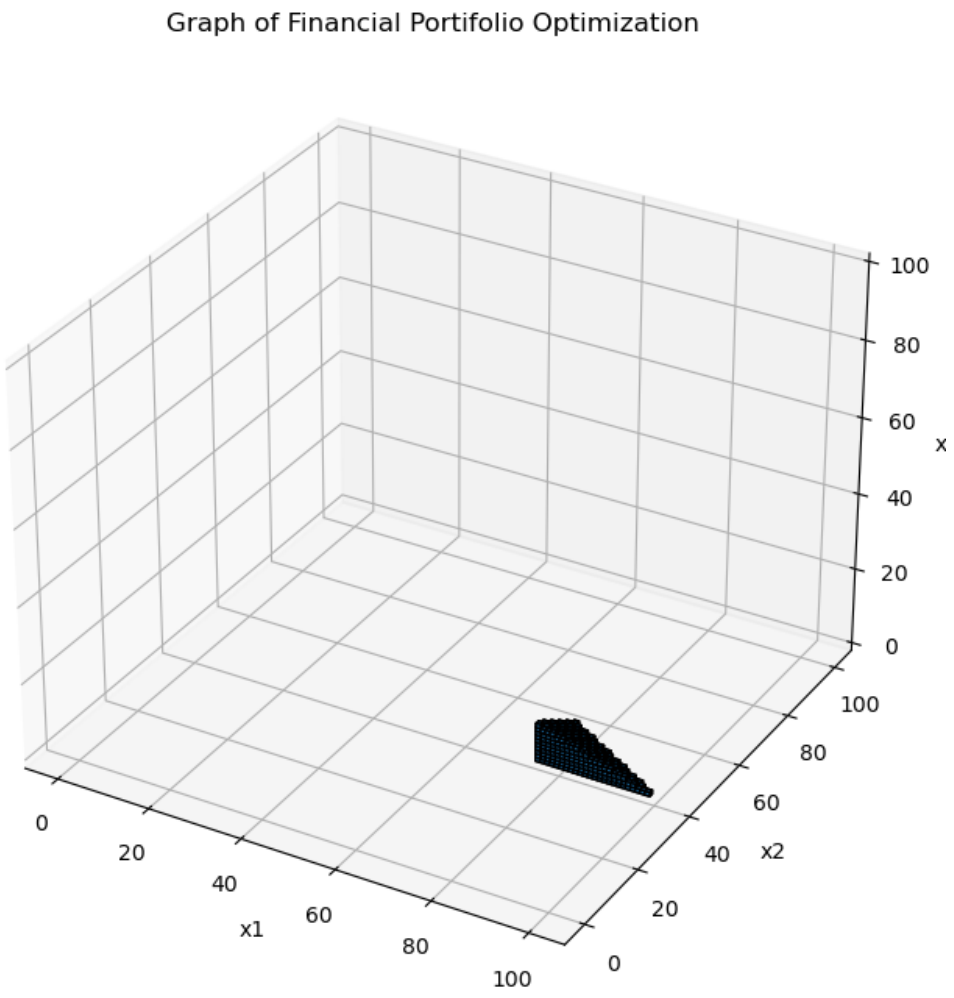
```
x2 = LpVariable(name="x2", lowBound=0) # Quantity of product B
x3 = LpVariable(name="x3", lowBound=0) # Quantity of product C


# Define the objective function
model += 5 * x1 + 3 * x2 + 4 * x3, "Objective"

# Define constraints
model += 2 * x1 + 3 * x2 + x3 <= 1000, "Raw_material_Constraint"
model += 4 * x1 + 2 * x2 + 5 * x3 <= 120, "Labour_hours_Constraint"
model += x1 >= 200, "Demand 1_constraint"
model += x2 >= 300, "Demand 2_constraint"
model += x1 >= 150, "Demand 3_constraint"


# Solve the linear programming problem
model.solve()

# Display the results
print("Optimal Solution:")
print(f"Quantity of Product A (x): {x1.varValue}")
print(f"Quantity of product B (y): {x2.varValue}")
print(f"Quantity of product C (y): {x2.varValue}")
print(f"Maximum Profit (Z): {model.objective.value()}")

Optimal Solution:
Quantity of Product A (x): 200.0
Quantity of product B (y): 200.0
Quantity of product C (y): 200.0
Maximum Profit (Z): 1600.0
```



7 mumber six

```
    #Importing necessary libraries
from pulp import LpProblem, LpMaximize, LpVariable

# Create a linear programming problem
model = LpProblem(name="Financial_Portifolio_Optimization", sense=LpMaximize)

#Define Decision Variables
x1 = LpVariable(name="x1", lowBound=0) # Quantity of product A
x2 = LpVariable(name="x2", lowBound=0) # Quantity of product B
x3 = LpVariable(name="x3", lowBound=0) # Quantity of product C


# Define the objective function
```

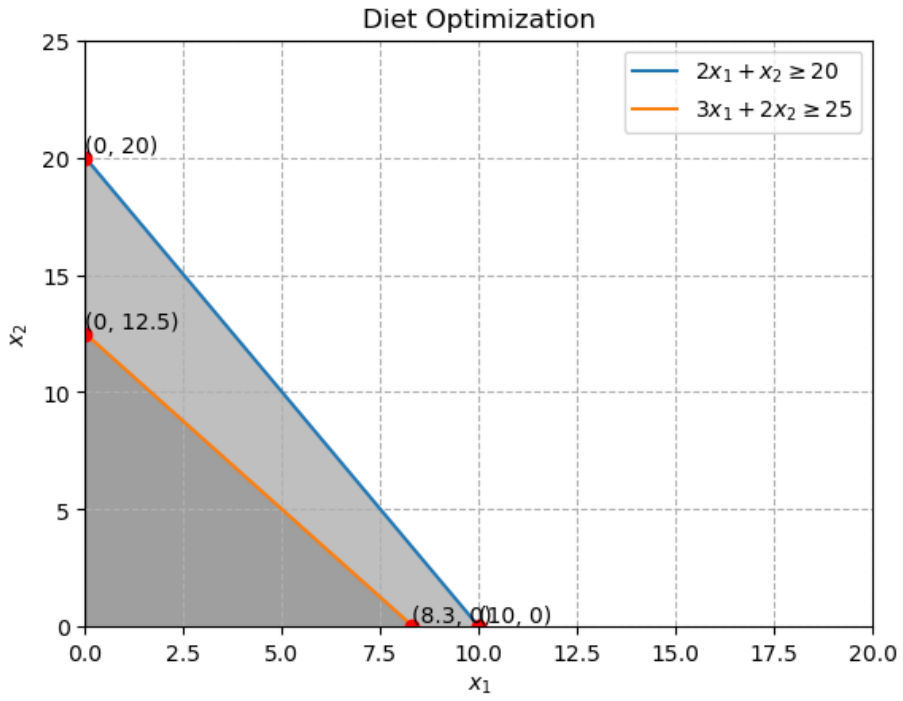
```
model += 0.08 * x1 + 0.1 * x2 + 0.12 * x3, "Objective"

# Define constraints
model += 2 * x1 + 3 * x2 + x3 <= 10000, "Budget_Constraint"
model += x1 >= 2000, "Minimum_investment_constraint 1_constraint"
model += x2 >= 1500, "Minimum_investment_constraint 2_constraint"
model += x3 >= 1000, "Minimum_investment_constraint 3_constraint"

# Solve the linear programming problem
model.solve()

# Display the results
print("Optimal Solution:")
print(f"Quantity of Product A (x): {x1.varValue}")
print(f"Quantity of product B (y): {x2.varValue}")
print(f"Quantity of product C (y): {x2.varValue}")
print(f"Maximum Profit (Z): {model.objective.value()}")

Optimal Solution:
Quantity of Product A (x): 2000.0
Quantity of product B (y): 1500.0
Quantity of product C (y): 1500.0
Maximum Profit (Z): 490.0
```



8 number seven

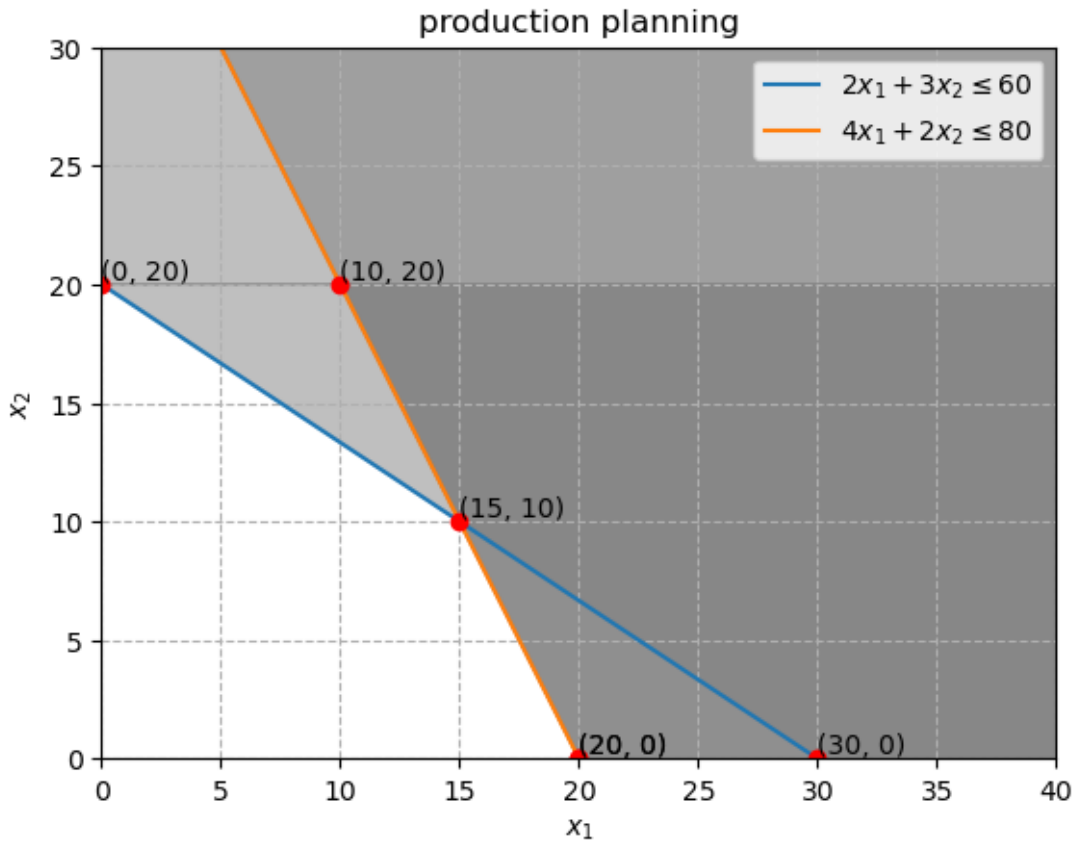
```
    #Importing necessary libraries
from pulp import LpProblem, LpMaximize, LpVariable

# Create a linear programming problem
model = LpProblem(name="Diet_Optimization", sense=LpMaximize)

#Define Decision Variables
x1 = LpVariable(name="x1", lowBound=0) # Quantity of product A
x2 = LpVariable(name="x2", lowBound=0) # Quantity of product B
```

```
# Define the objective function
model += 3 * x1 + 2 * x2, "Objective"

# Define constraints
model += 2 * x1 + x2 >= 20, "Proteins_Nutritional_Requirement_Constraint"
model += 3 * x1 + 2 * x2 >= 25, "Vitamins_Nutritional_requirenent_Constraint"

# Solve the linear programming problem
model.solve()

# Display the results
print("Optimal Solution:")
print(f"Quantity of Product A (x1): {x1.varValue}")
print(f"Quantity of product B (x2): {x2.varValue}")
print(f"Maximum Profit (Z): {model.objective.value()}")

Optimal Solution:
Quantity of Product A (x1): 10.0
Quantity of product B (x2): 0
Maximum Profit (Z): 30
```



9 number eight

```
    # Importing necessay Libraries
from pulp import LpProblem, LpMaximize, LpVariable

# Create a LP minimization problem
model = LpProblem(name="Production_planning", sense=LpMaximize)

#Define Decision Variables
x1 = LpVariable('x1', lowBound=0) # Quantity of Product A
x2 = LpVariable('x2', lowBound=0) # Quantity of Product B

# Define the objective function
model += 4 * x1 + 2 * x2, "Objective"

# Define inequalities as constraints
model += 2 * x1 + 3 * x2 <= 60, "Labour_Resource_Constraint"
model += 4 * x1 + 2 * x2 <= 80, "Raw_Material_Resource_constraint"

# Solve the linear programming problem
model.solve()

# Display the results
print("Optimal Solution for product planning:")
print(f"Quantity of Product A (x1): {x1.varValue}")
print(f"Quantity of Product B (x2): {x2.varValue}")
print(f"Maximum Profit (Z): {model.objective.value()}")

Optimal Solution for product planning:
Quantity of Product A (x1): 20.0
Quantity of Product B (x2): 0.0
Maximum Profit (Z): 80.0
```


production planning
$2x_1 + 3x_2 \leq 60$
$4x_1 + 2x_2 \leq 80$
(0, 20)
(10, 20)
(15, 10)
(20, 0)
(30, 0)
x_1
x_2
0
5
10
15
20
25
30
35
40