MOUNTAINS OF THE MOON UNIVERSITY
FACULTY OF SCIENCE INNOVATION AND TECHNOLOGY
LECTURER: SAMUEL OCEN
STUDENT: AYEBALE PETER
REG: 2023.u.mmu.bcs.00109
COURSEUNIT: LINEAR PROGRAMMING

Peter Ayebale

February 2024

1 SOLUTION

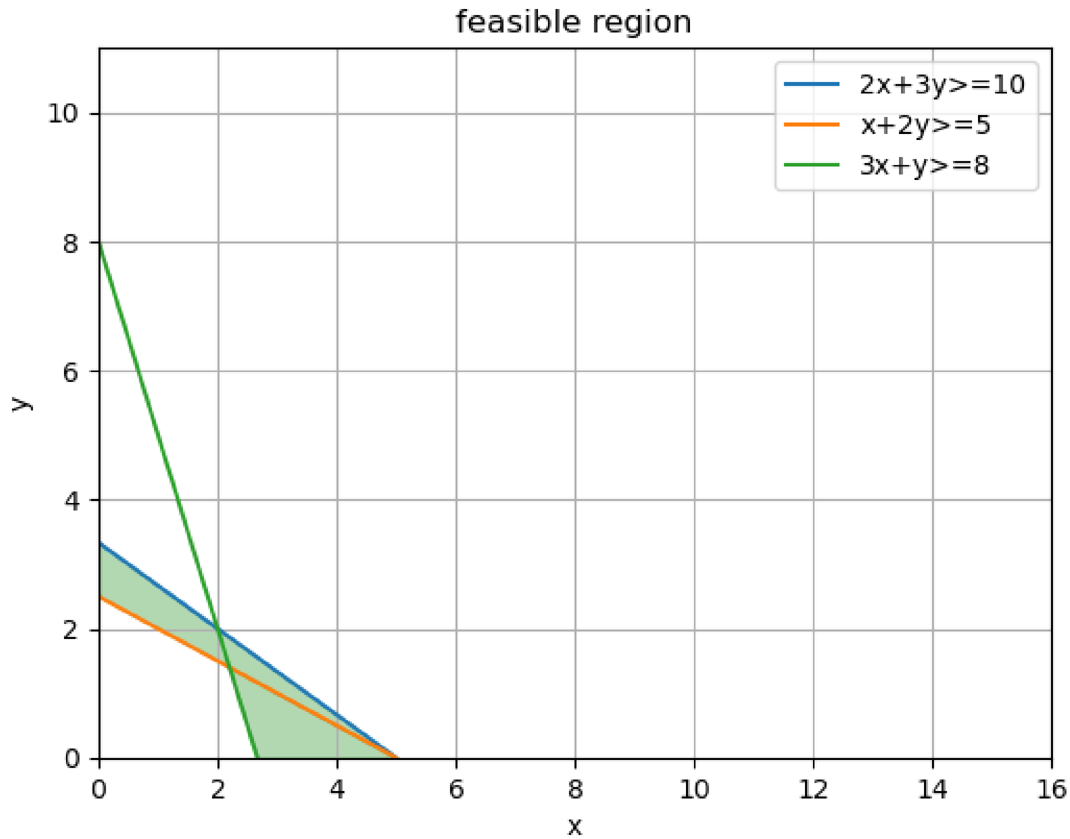
1. minimise the cost of allocating resources(cpu,memory,storage) to virtual machines in a cloud environment.

cpu.$2x + 3y => 10$
memory.$x + 2y => 5$
storage.$3x + y => 8$ MINIMISE.$Z = 4X + 5Y$

```
    #import libraries
from pulp import *
import numpy as np
import matplotlib.pyplot as plt
#define linear problem
obuzibu=LpProblem(name="minimise resourse",sense=LpMinimize)

#define the decision variables
x=LpVariable("X",0)
y=LpVariable("Y",0)

#define the objective
obuzibu+= 4*x + 5*y,"objective"

#define constraints
obuzibu += 2*x + 3*y >=10,"cpu"
```

```
obuzibu += 1*x + 2*y >=5,"memory"
obuzibu += 3*x + 1*y >=8,"storage"

#solve
obuzibu.solve()

#display the results
print("OPTIMUM SOLUTION")
print(f"X={x.varValue}")
print(f"Y={y.varValue}")
print(f"minimum cost={obuzibu.objective.value()}")

#graph
#x array
x=np.linspace(0,5,400)
#coverting constraints to inequalities
y1=(10-2*x)/3
y2=(5-x)/2
y3=(8-3*x)

#plot constraints
plt.plot(x,y1 ,label="2x+3y>=10")
plt.plot(x,y2 ,label="x+2y>=5")
plt.plot(x,y3 ,label="3x+y>=8")

#plot the feasible region
y1=np.maximum(y1,0)
y2=np.maximum(y2,0)
y3=np.maximum(y3,0)

plt.fill_between(x,y1,y2,where=(y1>y2),color='green',alpha=0.3,)
plt.fill_between(x,y2,y3,where=(y2>y3),color='green',alpha=0.3,)

#axis limits and labels
plt.xlim(0,16)
plt.ylim(0,11)
plt.xlabel("x")
plt.ylabel("y")
plt.legend()
plt.title("feasible region")
plt.grid()
plt.show
```

OPTIMUM SOLUTION
X=2.0
Y=2.0
minimum cost=18.0



2. optimise the distribution of work loads across multiple servers to minimise the overall response time.

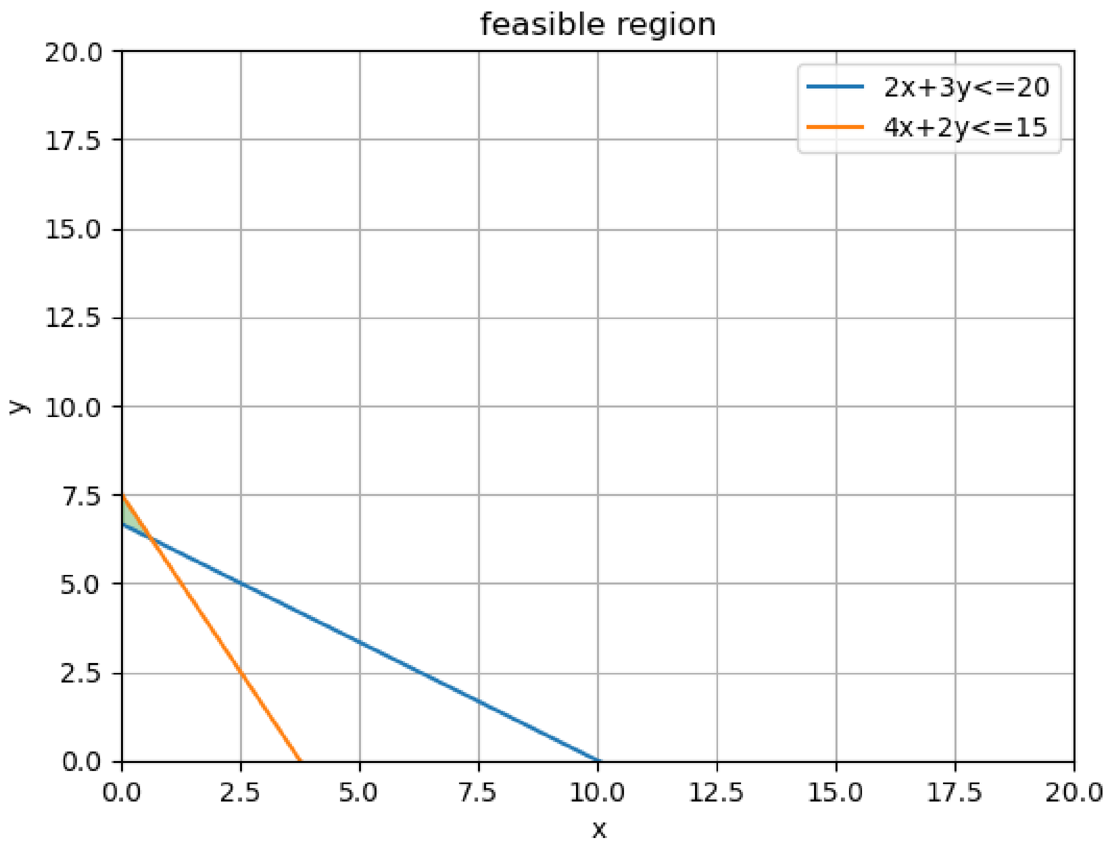
SERVER ONE CAPACITY.$2x + 3y =< 20$
SERVER TWO CAPACITY.$4x + 2y =< 15$
MINIMISE.$Z = 5X + 4Y$

```
    #importing necessary libraries
from pulp import *
import numpy as np
import matplotlib.pyplot as plt


#create a linear programming problem
obuzibu=LpProblem(name="OVERALLRESPONSE TIME", sense=LpMinimize)
```

```
#define decision variables
x = LpVariable(name="x", lowBound=0) #q of p P1
y = LpVariable(name="y", lowBound=0) #q of p P2

#define objective function
obuzibu += 5 * x + 4 * y,"objective"

#define constraints
obuzibu += 2 * x + 3 * y <= 20, "SERVER1CAPACITY"
obuzibu += 4 * x + 2 * y <= 15, "SERVER2CAPACITY"




#solve the linear programming problem
obuzibu.solve()

#display the results
print("optimum solution:")
print(f"X:{x.varValue}")
print(f"Y:{y.varValue}")
print(f"MIN OVERALL RESPONSE TIME (Z):{obuzibu.objective.value()}")

#graph
#x array
x=np.linspace(0,16,2000)
#coverting constraints to inequalities
y1=(20-2*x)/3
y2=(15-4*x)/2

#plot constraints
plt.plot(x,y1 ,label="2x+3y<=20")
plt.plot(x,y2 ,label="4x+2y<=15")

#plot the feasible region
y1=np.maximum(y1,0)
y2=np.maximum(y2,0)

plt.fill_between(x,y1,y2,where=(y2>y1),color='green',alpha=0.3)

#axis limits and labels
plt.xlim(0,20)
plt.ylim(0,20)
plt.xlabel("x")
plt.ylabel("y")
plt.legend()
```

```
plt.title("feasible region")
plt.grid()
plt.show()
```

OPTIMUM SOLUTION
X=0.0
Y=0.0
MIN OVERALL RESPONSE TIME=0.0



3. minimise the total energy consumption of a cloud data center while meeting the resource demands of various applications.

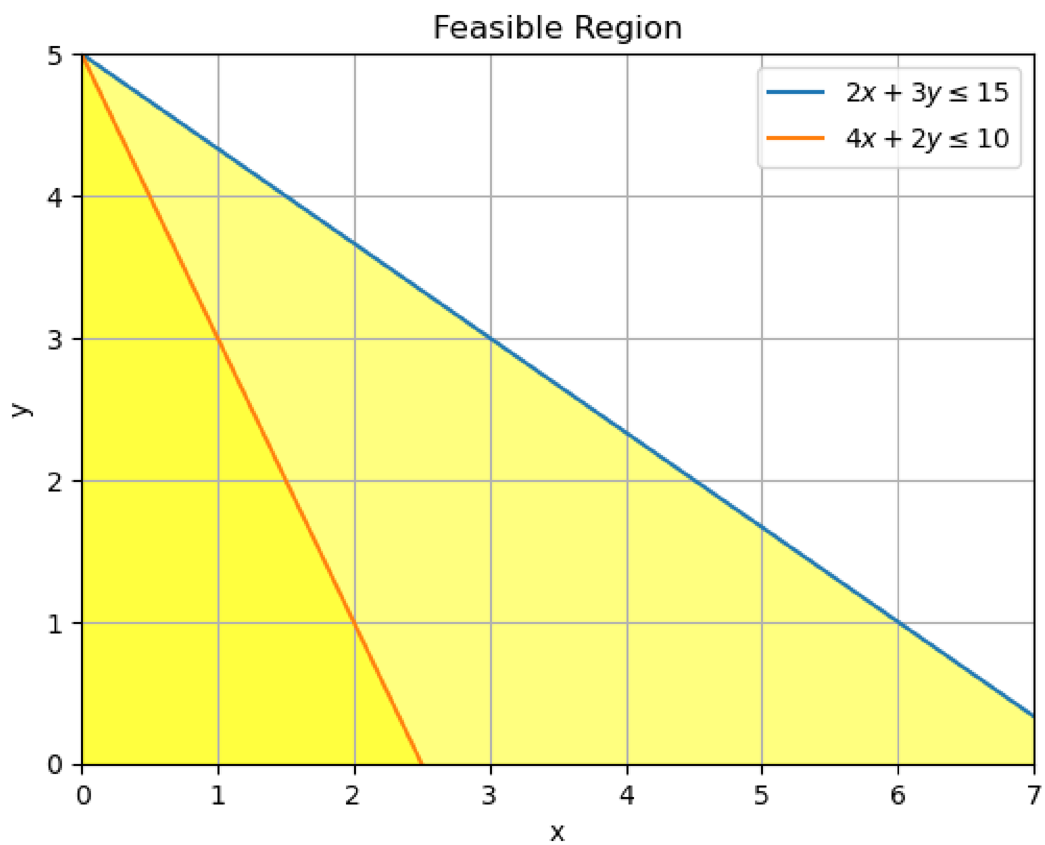
cpu.$2x + 3y => 15$
memory.$4x + 2y => 10$
MINIMISE.$Z = 3X + 2Y$

```
 #importing necessary libraries
from pulp import *
import numpy as np
import matplotlib.pyplot as plt
```

```
#create a linear programming problem
obuzibu=LpProblem(name="ENERGY CO SUMPTION MINIMIZATION", sense=LpMinimize)

#define decision variables
x = LpVariable(name="x", lowBound=0) #q of p P1
y = LpVariable(name="y", lowBound=0) #q of p P2

#define objective function
obuzibu += 3 * x + 2 * y,"objective"

#define constraints
obuzibu += 2 * x + 3 * y >= 15, "CPU"
obuzibu += 4 * x + 2 * y >= 10, "memory"




#solve the linear programming problem
obuzibu.solve()

#display the results
print("optimum solution:")
print(f"X:{x.varValue}")
print(f"Y:{y.varValue}")
print(f"MIN ENERGY CONSUMPTION (Z):{obuzibu.objective.value()}")

#graph
#x array
x=np.linspace(0,16,2000)
#coverting constraints to inequalities
y1=(15-2*x)/3
y2=(10-4*x)/2

#plot constraints
plt.plot(x,y1 ,label="2x+3y>=15")
plt.plot(x,y2 ,label="4x+2y>=10")

#plot the feasible region
y1=np.maximum(y1,0)
y2=np.maximum(y2,0)

plt.fill_between(x,y1,y2,where=(y2<y1),color='yellow',alpha=0.3,label="feasible region")

#axis limits and labels
plt.xlim(0,20)
```

```
plt.ylim(0,20)
plt.xlabel("x")
plt.ylabel("y")
plt.legend()
plt.title("feasible region")
plt.grid()
plt.show()
```

optimum solution
X=0.0
Y=5.0
MIN ENERGY CONSUMPTION=10.0



4. allocate resources among multiple tenants in a multitenant cloud environment, ensuring fairness and meeting service level agreement for each tenant.

tenant1.$2x + 3y => 12$

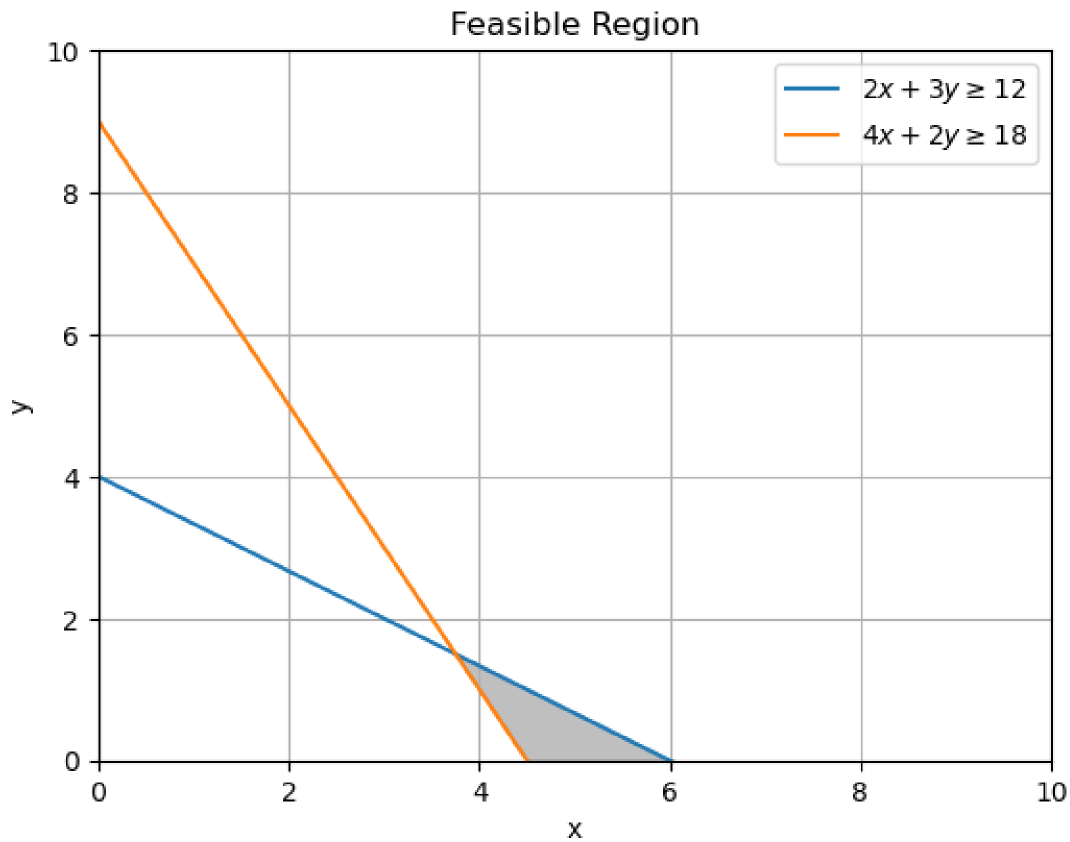
tenant2.$x + 2y => 18$

MINIMISE.$Z = 5X + 4Y$

```
    #importing necessary libraries
from pulp import *

#create a linear programming problem
obuzibu=LpProblem(name="SLA", sense=LpMinimize)

#define decision variables
x = LpVariable(name="x", lowBound=0) #q of p P1
y = LpVariable(name="y", lowBound=0) #q of p P2

#define objective function
obuzibu += 5 * x + 4 * y,"objective"

#define constraints
obuzibu += 2 * x + 3 * y >= 12, "TENANTONE"
obuzibu += 4 * x + 2 * y >= 18, "TENANTTWO"




#solve the linear programming problem
obuzibu.solve()

#display the results
print("optimum solution:")
print(f"X:{x.varValue}")
print(f"Y:{y.varValue}")
print(f"MIN SLA (Z):{obuzibu.objective.value()}")

#graph
#x array
x=np.linspace(0,16,2000)
#coverting constraints to inequalities
y1=(12-2*x)/3
y2=(18-4*x)/2

#plot constraints
plt.plot(x,y1 ,label="2x+3y>=12")
plt.plot(x,y2 ,label="4x+2y>=18")

#plot the feasible region
y3=np.minimum.reduce([y1,y2]) #upper boundary of the feasible region

plt.fill_between(x,y3,0,color='gray',alpha=0.3,label="feasible region")
```

```
#axis limits and labels
plt.xlim(0,16)
plt.ylim(0,11)
plt.xlabel("x")
plt.ylabel("y")
plt.legend()
plt.title("feasible region")
plt.grid()
plt.show()
```

optimum solution
X=3.75
Y=1.5
MIN SLA (Z) =24.75

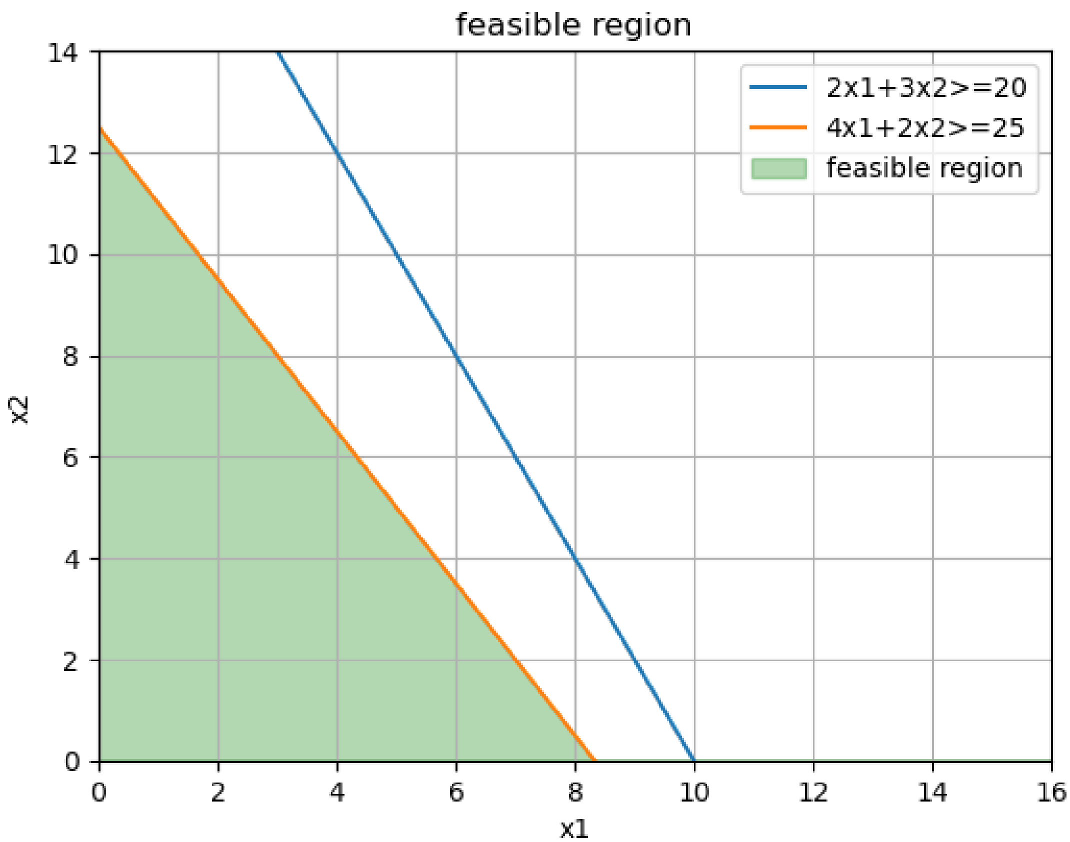


7. minimise the cost of a daily diet while meeting nutritional requirements.
proteins.$2x1 + x2 => 20$
vitamins.$3x1 + 2x2 => 25$
MINIMISE.$Z = 3x1 + 2x2$

```
    #importing necessary libraries
from pulp import *

#create a linear programming problem
obuzibu=LpProblem(name="DAILY DIET", sense=LpMinimize)

#define decision variables
x1 = LpVariable(name="x1", lowBound=0) #q of p P1
x2 = LpVariable(name="x2", lowBound=0) #q of p P2

#define objective function
obuzibu += 3 * x1 + 2 * x2,"objective"

#define constraints
```

```
obuzibu += 2 * x1 +1  * x2 >= 20, "proteins"
obuzibu +=  3 * x1 + 2 * x2 >= 25, "vitamins"




#solve the linear programming problem
obuzibu.solve()

#display the results
print("optimum solution:")
print(f"X:{x1.varValue}")
print(f"Y:{x2.varValue}")
print(f"diet minimization (Z):{obuzibu.objective.value()}")

#graph
#x array
x=np.linspace(0,16,2000)
#coverting constraints to inequalities
x21=(20-2*x)/1
x22=(25-3*x)/2

#plot constraints
plt.plot(x,x21 ,label="2x1+3x2>=20")
plt.plot(x,x22 ,label="4x1+2x2>=25")

#plot the feasible region
x23=np.minimum.reduce([x21,x22]) #upper boundary of the feasible region

plt.fill_between(x,x23,0,color='green',alpha=0.3,label="feasible region")

#axis limits and labels
plt.xlim(0,16)
plt.ylim(0,14)
plt.xlabel("x1")
plt.ylabel("x2")
plt.legend()
plt.title("feasible region")
plt.grid()
plt.show()
```

optimum solution:
X=10.0
Y=0.0
diet minimization=30.0



8. maximise profit in a production process involving two products.
labor.$2x1 + 3x2 =< 60$
raw material.$4x1 + 2x2 =< 80$
MAXIMISE.$Z = 5x1 + 3x2$

```
    #importing necessary libraries
from pulp import *
import numpy as np
import matplotlib.pyplot as plt

#create a linear programming problem
obuzibu=LpProblem(name="profit max pro", sense=LpMaximize)
```

```
#define decision variables
x1 = LpVariable(name="x1", lowBound=0) #q of p P1
x2 = LpVariable(name="x2", lowBound=0) #q of p P2

#define objective function
obuzibu += 5 * x1 + 3 * x2,"objective"

#define constraints
obuzibu += 2 * x1 + 3 * x2 <= 60, "labor"
obuzibu += 4 * x1 + 2 * x2 <= 80, "raw material"




#solve the linear programming problem
obuzibu.solve()

#display the results
print("optimum solution:")
print(f"X:{x1.varValue}")
print(f"Y:{x2.varValue}")
print(f"profit maximization (Z):{obuzibu.objective.value()}")

#graph
#x array
x1=np.linspace(0,50,2000)
#coverting constraints to inequalities
y1=(60-2*x1)/3
y2=(80-4*x1)/2

#plot constraints
plt.plot(x1,y1 ,label="2x1+3x2<=60")
plt.plot(x1,y2 ,label="4x1+2x2<=80")

#plot the feasible region
y3=np.maximum.reduce([y1,y2])

plt.fill_between(x1,y3,11,color='green',alpha=0.3,label="feasible region")

#axis limits and labels
plt.xlim(0,50)
plt.ylim(0,50)
plt.xlabel("x1")
plt.ylabel("x2")
plt.legend()
plt.title("feasible region")
```

```
plt.grid()
plt.show()
```

optimum solution:
X=15.0
Y=10.0
profit maximization (Z)=105.0