ASSINGMENT 1

Name: varna nemulla

Id: 700744920

Question 1

The following is a list of 10 students ages: ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]

- **Sort the list and find the min and max age**
- **Add the min age and the max age again to the list**
- **Find the median age (one middle item or two middle items divided by two)**
- **Find the average age (sum of all items divided by their number)**
- **Find the range of the ages (max minus min)**

```
import statistics #Importing library called statistics which helps in calculating mathematical data
ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]
ages.sort()# Sorts age list in ascending order by default
print ("Sorted age:", ages) # Displays sorted values
# Minimum
print ("Min:", min(ages)) # Displays min value as we used min() method
# Maximum
print ("Max:", max(ages)) # Displays max value as we used max() method
# Adding again min and max values so we use append() method to insert values to the list
ages.append(min(ages))
ages.append(max(ages))
print ("Added min and max values again:",ages) #Displays the list again with new values
# Median (one middle item or two middle items divided by two, as we imported statistics library it calculates easily and provides the opt)
mdn_age = statistics.median(ages)
print ("Median:", mdn_age
# Average age
average= sum(ages)/len(ages)
```

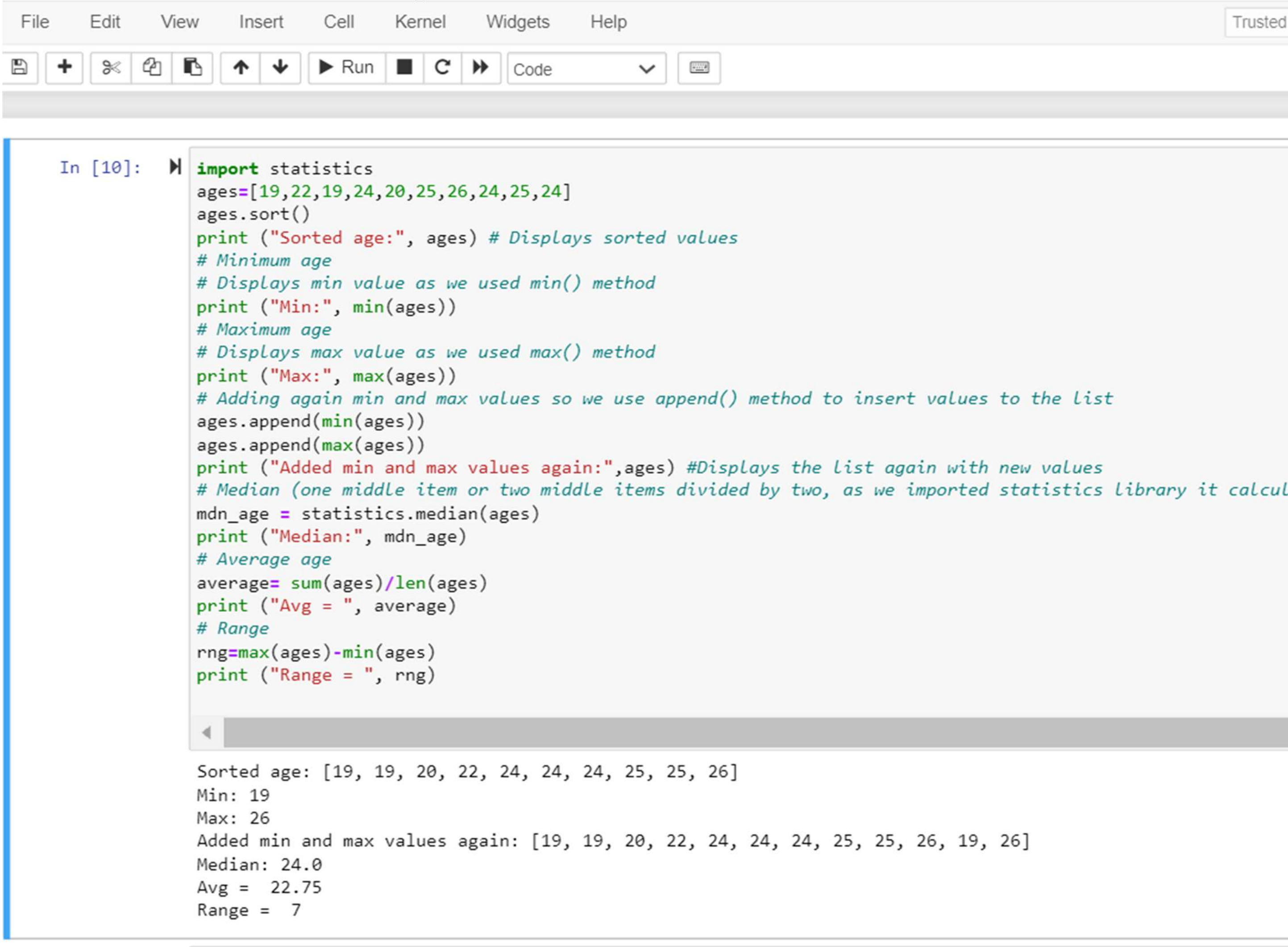
print ("Avg = ", average)

Range

rng=max(ages)-min(ages)

print ("Range = ", rng)

Screenshot of source code and output:



```
import statistics
ages=[19,22,19,24,20,25,26,24,25,24]
ages.sort()
print ("Sorted age:", ages) # Displays sorted values
# Minimum age
# Displays min value as we used min() method
print ("Min:", min(ages))
# Maximum age
# Displays max value as we used max() method
print ("Max:", max(ages))
# Adding again min and max values so we use append() method to insert values to the list
ages.append(min(ages))
ages.append(max(ages))
print ("Added min and max values again:",ages) #Displays the list again with new values
# Median (one middle item or two middle items divided by two, as we imported statistics library it calcul
mdn_age = statistics.median(ages)
print ("Median:", mdn_age)
# Average age
average= sum(ages)/len(ages)
print ("Avg = ", average)
# Range
rng=max(ages)-min(ages)
print ("Range = ", rng)
```

```
Sorted age: [19, 19, 20, 22, 24, 24, 24, 25, 25, 26]
Min: 19
Max: 26
Added min and max values again: [19, 19, 20, 22, 24, 24, 24, 25, 25, 26, 19, 26]
Median: 24.0
Avg =  22.75
Range =  7
```

Description: From the above given list I have found the minimum and maximum ages of students and also added the minimum and maximum age to the list again and also found the median age by dividing it by two and average by adding all the items and then dividing by their number and at last finding the range of the ages.

Question 2

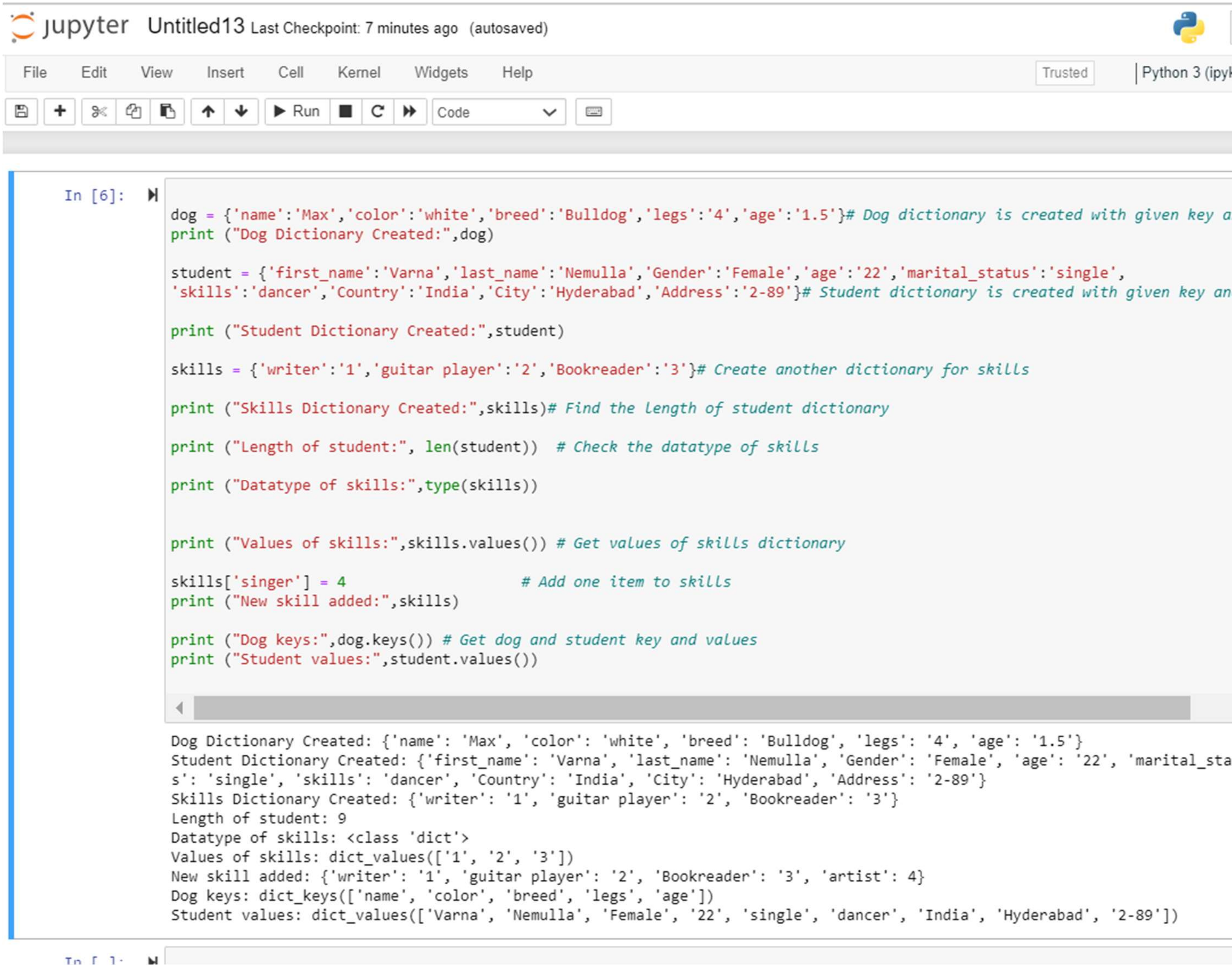
• Create an empty dictionary called dog

• Add name, color, breed, legs, age to the dog dictionary

- **Create a student dictionary and add first_name, last_name, gender, age, marital status, skills, country, city and address as keys for the dictionary**
- **Get the length of the student dictionary**
- **Get the value of skills and check the data type, it should be a list**
- **Modify the skills values by adding one or two skills**
- **Get the dictionary keys as a list**
- **Get the dictionary values as a list**

```
dog = {'name':'Max','color':'white','breed':'Bulldog','legs':'4','age':'1.5'}# Dog dictionary is created with given key and values

print ("Dog Dictionary Created:",dog)

student = {'first_name':'Varna','last_name':'Nemulla','Gender':'Female','age':'22','marital_status':'single',

'skills':'dancer','Country':'India','City':'Hyderabad','Address':'2-89'}# Student dictionary is created with given key and values

print ("Student Dictionary Created:",student)

skills = {'writer':'1','guitar player':'2','Bookreader':'3'}# Create another dictionary for skills

print ("Skills Dictionary Created:",skills)# Find the length of student dictionary

print ("Length of student:", len(student))  # Check the datatype of skills

print ("Datatype of skills:",type(skills))

print ("Values of skills:",skills.values()) # Get values of skills dictionary

skills['singer'] = 4                    # Add one item to skills

print ("New skill added:",skills)

print ("Dog keys:",dog.keys()) # Get dog and student key and values

print ("Student values:",student.values())
```

Screenshot of source code and output:



```
dog = {'name':'Max','color':'white','breed':'Bulldog','legs':'4','age':'1.5'}# Dog dictionary is created with given key a
print ("Dog Dictionary Created:",dog)

student = {'first_name':'Varna','last_name':'Nemulla','Gender':'Female','age':'22','marital_status':'single',
'skills':'dancer','Country':'India','City':'Hyderabad','Address':'2-89'}# Student dictionary is created with given key an

print ("Student Dictionary Created:",student)

skills = {'writer':'1','guitar player':'2','Bookreader':'3'}# Create another dictionary for skills

print ("Skills Dictionary Created:",skills)# Find the length of student dictionary

print ("Length of student:", len(student))  # Check the datatype of skills

print ("Datatype of skills:",type(skills))


print ("Values of skills:",skills.values()) # Get values of skills dictionary

skills['singer'] = 4                    # Add one item to skills
print ("New skill added:",skills)

print ("Dog keys:",dog.keys()) # Get dog and student key and values
print ("Student values:",student.values())
```

```
Dog Dictionary Created: {'name': 'Max', 'color': 'white', 'breed': 'Bulldog', 'legs': '4', 'age': '1.5'}
Student Dictionary Created: {'first_name': 'Varna', 'last_name': 'Nemulla', 'Gender': 'Female', 'age': '22', 'marital_sta
s': 'single', 'skills': 'dancer', 'Country': 'India', 'City': 'Hyderabad', 'Address': '2-89'}
Skills Dictionary Created: {'writer': '1', 'guitar player': '2', 'Bookreader': '3'}
Length of student: 9
Datatype of skills: <class 'dict'>
Values of skills: dict_values(['1', '2', '3'])
New skill added: {'writer': '1', 'guitar player': '2', 'Bookreader': '3', 'artist': 4}
Dog keys: dict_keys(['name', 'color', 'breed', 'legs', 'age'])
Student values: dict_values(['Varna', 'Nemulla', 'Female', '22', 'single', 'dancer', 'India', 'Hyderabad', '2-89'])
```

Description: first I had created an empty dictionary and name it as dog, then added the name , color, breed, legs and age to the dog dictionary. And then I have created a student dictionary and added keys as , country, city, the first_name, last_name, gender, age ,marital status, skills, country, city and address. I found the length of the student dictionary checking the data type and modify the skills values and adding one or two skills and finally getting the dictionary keys and dictionary values as a list.

Question 3

- **Create a tuple containing names of your sisters and your brothers (imaginary siblings are fine)**

- **Join brothers and sisters tuples and assign it to siblings**

- **How many siblings do you have?**

- **Modify the siblings tuple and add the name of your father and mother and assign it to family_members** my_sisters = ('Ashwini', 'Navaneetha','Swarna','Ankitha')

my_brothers = ('Laxman','Gangader','Sai','Ashwanth')   # Create another tuple as siblings and join the sister’s and brother’s tuple

siblings = my_sisters + my_brothers

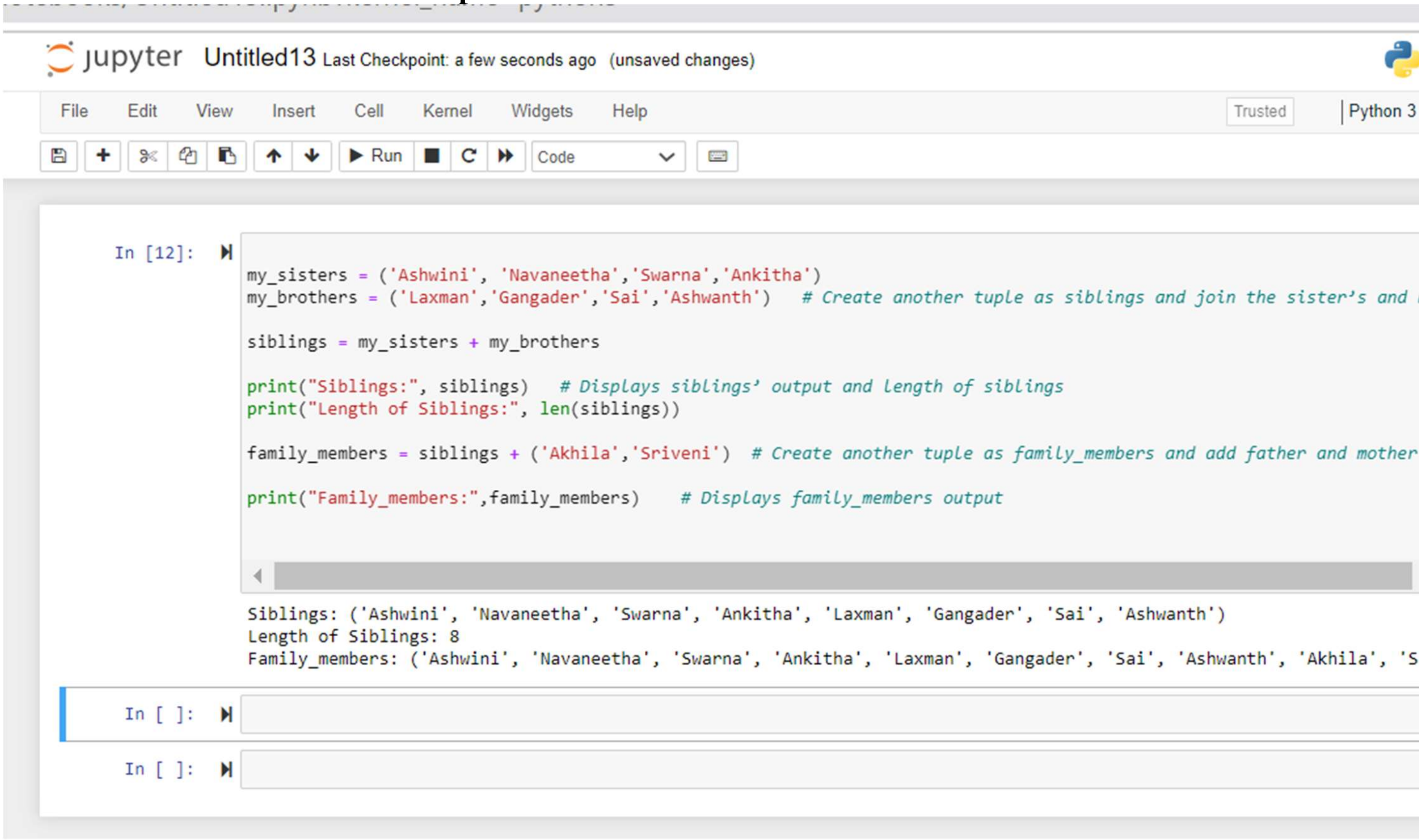
print("Siblings:", siblings)   # Displays siblings’ output and length of siblings

print("Length of Siblings:", len(siblings))

family_members = siblings + ('Akhila','Sriveni') # Create another tuple as family_members and add father and mother name to it

print("Family_members:",family_members)   # Displays family_members output

Screenshot of source code and output:



```
my_sisters = ('Ashwini', 'Navaneetha','Swarna','Ankitha')
my_brothers = ('Laxman','Gangader','Sai','Ashwanth')   # Create another tuple as siblings and join the sister’s and 

siblings = my_sisters + my_brothers

print("Siblings:", siblings)   # Displays siblings’ output and length of siblings
print("Length of Siblings:", len(siblings))

family_members = siblings + ('Akhila','Sriveni')  # Create another tuple as family_members and add father and mother

print("Family_members:",family_members)    # Displays family_members output
```

```
Siblings: ('Ashwini', 'Navaneetha', 'Swarna', 'Ankitha', 'Laxman', 'Gangader', 'Sai', 'Ashwanth')
Length of Siblings: 8
Family_members: ('Ashwini', 'Navaneetha', 'Swarna', 'Ankitha', 'Laxman', 'Gangader', 'Sai', 'Ashwanth', 'Akhila', 'S
```

Description: I have created a tuple and given names of my brothers and sisters and join them and assigned them to siblings. I made modifications for the sibling tuple and added the name of my father and mother and assigned it as family_members.

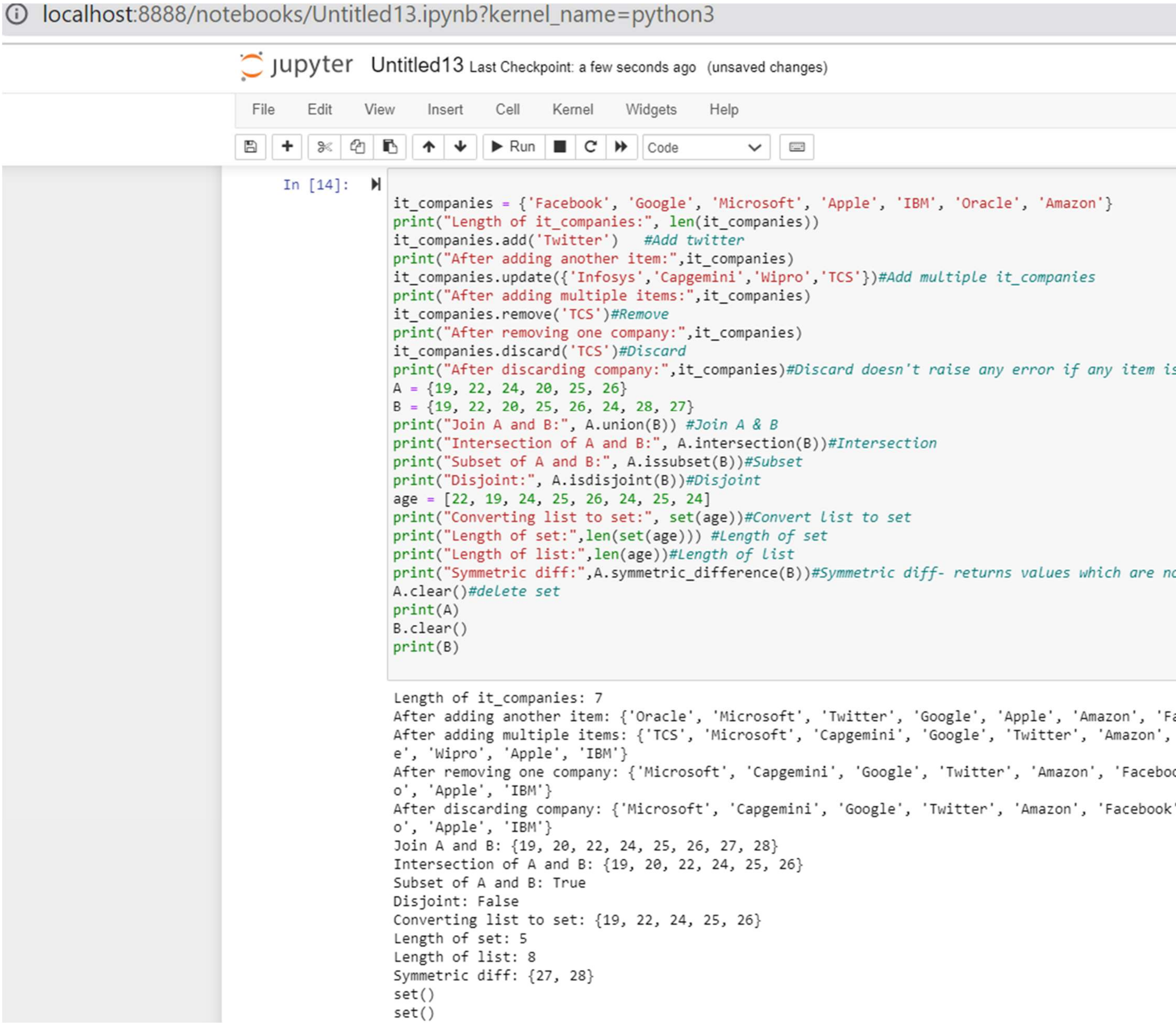
Question 4

it_companies = {'Facebook', 'Google', 'Microsoft', 'Apple', 'IBM', 'Oracle', 'Amazon'} A = {19, 22, 24, 20, 25, 26} B = {19, 22, 20, 25, 26, 24, 28, 27} age = [22, 19, 24, 25, 26, 24, 25, 24] • Find the length of the set it_companies • Add 'Twitter' to it_companies • Insert multiple IT companies at once to the set it_companies • Remove one of the companies from the set it_companies • What is the difference

between remove and discard • Join A and B • Find A intersection B • Is A subset of B • Are A and B disjoint sets • Join A with B and B with A • What is the symmetric difference between A and B • Delete the sets completely • Convert the ages to a set and compare the length of the list and the set.

```
it_companies = {'Facebook', 'Google', 'Microsoft', 'Apple', 'IBM', 'Oracle', 'Amazon'}

print("Length of it_companies:", len(it_companies))

it_companies.add('Twitter')   #Add twitter

print("After adding another item:",it_companies)

it_companies.update({'Infosys','Capgemini','Wipro','TCS'})#Add multiple it_companies

print("After adding multiple items:",it_companies)

it_companies.remove('TCS')#Remove

print("After removing one company:",it_companies)

it_companies.discard('TCS')#Discard

print("After discarding company:",it_companies)#Discard doesn't raise any error if any item is not present in the set

A = {19, 22, 24, 20, 25, 26}

B = {19, 22, 20, 25, 26, 24, 28, 27}

print("Join A and B:", A.union(B)) #Join A & B

print("Intersection of A and B:", A.intersection(B))#Intersection

print("Subset of A and B:", A.issubset(B))#Subset

print("Disjoint:", A.isdisjoint(B))#Disjoint

age = [22, 19, 24, 25, 26, 24, 25, 24]

print("Converting list to set:", set(age))#Convert list to set

print("Length of set:",len(set(age))) #Length of set

print("Length of list:",len(age))#Length of list

print("Symmetric diff:",A.symmetric_difference(B))#Symmetric diff- returns values which are not in common with other set

A.clear()#delete set

print(A)

B.clear()

print(B)
```

Screenshot of source code and output:


localhost:8888/notebooks/Untitled13.ipynb?kernel_name=python3
jupyter
Untitled13 Last Checkpoint: a few seconds ago (unsaved changes)
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Run
Code
In [14]:

```
it_companies = {'Facebook', 'Google', 'Microsoft', 'Apple', 'IBM', 'Oracle', 'Amazon'}
print("Length of it_companies:", len(it_companies))
it_companies.add('Twitter')   #Add twitter
print("After adding another item:",it_companies)
it_companies.update({'Infosys','Capgemini','Wipro','TCS'})#Add multiple it_companies
print("After adding multiple items:",it_companies)
it_companies.remove('TCS')#Remove
print("After removing one company:",it_companies)
it_companies.discard('TCS')#Discard
print("After discarding company:",it_companies)#Discard doesn't raise any error if any item is
A = {19, 22, 24, 20, 25, 26}
B = {19, 22, 20, 25, 26, 24, 28, 27}
print("Join A and B:", A.union(B)) #Join A & B
print("Intersection of A and B:", A.intersection(B))#Intersection
print("Subset of A and B:", A.issubset(B))#Subset
print("Disjoint:", A.isdisjoint(B))#Disjoint
age = [22, 19, 24, 25, 26, 24, 25, 24]
print("Converting list to set:", set(age))#Convert list to set
print("Length of set:",len(set(age))) #Length of set
print("Length of list:",len(age))#Length of list
print("Symmetric diff:",A.symmetric_difference(B))#Symmetric diff- returns values which are no
A.clear()#delete set
print(A)
B.clear()
print(B)
```

```
Length of it_companies: 7
After adding another item: {'Oracle', 'Microsoft', 'Twitter', 'Google', 'Apple', 'Amazon', 'Fa
After adding multiple items: {'TCS', 'Microsoft', 'Capgemini', 'Google', 'Twitter', 'Amazon',
e', 'Wipro', 'Apple', 'IBM'}
After removing one company: {'Microsoft', 'Capgemini', 'Google', 'Twitter', 'Amazon', 'Faceboo
o', 'Apple', 'IBM'}
After discarding company: {'Microsoft', 'Capgemini', 'Google', 'Twitter', 'Amazon', 'Facebook'
o', 'Apple', 'IBM'}
Join A and B: {19, 20, 22, 24, 25, 26, 27, 28}
Intersection of A and B: {19, 20, 22, 24, 25, 26}
Subset of A and B: True
Disjoint: False
Converting list to set: {19, 22, 24, 25, 26}
Length of set: 5
Length of list: 8
Symmetric diff: {27, 28}
set()
set()
```

➔ **Description:** I have found the length of the set it_companies and added twitter to the set it_companies and also inserted multiple IT companies to the set it_companies and differentiated the remove and discard. Then joined the tuples A and B and found the A and B intersection and found that if A subset of B then the looking at if A and B are disjoint sets and the joined the A with B and B with A and finding the symmetric difference between A and B and then deleting the sets completely and converted the ages to a set and compare the length of the list and the set.

Question 5

The radius of a circle is 30 meters. • Calculate the area of a circle and assign the value to a variable name of _area_of_circle_ • Calculate the circumference of a circle and assign the value to a variable name of _circum_of_circle_ • Take radius as user input and calculate the area.

r = int(input("enter r:"))   # Initialise r where r value can be read from user inpt

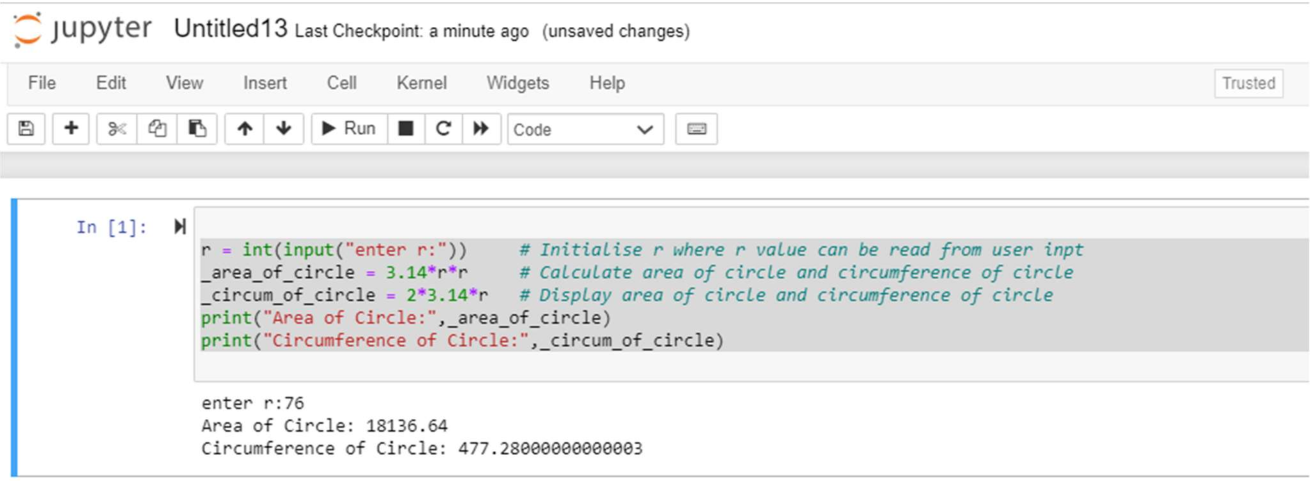
_area_of_circle = 3.14*r*r   # Calculate area of circle and circumference of circle

_circum_of_circle = 2*3.14*r   # Display area of circle and circumference of circle

print("Area of Circle:",_area_of_circle)

print("Circumference of Circle:",_circum_of_circle)

Screenshot of source code and output:



```
r = int(input("enter r:"))      # Initialise r where r value can be read from user inpt
_area_of_circle = 3.14*r*r      # Calculate area of circle and circumference of circle
_circum_of_circle = 2*3.14*r    # Display area of circle and circumference of circle
print("Area of Circle:",_area_of_circle)
print("Circumference of Circle:",_circum_of_circle)
```

```
enter r:76
Area of Circle: 18136.64
Circumference of Circle: 477.28000000000003
```

Description: In the program, we are trying to find the area and circumference of a circle by mentioning the radius

Question 6

“I am a teacher and I love to inspire and teach people” • How many unique words have been used in the sentence? Use the split methods and set to get the unique words.

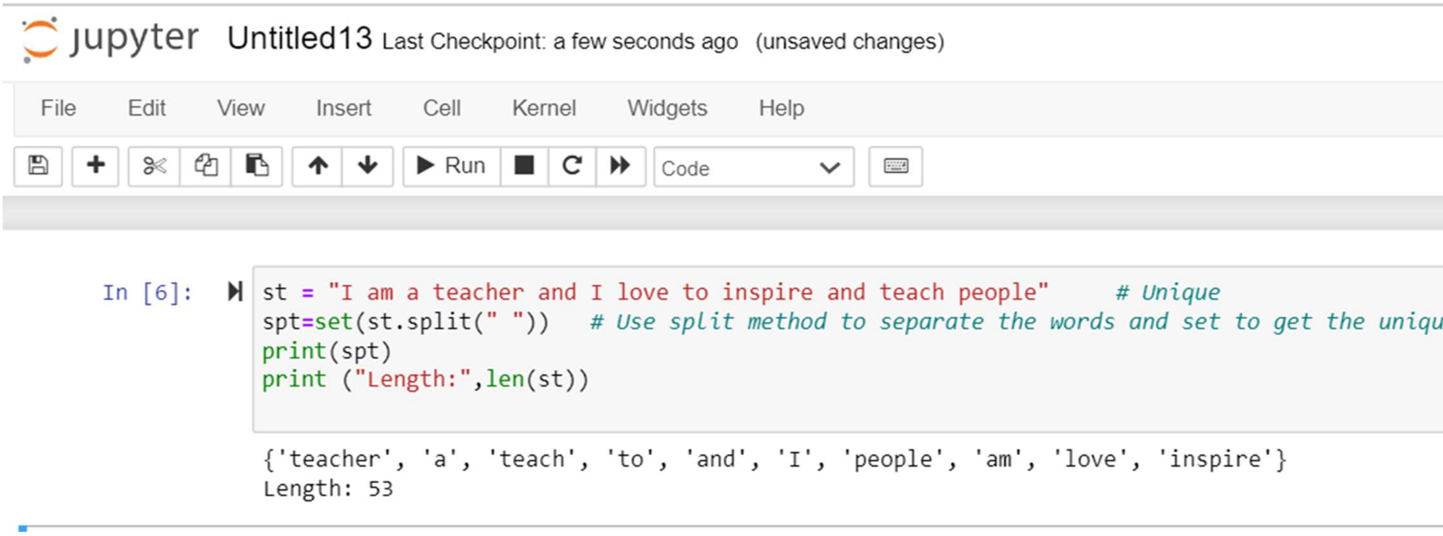
st = "I am a teacher and I love to inspire and teach people"   # Unique

spt=set(st.split(" "))   # Use split method to separate the words and set to get the unique values

print(spt)

print ("Length:",len(st))

Screenshot of source code and output:


jupyter
Untitled13 Last Checkpoint: a few seconds ago (unsaved changes)
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Run
Code
In [6]:

```
st = "I am a teacher and I love to inspire and teach people"     # Unique
spt=set(st.split(" "))   # Use split method to separate the words and set to get the uniqu
print(spt)
print ("Length:",len(st))
```

```
{'teacher', 'a', 'teach', 'to', 'and', 'I', 'people', 'am', 'love', 'inspire'}
Length: 53
```

Description: The above program is to find the count of unique words in a string. I have considered the string "I am a teacher and I love to inspire and teach people" from which the count of unique words is found. Used a set to convert the list into a set.

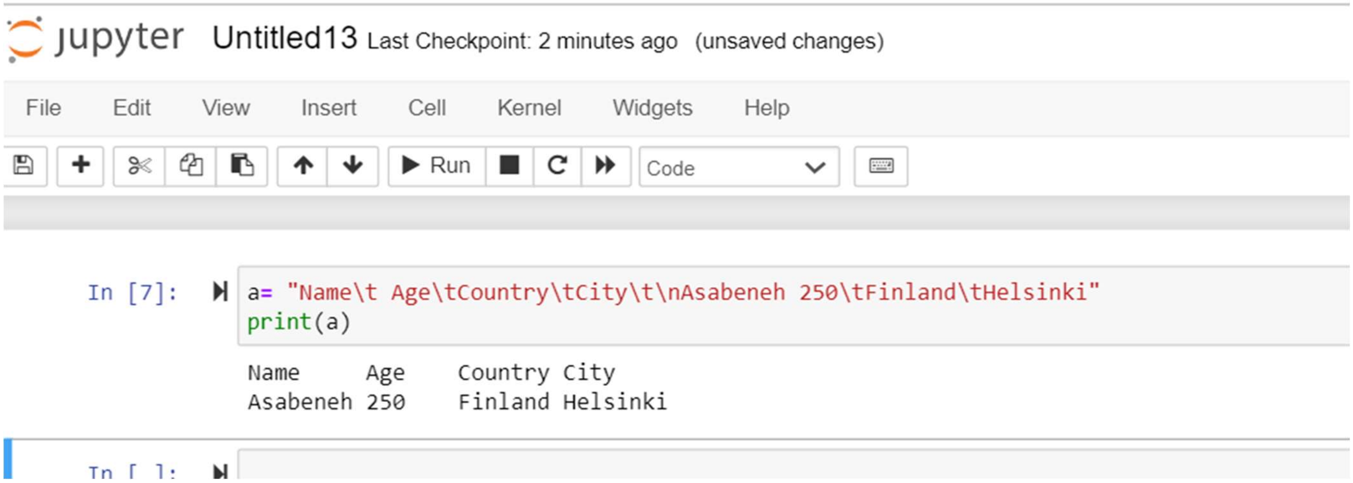
Question 7

Use a tab escape sequence to get the following lines. Name Age Country City Asabeneh 250 Finland Helsinki

a= "Name\t Age\tCountry\tCity\t\nAsabeneh 250\tFinland\tHelsinki"

print(a)

Screenshot of source code and output:



```
a= "Name\t Age\tCountry\tCity\t\nAsabeneh 250\tFinland\tHelsinki"
print(a)
```

```
Name     Age    Country City
Asabeneh 250    Finland Helsinki
```

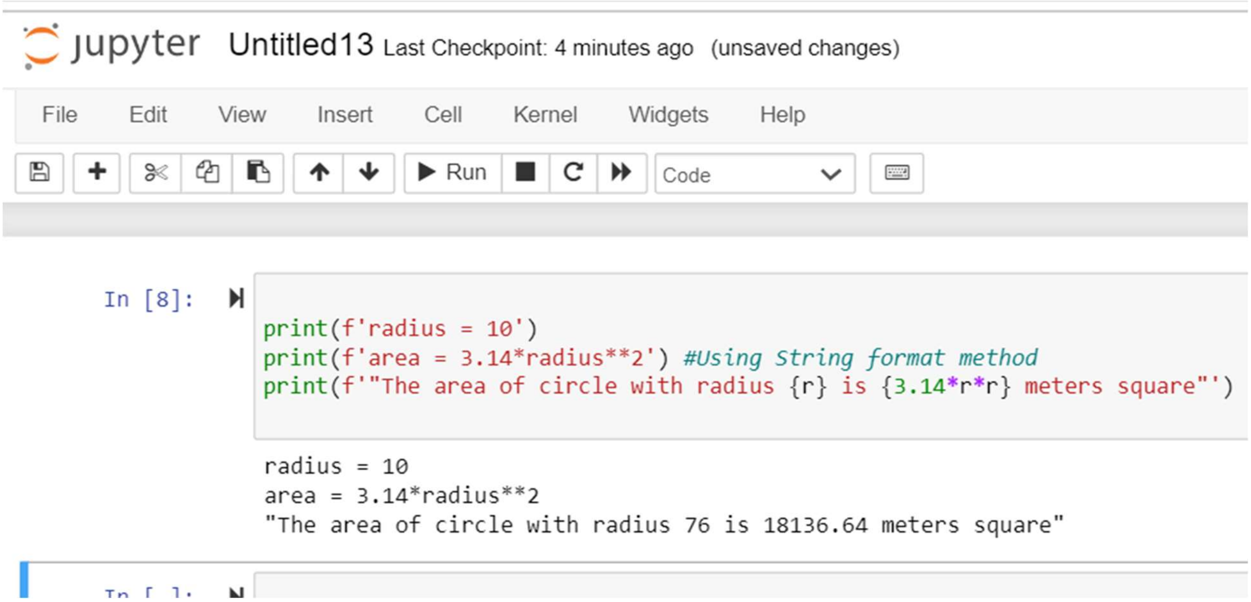
Description: the name, age, country and city of a person is being printed. I have considered the name as asabeneh, age25, country as finland and city is Helsinki.

Question 8

Use the string formatting method to display the following: radius = 10 area = 3.14 * radius ** 2 “The area of a circle with radius 10 is 314 meters square.”

```
#Using String format method

print(f'radius = 10')

print(f'area = 3.14*radius**2')

print(f'"The area of circle with radius {r} is {3.14*r*r} meters square"')
```

Screenshot of source code and output:



```
print(f'radius = 10')
print(f'area = 3.14*radius**2') #Using String format method
print(f'"The area of circle with radius {r} is {3.14*r*r} meters square"')
```

```
radius = 10
area = 3.14*radius**2
"The area of circle with radius 76 is 18136.64 meters square"
```

Description: here we are trying to find the area of a circle with the radius of 10 using the area of circle formula (3.14*r*r).

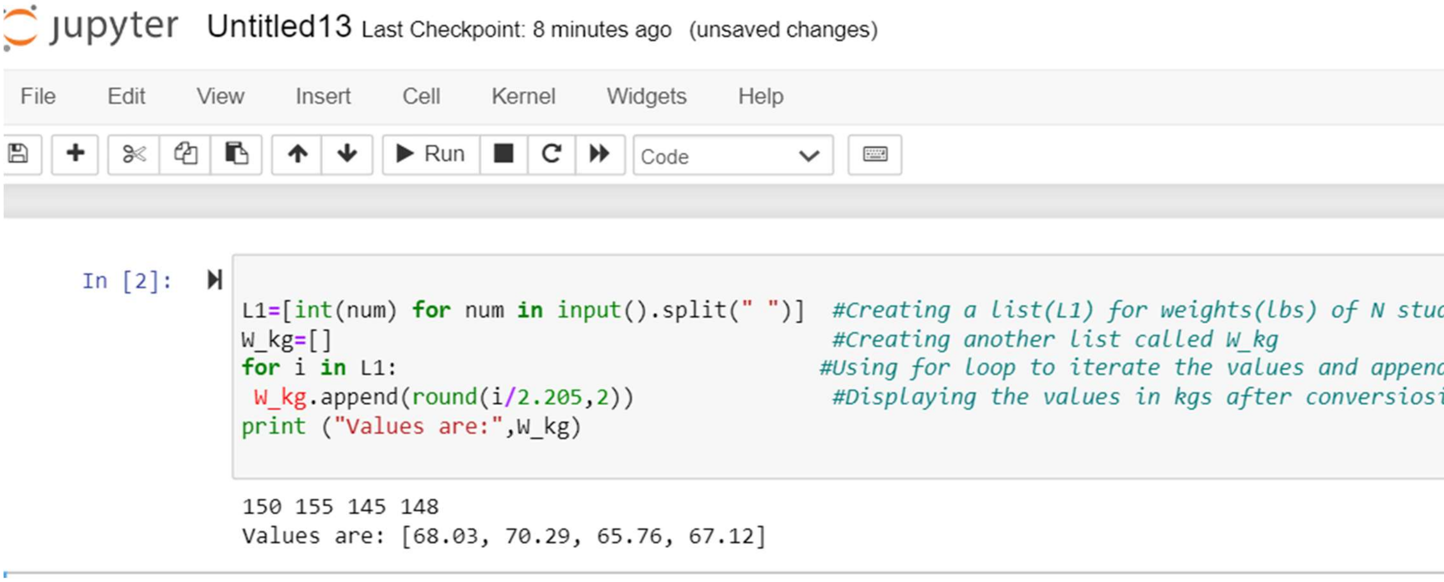
Question 9

Write a program, which reads weights (lbs.) of N students into a list and convert these weights to kilograms in a separate list using Loop. N: No of students (Read input from user) Ex: L1: [150, 155, 145, 148] Output: [68.03, 70.3, 65.77, 67.13]

```
L1=[int(num) for num in input().split(" ")]  #Creating a list(L1) for weights(lbs) of N students

W_kg=[]            #Creating another list called W_kg

for i in L1:   #Using for loop to iterate the values and appending the list

  W_kg.append(round(i/2.205,2))
```

print ("Values are:",W_kg)#Displaying the values in kgs after conversion

Screenshot of source code and output:



```
L1=[int(num) for num in input().split(" ")]  #Creating a list(L1) for weights(lbs) of N stud
W_kg=[]                                      #Creating another list called W_kg
for i in L1:                                #Using for loop to iterate the values and append
 W_kg.append(round(i/2.205,2))               #Displaying the values in kgs after conversiosi
print ("Values are:",W_kg)
```

```
150 155 145 148
Values are: [68.03, 70.29, 65.76, 67.12]
```

Description: the weight of the students converted from Lbs to kgs using python. We have to enter the number of students post which we will have to enter the weight of students in Lbs. after entering the weight, it’ll be converted to kg.

Question 10

The diagram below shows a dataset with 2 classes and 8 data points, each with only one feature value, labeled f. Note that there are two data points with the same feature value of 6. These are shown as two x’s one above the other. Provide stepwise mathematical solution, do not write code for it. 1. Divide this data equally into two parts. Use first part as training and second part as testing. Using KNN classifier, for K=3, what would be the predicted outputs for the test samples? Show how you arrived at your answer. 2. Compute the confusion matrix for this and calculate accuracy, sensitivity and specificity values.

Dataset with total 8 points: \n",

```
[ [ 1  0]
 [ 2  0]
 [ 3  0]
 [ 6  0]
 [ 6  0]
 [ 7  0]
 [10  0]
 [11  0]]
Size:  8
```

Class label for the datasetX Y: [0 0 1 1 1 0 0 0]

Training Data X: \n",

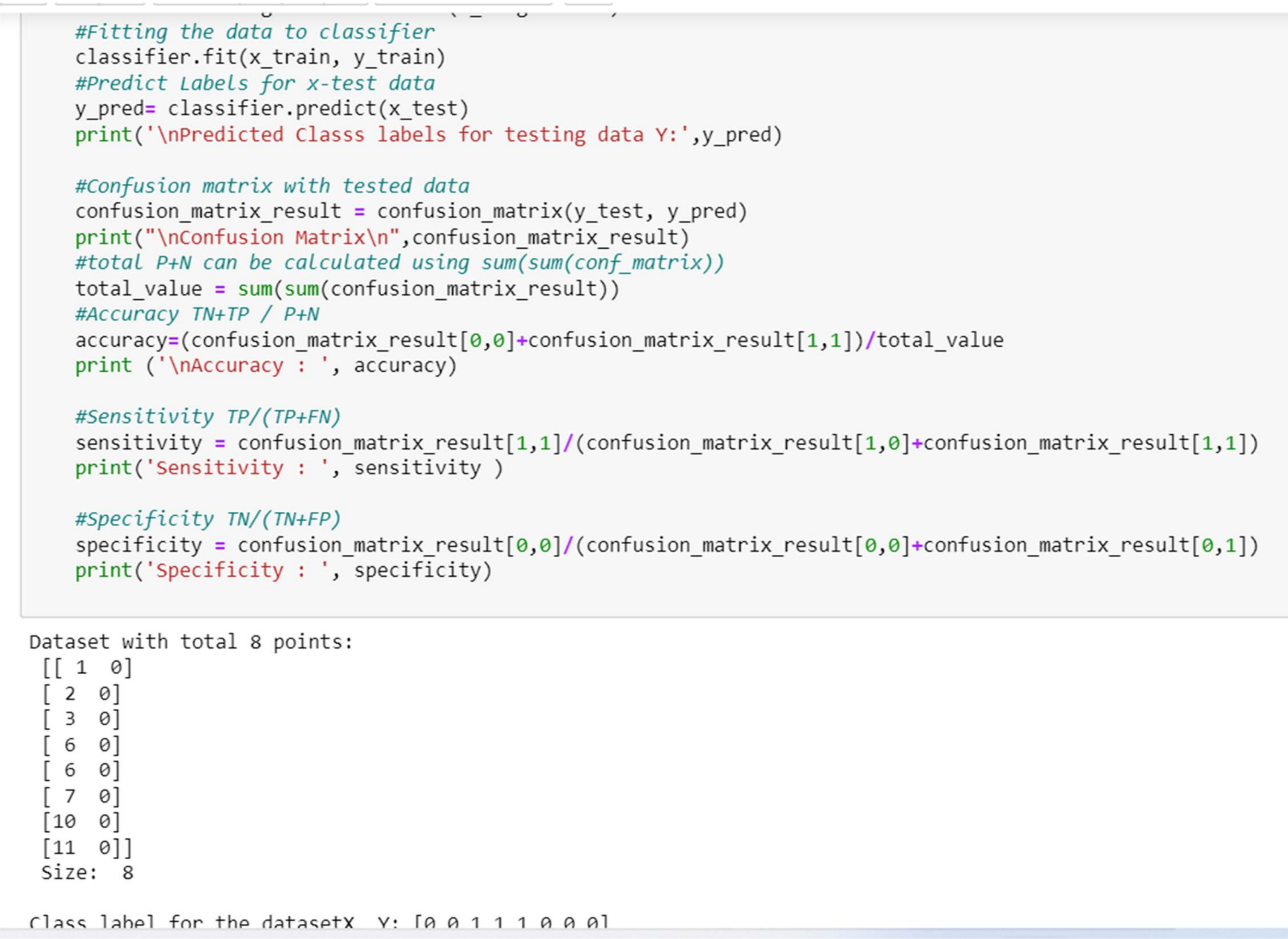
```
[1 0]
[2 0]
[3 0]
```

```
   [6 0]
   Training Data label Y: [0 0 1 1]

   Predicted Classs labels for testing data Y: [1 1 1 1]

   Confusion Matrix\n",
   [[0 3]
   " [0 1]]

   "Accuracy :  0.25
   "Sensitivity :  1.0
   "Specificity :  0.0
  ]
 }
],
source": [
```

```
import numpy as nm

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import confusion_matrix


X= nm.array([[1,0],[2,0],[3,0],[6,0],[6,0],[7,0],[10,0],[11,0]])

print("Dataset with total 8 points: \n",X,"\n Size: ",len(X))

y =nm.array([0,0,1,1,1,0,0,0])

print("\nClass label for the datasetX  Y:",y)

x_train, x_test, y_train, y_test= train_test_split(X, y, test_size= 0.50, random_state=0,shuffle=False )

print("\nTraining Data X: \n", x_train)

print("\nTraining Data label Y:", y_train)

#feature Scaling-used to normalize the range of independent variables or features of data.

#i.e.to standardize the data prior to performing classification

stand_scalar= StandardScaler()

x_train= stand_scalar.fit_transform(x_train)

x_test= stand_scalar.transform(x_test)
```

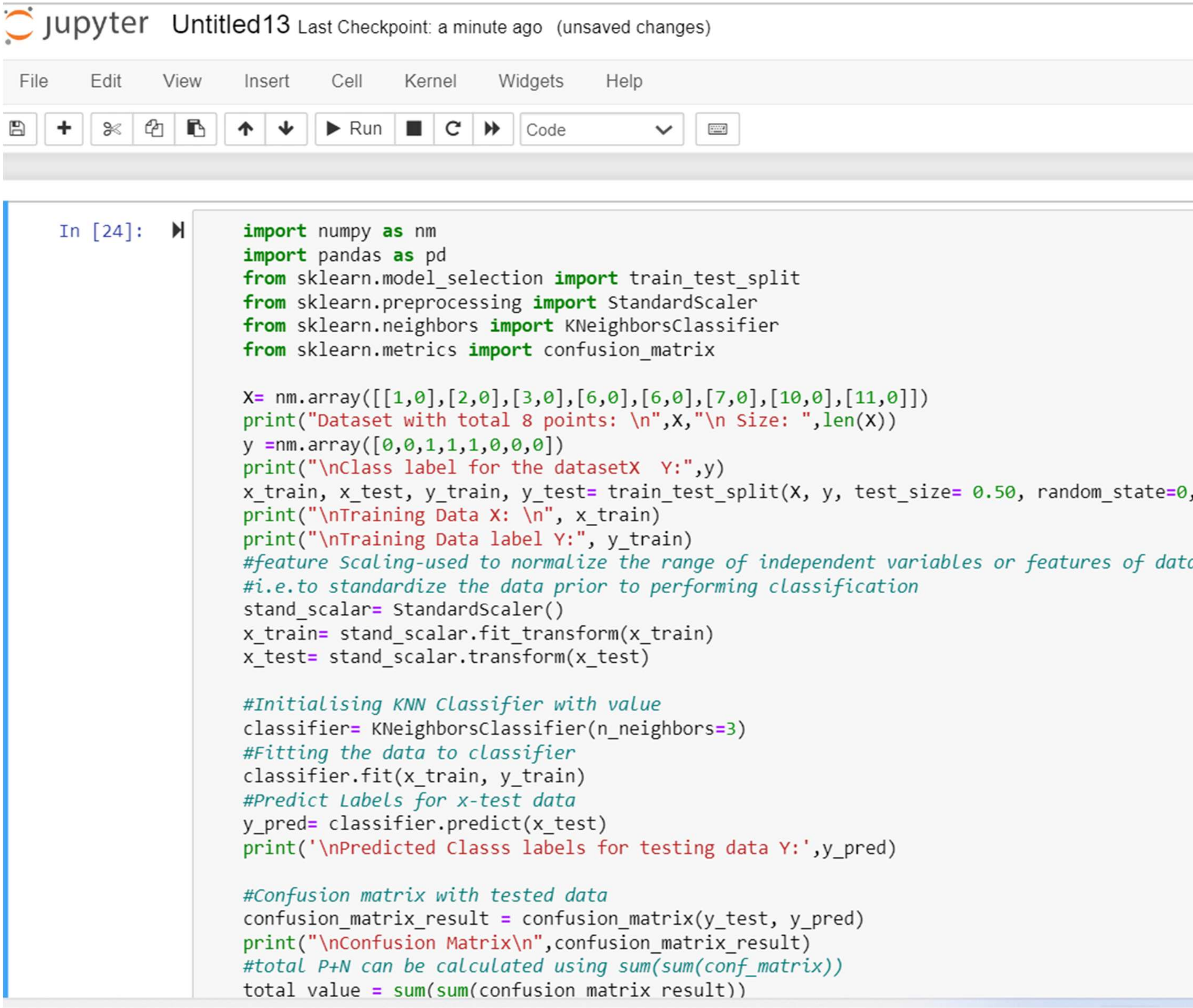
```
   #Initialising KNN Classifier with value
   classifier= KNeighborsClassifier(n_neighbors=3)
   #Fitting the data to classifier
   classifier.fit(x_train, y_train)
   #Predict Labels for x-test data
   y_pred= classifier.predict(x_test)
   print('\nPredicted Classs labels for testing data Y:',y_pred)

   #Confusion matrix with tested data
   confusion_matrix_result = confusion_matrix(y_test, y_pred)
   print("\nConfusion Matrix\n",confusion_matrix_result)
   #total P+N can be calculated using sum(sum(conf_matrix))
   total_value = sum(sum(confusion_matrix_result))
   #Accuracy TN+TP / P+N
   accuracy=(confusion_matrix_result[0,0]+confusion_matrix_result[1,1])/total_value
   print ('\nAccuracy : ', accuracy)
   #Sensitivity TP/(TP+FN)
   sensitivity =
confusion_matrix_result[1,1]/(confusion_matrix_result[1,0]+confusion_matrix_result[1,1])
   print('Sensitivity : ', sensitivity )
   #Specificity TN/(TN+FP)
   specificity =
confusion_matrix_result[0,0]/(confusion_matrix_result[0,0]+confusion_matrix_result[0,1])
   print('Specificity : ', specificity)
```

Screenshot of source code and output:

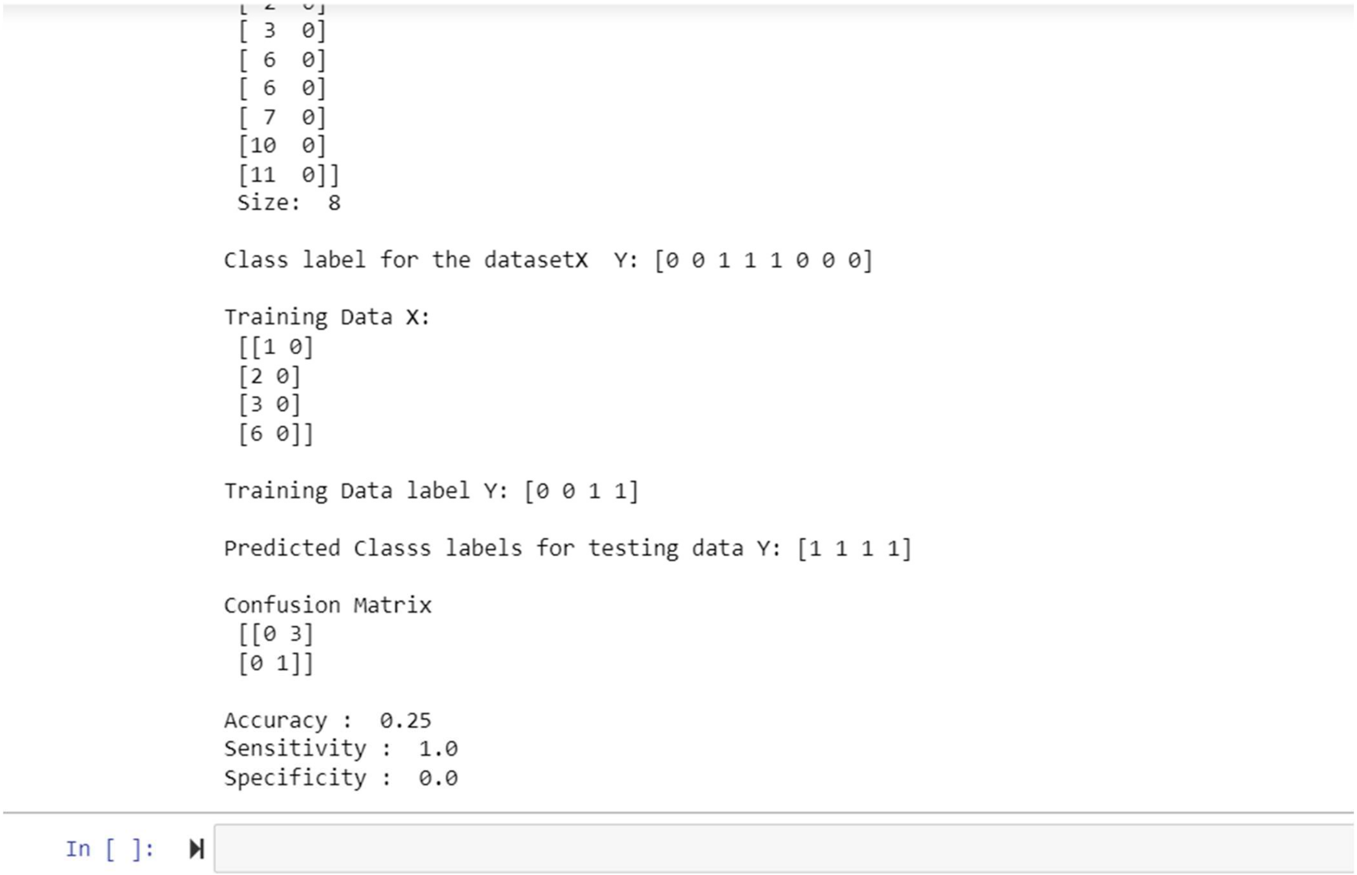


```
#Fitting the data to classifier
classifier.fit(x_train, y_train)
#Predict Labels for x-test data
y_pred= classifier.predict(x_test)
print('\nPredicted Classs labels for testing data Y:',y_pred)

#Confusion matrix with tested data
confusion_matrix_result = confusion_matrix(y_test, y_pred)
print("\nConfusion Matrix\n",confusion_matrix_result)
#total P+N can be calculated using sum(sum(conf_matrix))
total_value = sum(sum(confusion_matrix_result))
#Accuracy TN+TP / P+N
accuracy=(confusion_matrix_result[0,0]+confusion_matrix_result[1,1])/total_value
print ('\nAccuracy : ', accuracy)

#Sensitivity TP/(TP+FN)
sensitivity = confusion_matrix_result[1,1]/(confusion_matrix_result[1,0]+confusion_matrix_result[1,1])
print('Sensitivity : ', sensitivity )

#Specificity TN/(TN+FP)
specificity = confusion_matrix_result[0,0]/(confusion_matrix_result[0,0]+confusion_matrix_result[0,1])
print('Specificity : ', specificity)
```

```
Dataset with total 8 points:
 [[ 1  0]
 [ 2  0]
 [ 3  0]
 [ 6  0]
 [ 6  0]
 [ 7  0]
 [10  0]
 [11  0]]
 Size:  8

Class label for the datasetX  Y: [0 0 1 1 1 0 0 0]
```



```
In [24]:

import numpy as nm
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix

X= nm.array([[1,0],[2,0],[3,0],[6,0],[6,0],[7,0],[10,0],[11,0]])
print("Dataset with total 8 points: \n",X,"\n Size: ",len(X))
y =nm.array([0,0,1,1,1,0,0,0])
print("\nClass label for the datasetX  Y:",y)
x_train, x_test, y_train, y_test= train_test_split(X, y, test_size= 0.50, random_state=0,
print("\nTraining Data X: \n", x_train)
print("\nTraining Data label Y:", y_train)
#feature Scaling-used to normalize the range of independent variables or features of data
#i.e.to standardize the data prior to performing classification
stand_scalar= StandardScaler()
x_train= stand_scalar.fit_transform(x_train)
x_test= stand_scalar.transform(x_test)

#Initialising KNN Classifier with value
classifier= KNeighborsClassifier(n_neighbors=3)
#Fitting the data to classifier
classifier.fit(x_train, y_train)
#Predict Labels for x-test data
y_pred= classifier.predict(x_test)
print('\nPredicted Classs labels for testing data Y:',y_pred)

#Confusion matrix with tested data
confusion_matrix_result = confusion_matrix(y_test, y_pred)
print("\nConfusion Matrix\n",confusion_matrix_result)
#total P+N can be calculated using sum(sum(conf_matrix))
total_value = sum(sum(confusion_matrix_result))
```



```
 [ 2  0]
 [ 3  0]
 [ 6  0]
 [ 6  0]
 [ 7  0]
 [10  0]
 [11  0]]
 Size:  8

Class label for the datasetX  Y: [0 0 1 1 1 0 0 0]

Training Data X:
 [[1 0]
 [2 0]
 [3 0]
 [6 0]]

Training Data label Y: [0 0 1 1]

Predicted Classs labels for testing data Y: [1 1 1 1]

Confusion Matrix
 [[0 3]
 [0 1]]

Accuracy :  0.25
Sensitivity :  1.0
Specificity :  0.0
```

```
In [ ]:
```

Description: Here I took the values as training set and testing from the total dataset points from this I delivered the accuracy and sensitivity, and specificity.

Video Link: https://drive.google.com/file/d/1TgP0esrV264Y5gtRE0bZpptNRlYp4zMK/view?usp=sharing