Machine Learning – Assignment 1

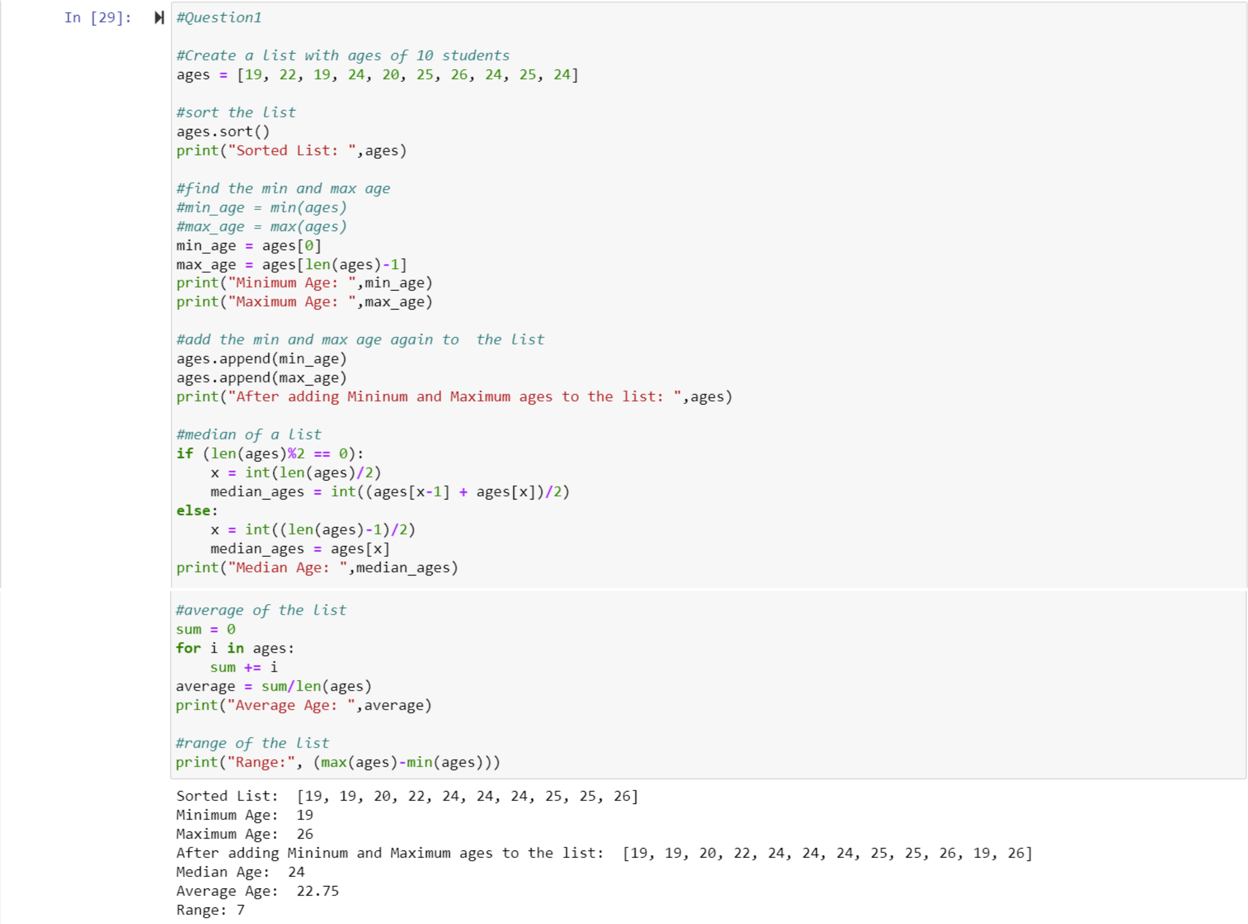
CS 5710 (CRN 22002)

Student ID: 700745451

Student Name: Kamala Ramesh

Question 1:

In this program, we create a list with ages of ten students and perform basic operation in a list such as sort, finding minimum and maximum ages, adding values to the existing list, finding the median, average and range of list values.

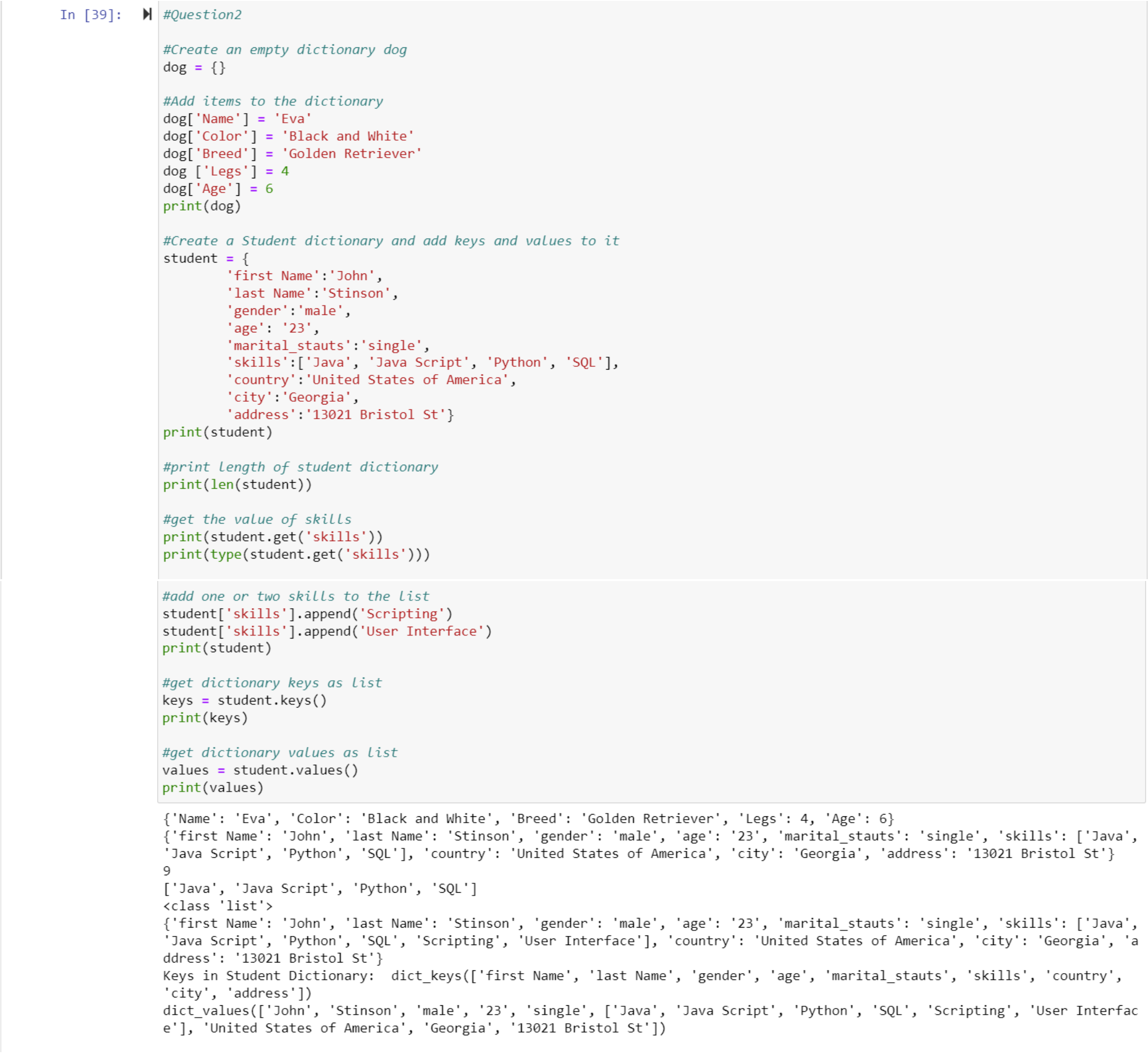


```
In [29]:
#Question1

#Create a list with ages of 10 students
ages = [19, 22, 19, 24, 20, 25, 26, 24, 25, 24]

#sort the list
ages.sort()
print("Sorted List: ",ages)

#find the min and max age
#min_age = min(ages)
#max_age = max(ages)
min_age = ages[0]
max_age = ages[len(ages)-1]
print("Minimum Age: ",min_age)
print("Maximum Age: ",max_age)

#add the min and max age again to  the list
ages.append(min_age)
ages.append(max_age)
print("After adding Mininum and Maximum ages to the list: ",ages)

#median of a list
if (len(ages)%2 == 0):
    x = int(len(ages)/2)
    median_ages = int((ages[x-1] + ages[x])/2)
else:
    x = int((len(ages)-1)/2)
    median_ages = ages[x]
print("Median Age: ",median_ages)

#average of the list
sum = 0
for i in ages:
    sum += i
average = sum/len(ages)
print("Average Age: ",average)

#range of the list
print("Range:", (max(ages)-min(ages)))
```

```
Sorted List:  [19, 19, 20, 22, 24, 24, 24, 25, 25, 26]
Minimum Age:  19
Maximum Age:  26
After adding Mininum and Maximum ages to the list:  [19, 19, 20, 22, 24, 24, 24, 25, 25, 26, 19, 26]
Median Age:  24
Average Age:  22.75
Range: 7
```

Question 2:

In this program, we create an empty dictionary called dog and add items to the dictionary. Then we create another dictionary called students with items added. Also, here we have performed basic operations on a dictionary like finding the length, retrieving values, when one of the dictionary values is a list, appending items to the list, printing all keys of the dictionary and printing all the values of the dictionary.



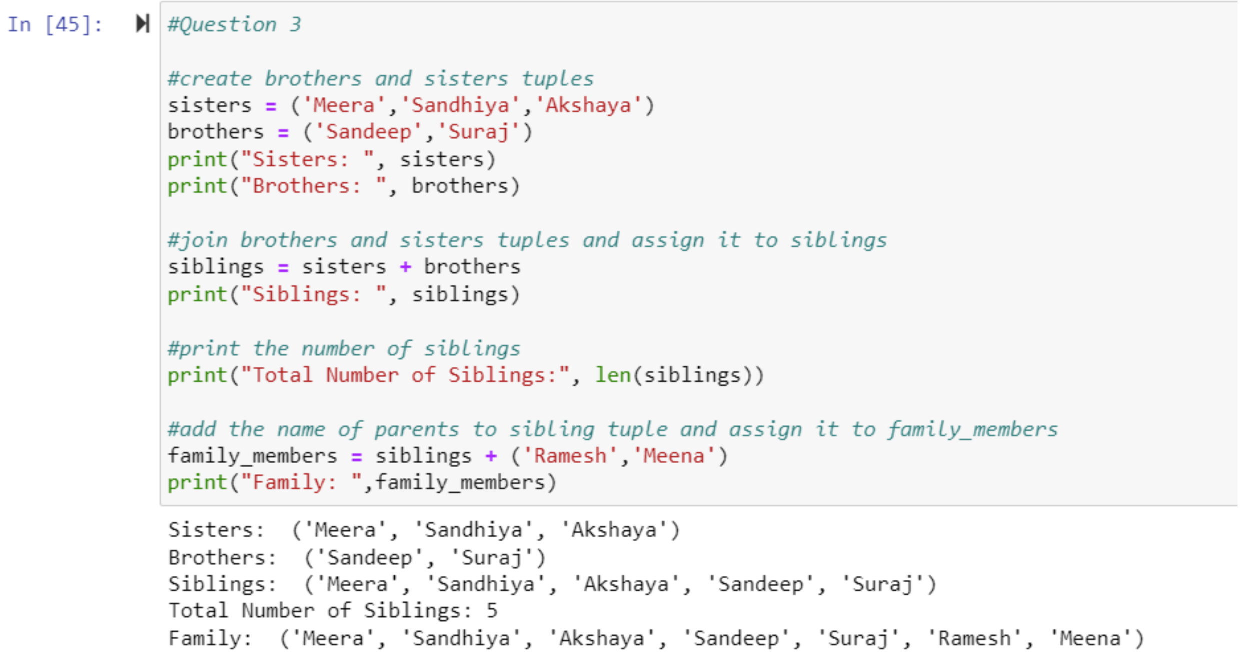
```
#Question2

#Create an empty dictionary dog
dog = {}

#Add items to the dictionary
dog['Name'] = 'Eva'
dog['Color'] = 'Black and White'
dog['Breed'] = 'Golden Retriever'
dog ['Legs'] = 4
dog['Age'] = 6
print(dog)

#Create a Student dictionary and add keys and values to it
student = {
        'first Name':'John',
        'last Name':'Stinson',
        'gender':'male',
        'age': '23',
        'marital_stauts':'single',
        'skills':['Java', 'Java Script', 'Python', 'SQL'],
        'country':'United States of America',
        'city':'Georgia',
        'address':'13021 Bristol St'}
print(student)

#print length of student dictionary
print(len(student))

#get the value of skills
print(student.get('skills'))
print(type(student.get('skills')))
```

```
#add one or two skills to the list
student['skills'].append('Scripting')
student['skills'].append('User Interface')
print(student)

#get dictionary keys as list
keys = student.keys()
print(keys)

#get dictionary values as list
values = student.values()
print(values)
```

```
{'Name': 'Eva', 'Color': 'Black and White', 'Breed': 'Golden Retriever', 'Legs': 4, 'Age': 6}
{'first Name': 'John', 'last Name': 'Stinson', 'gender': 'male', 'age': '23', 'marital_stauts': 'single', 'skills': ['Java',
'Java Script', 'Python', 'SQL'], 'country': 'United States of America', 'city': 'Georgia', 'address': '13021 Bristol St'}
9
['Java', 'Java Script', 'Python', 'SQL']
<class 'list'>
{'first Name': 'John', 'last Name': 'Stinson', 'gender': 'male', 'age': '23', 'marital_stauts': 'single', 'skills': ['Java',
'Java Script', 'Python', 'SQL', 'Scripting', 'User Interface'], 'country': 'United States of America', 'city': 'Georgia', 'a
ddress': '13021 Bristol St'}
Keys in Student Dictionary:  dict_keys(['first Name', 'last Name', 'gender', 'age', 'marital_stauts', 'skills', 'country',
'city', 'address'])
dict_values(['John', 'Stinson', 'male', '23', 'single', ['Java', 'Java Script', 'Python', 'SQL', 'Scripting', 'User Interfac
e'], 'United States of America', 'Georgia', '13021 Bristol St'])
```

Question 3:

This program deals with tuples. Here we create two tuples containing the names of brothers and sisters. We join these two tuples and assign it to a tuple called siblings. Finding the length of the tuple and adding the parents to sibling and assign it to tuple called family_members.



```
In [45]:  #Question 3

          #create brothers and sisters tuples
          sisters = ('Meera','Sandhiya','Akshaya')
          brothers = ('Sandeep','Suraj')
          print("Sisters: ", sisters)
          print("Brothers: ", brothers)

          #join brothers and sisters tuples and assign it to siblings
          siblings = sisters + brothers
          print("Siblings: ", siblings)

          #print the number of siblings
          print("Total Number of Siblings:", len(siblings))

          #add the name of parents to sibling tuple and assign it to family_members
          family_members = siblings + ('Ramesh','Meena')
          print("Family: ",family_members)
```

```
Sisters:  ('Meera', 'Sandhiya', 'Akshaya')
Brothers:  ('Sandeep', 'Suraj')
Siblings:  ('Meera', 'Sandhiya', 'Akshaya', 'Sandeep', 'Suraj')
Total Number of Siblings: 5
Family:  ('Meera', 'Sandhiya', 'Akshaya', 'Sandeep', 'Suraj', 'Ramesh', 'Meena')
```

Question 4:

This program deals with sets, here are creating three sets namely, it_companies, A, B and a list, ages.

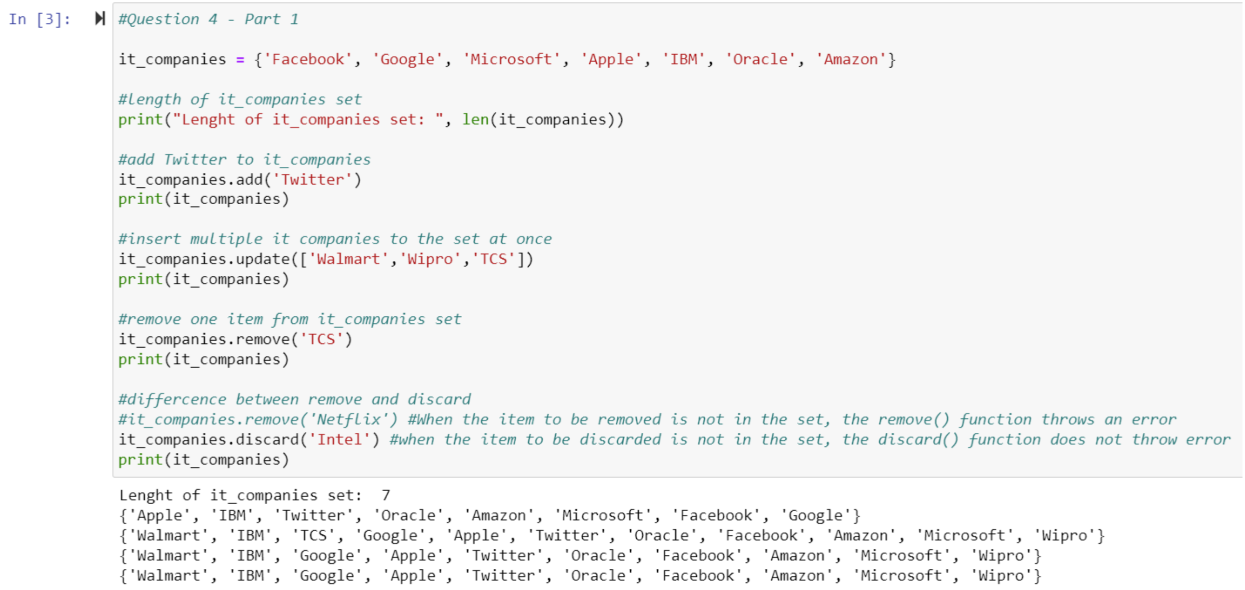
Part 1:

Here, on the first set it_companies we perform basic operation like, finding the length of the set, adding one item to the set, adding multiple items at once to the set. We also tested the difference between remove() and discard()

When the item to be deleted is not in the set, the discard() function does not throw an error where as the remove() function throws an error.

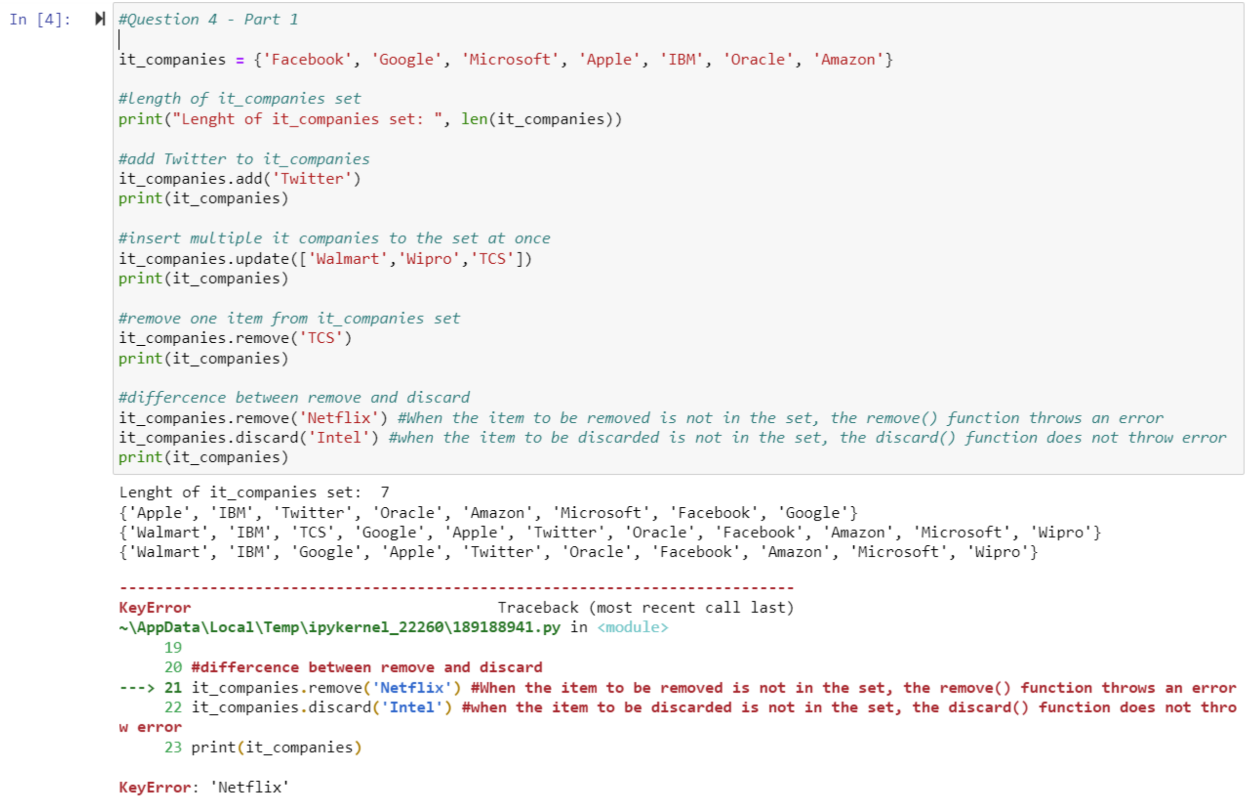
it_companies.remove('TCS') → TCS is removed from the set

it_companies.discard('Intel') → Intel is not in the set, but even then the code got executed without any error



```
In [3]:  #Question 4 - Part 1

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

         #length of it_companies set
         print("Lenght of it_companies set: ", len(it_companies))

         #add Twitter to it_companies
         it_companies.add('Twitter')
         print(it_companies)

         #insert multiple it companies to the set at once
         it_companies.update(['Walmart','Wipro','TCS'])
         print(it_companies)

         #remove one item from it_companies set
         it_companies.remove('TCS')
         print(it_companies)

         #differcence between remove and discard
         #it_companies.remove('Netflix') #When the item to be removed is not in the set, the remove() function throws an error
         it_companies.discard('Intel') #when the item to be discarded is not in the set, the discard() function does not throw error
         print(it_companies)

         Lenght of it_companies set:  7
         {'Apple', 'IBM', 'Twitter', 'Oracle', 'Amazon', 'Microsoft', 'Facebook', 'Google'}
         {'Walmart', 'IBM', 'TCS', 'Google', 'Apple', 'Twitter', 'Oracle', 'Facebook', 'Amazon', 'Microsoft', 'Wipro'}
         {'Walmart', 'IBM', 'Google', 'Apple', 'Twitter', 'Oracle', 'Facebook', 'Amazon', 'Microsoft', 'Wipro'}
         {'Walmart', 'IBM', 'Google', 'Apple', 'Twitter', 'Oracle', 'Facebook', 'Amazon', 'Microsoft', 'Wipro'}
```

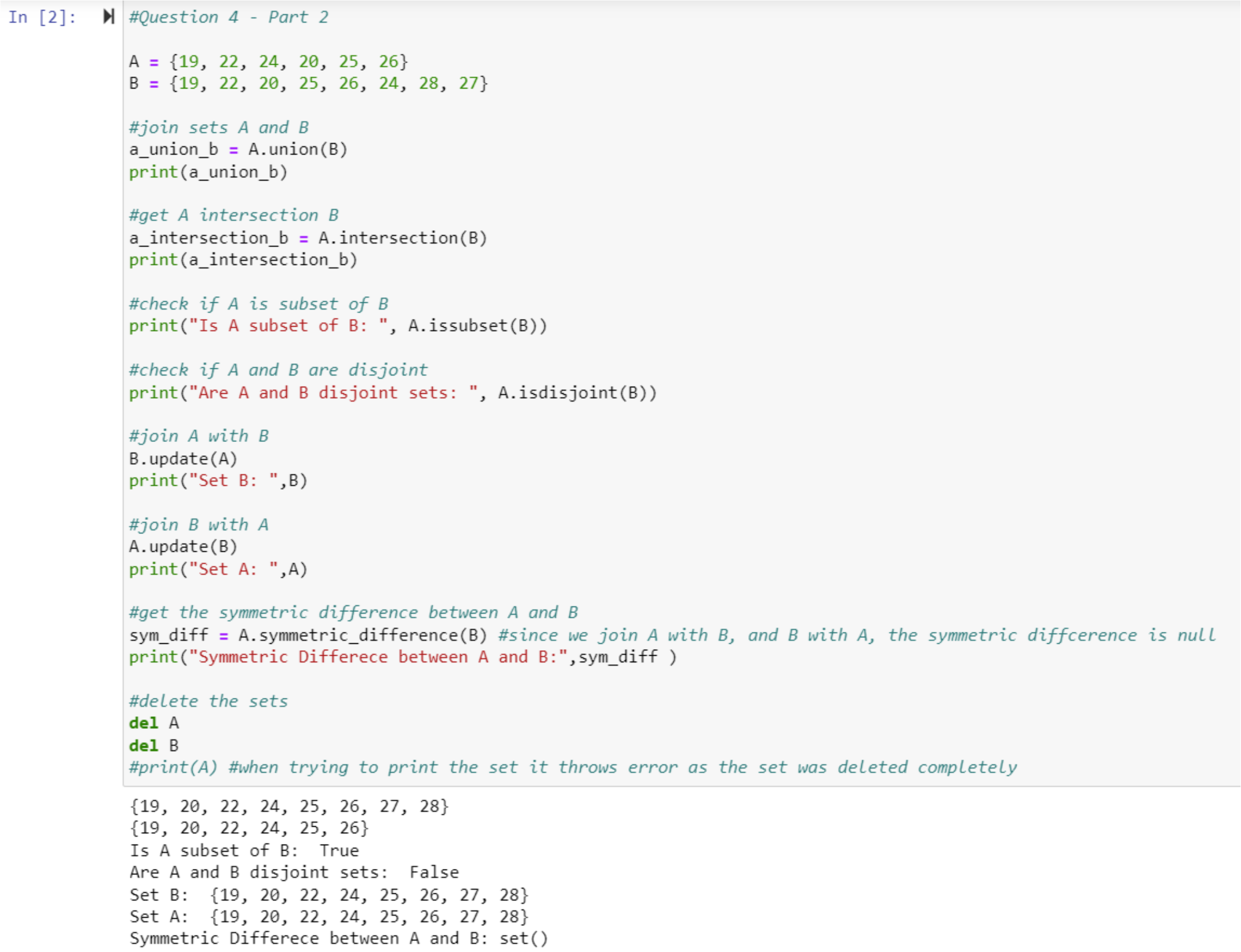
it_companies.remove('Netflix') → Netflix is not in the set, so it throws an error



```
In [4]:  #Question 4 - Part 1

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

         #length of it_companies set
         print("Lenght of it_companies set: ", len(it_companies))

         #add Twitter to it_companies
         it_companies.add('Twitter')
         print(it_companies)

         #insert multiple it companies to the set at once
         it_companies.update(['Walmart','Wipro','TCS'])
         print(it_companies)

         #remove one item from it_companies set
         it_companies.remove('TCS')
         print(it_companies)

         #differcence between remove and discard
         it_companies.remove('Netflix') #When the item to be removed is not in the set, the remove() function throws an error
         it_companies.discard('Intel') #when the item to be discarded is not in the set, the discard() function does not throw error
         print(it_companies)

         Lenght of it_companies set:  7
         {'Apple', 'IBM', 'Twitter', 'Oracle', 'Amazon', 'Microsoft', 'Facebook', 'Google'}
         {'Walmart', 'IBM', 'TCS', 'Google', 'Apple', 'Twitter', 'Oracle', 'Facebook', 'Amazon', 'Microsoft', 'Wipro'}
         {'Walmart', 'IBM', 'Google', 'Apple', 'Twitter', 'Oracle', 'Facebook', 'Amazon', 'Microsoft', 'Wipro'}

         ---------------------------------------------------------------------------
         KeyError                                  Traceback (most recent call last)
         ~\AppData\Local\Temp\ipykernel_22260\189188941.py in <module>
              19 
              20 #differcence between remove and discard
         ---> 21 it_companies.remove('Netflix') #When the item to be removed is not in the set, the remove() function throws an error
              22 it_companies.discard('Intel') #when the item to be discarded is not in the set, the discard() function does not thro
         w error
              23 print(it_companies)

         KeyError: 'Netflix'
```

Part 2:

Here we have two sets A and B, using which we perform the set operations like, union, intersection, subset, to check if the sets are disjoint, joining A with B and then B with A, then finding the symmetric difference (As A and B are joined the symmetric difference would be an empty set) and finally deleting the sets completely.

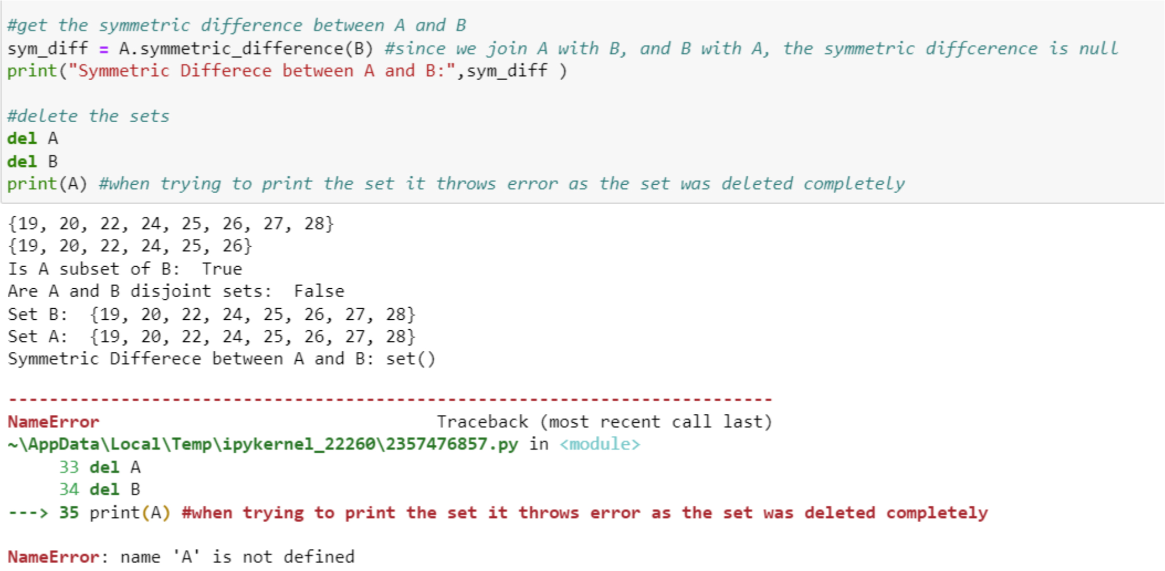


In [2]:

```
#Question 4 - Part 2

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

#join sets A and B
a_union_b = A.union(B)
print(a_union_b)

#get A intersection B
a_intersection_b = A.intersection(B)
print(a_intersection_b)

#check if A is subset of B
print("Is A subset of B: ", A.issubset(B))

#check if A and B are disjoint
print("Are A and B disjoint sets: ", A.isdisjoint(B))

#join A with B
B.update(A)
print("Set B: ",B)

#join B with A
A.update(B)
print("Set A: ",A)

#get the symmetric difference between A and B
sym_diff = A.symmetric_difference(B) #since we join A with B, and B with A, the symmetric diffcerence is null
print("Symmetric Differece between A and B:",sym_diff )

#delete the sets
del A
del B
#print(A) #when trying to print the set it throws error as the set was deleted completely
```

```
{19, 20, 22, 24, 25, 26, 27, 28}
{19, 20, 22, 24, 25, 26}
Is A subset of B:  True
Are A and B disjoint sets:  False
Set B:  {19, 20, 22, 24, 25, 26, 27, 28}
Set A:  {19, 20, 22, 24, 25, 26, 27, 28}
Symmetric Differece between A and B: set()
```

When we try to print the set after deleting it, it will throw an error that it is not defined.

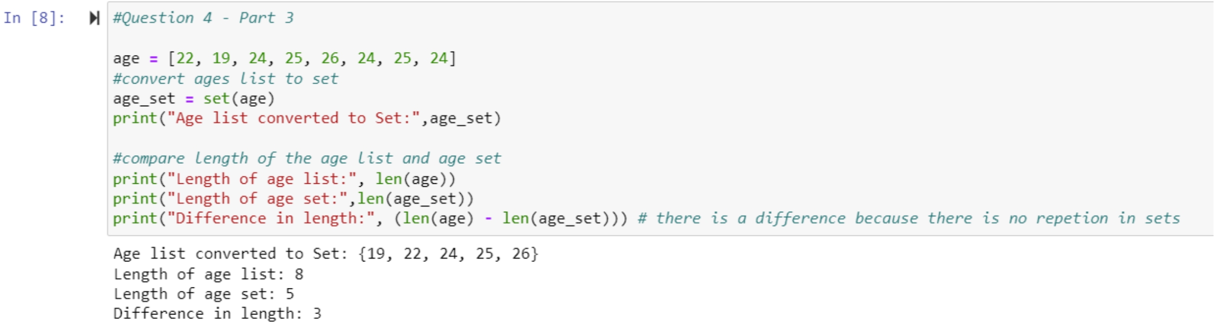


```
#get the symmetric difference between A and B
sym_diff = A.symmetric_difference(B) #since we join A with B, and B with A, the symmetric diffcerence is null
print("Symmetric Differece between A and B:",sym_diff )

#delete the sets
del A
del B
print(A) #when trying to print the set it throws error as the set was deleted completely
```

```
{19, 20, 22, 24, 25, 26, 27, 28}
{19, 20, 22, 24, 25, 26}
Is A subset of B:  True
Are A and B disjoint sets:  False
Set B:  {19, 20, 22, 24, 25, 26, 27, 28}
Set A:  {19, 20, 22, 24, 25, 26, 27, 28}
Symmetric Differece between A and B: set()

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_22260\2357476857.py in <module>
     33 del A
     34 del B
---> 35 print(A) #when trying to print the set it throws error as the set was deleted completely

NameError: name 'A' is not defined
```

Part 3:

Here we convert the ages list to a set and compare the length between list and set. Generally, the length of set might get reduced as sets doesn’t allow repetition.

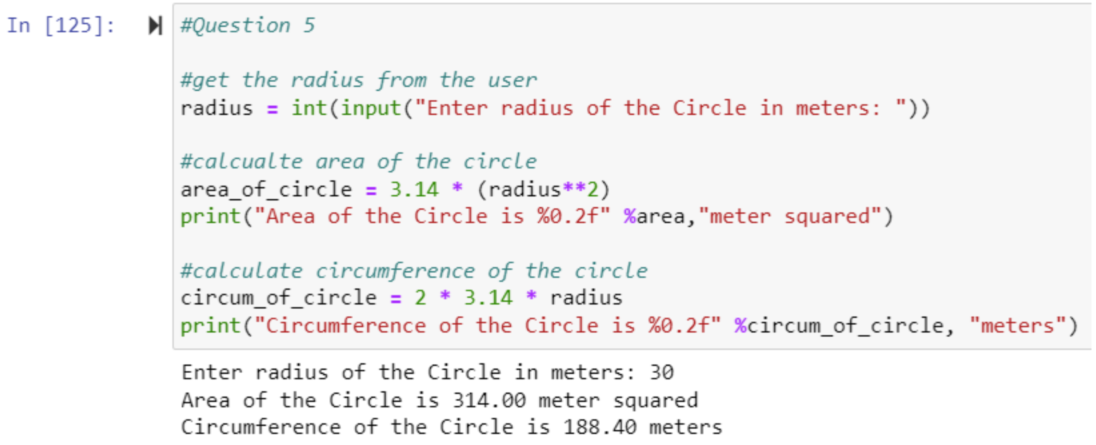


```
In [8]:  #Question 4 - Part 3

         age = [22, 19, 24, 25, 26, 24, 25, 24]
         #convert ages list to set
         age_set = set(age)
         print("Age list converted to Set:",age_set)

         #compare length of the age list and age set
         print("Length of age list:", len(age))
         print("Length of age set:",len(age_set))
         print("Difference in length:", (len(age) - len(age_set))) # there is a difference because there is no repetion in sets
```

```
Age list converted to Set: {19, 22, 24, 25, 26}
Length of age list: 8
Length of age set: 5
Difference in length: 3
```

Question 5:

Here we are getting the radius of the circle as input from the user. With the radius we are calculating the Area and Circumference of the circle.



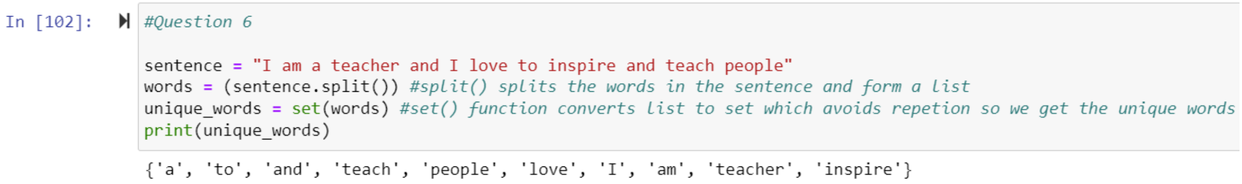
In [125]:

```
#Question 5

#get the radius from the user
radius = int(input("Enter radius of the Circle in meters: "))

#calcualte area of the circle
area_of_circle = 3.14 * (radius**2)
print("Area of the Circle is %0.2f" %area,"meter squared")

#calculate circumference of the circle
circum_of_circle = 2 * 3.14 * radius
print("Circumference of the Circle is %0.2f" %circum_of_circle, "meters")
```

```
Enter radius of the Circle in meters: 30
Area of the Circle is 314.00 meter squared
Circumference of the Circle is 188.40 meters
```

Question 6:

Here, we find the unique words from a given sentence, using split() and set.



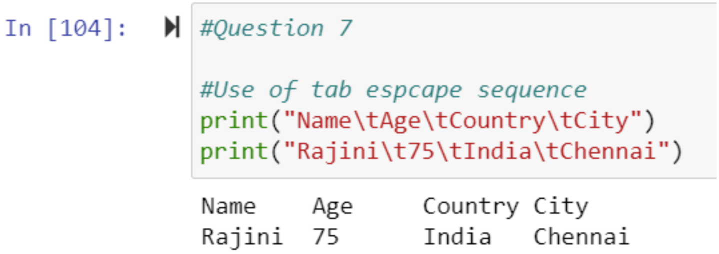
In [102]:

```
#Question 6

sentence = "I am a teacher and I love to inspire and teach people"
words = (sentence.split()) #split() splits the words in the sentence and form a list
unique_words = set(words) #set() function converts list to set which avoids repetion so we get the unique words
print(unique_words)
```

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

Question 7:

Here, we test how the tab escape sequence works by printing few lines.



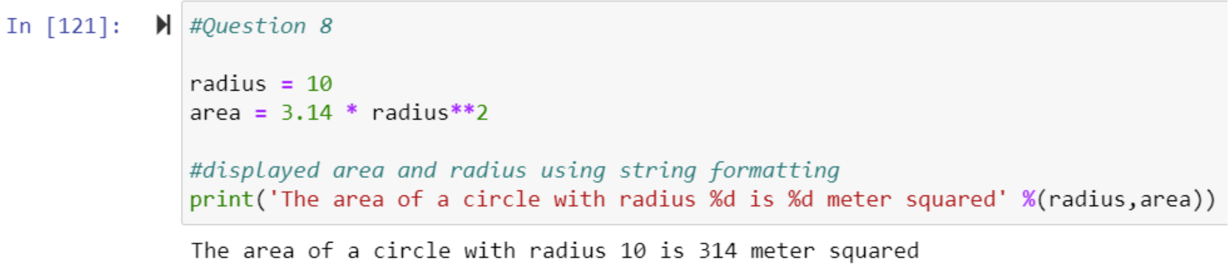
In [104]:

```
#Question 7

#Use of tab espcape sequence
print("Name\tAge\tCountry\tCity")
print("Rajini\t75\tIndia\tChennai")
```

```
Name    Age     Country City
Rajini  75      India   Chennai
```

Question 8:

Here, we use string formatting method to display the radius and area of the circle.

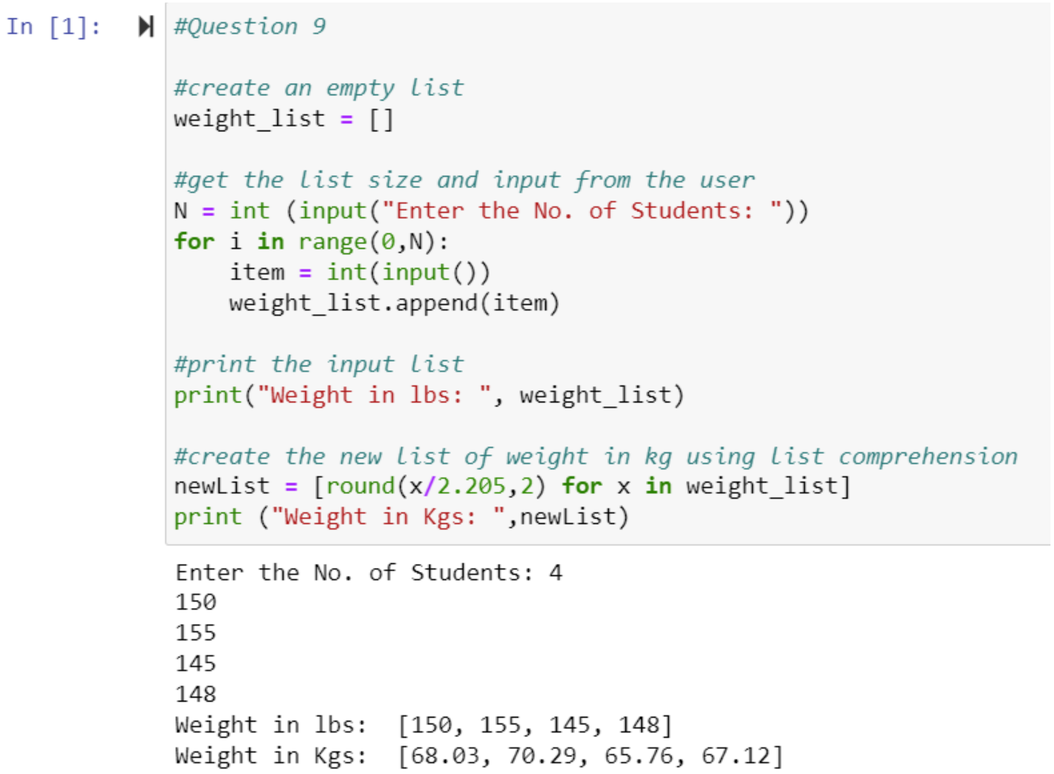


```
In [121]:

#Question 8

radius = 10
area = 3.14 * radius**2

#displayed area and radius using string formatting
print('The area of a circle with radius %d is %d meter squared' %(radius,area))
```

```
The area of a circle with radius 10 is 314 meter squared
```

Question 9:

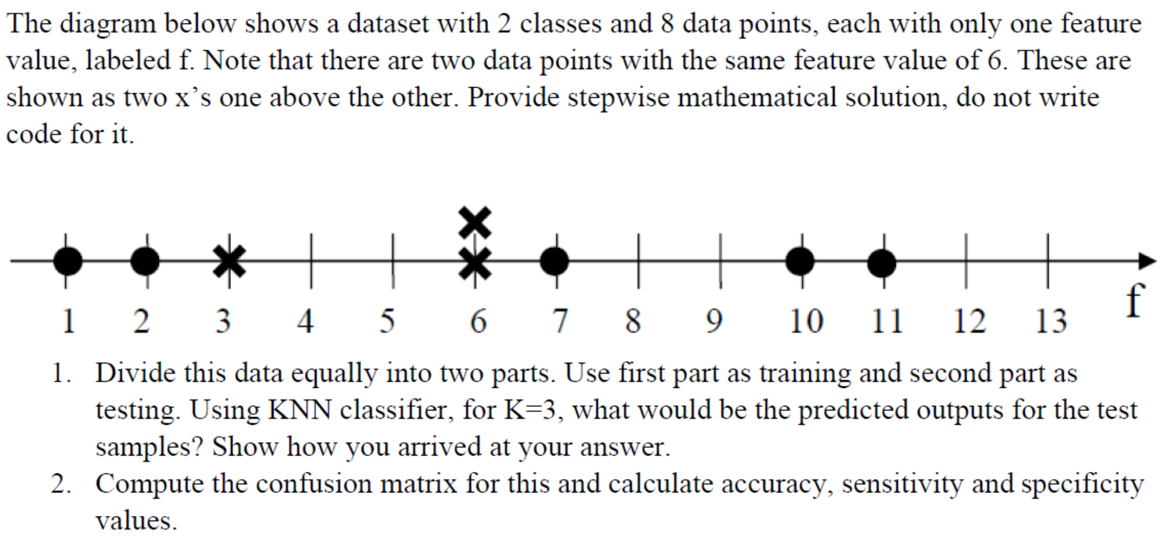
Here, we get the number of students and their weight in lbs from the user and store it in a list using loop. Using list comprehension, we are converting each element of the input (in lbs) list to kilograms and storing it in a new list.



```
In [1]:

#Question 9

#create an empty list
weight_list = []

#get the list size and input from the user
N = int (input("Enter the No. of Students: "))
for i in range(0,N):
    item = int(input())
    weight_list.append(item)

#print the input list
print("Weight in lbs: ", weight_list)

#create the new list of weight in kg using list comprehension
newList = [round(x/2.205,2) for x in weight_list]
print ("Weight in Kgs: ",newList)
```

```
Enter the No. of Students: 4
150
155
145
148
Weight in lbs:  [150, 155, 145, 148]
Weight in Kgs:  [68.03, 70.29, 65.76, 67.12]
```

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.

Given data points = [1, 2, 3, 6, 6, 7, 10, 11]

Here, [1,2,7,10,11] belongs to dot class and [3,6,6] belong to X class. Lets assume 0 for dot class and 1 for X class

● → Class 0

✖ → Class 1

Divide the data equally into two parts.

Lets take the training set as {1, 2, 3, 6}

Test set as {6, 7, 10, 11}

As there is only one feature, the distance for nearest neighbor can be calculated as d(p,q) = |p – q|

Using KNN classifier for k = 3, lets find the predicted outputs for test data

- Test Sample 6 :
  - d(6,1) = 5 ; d(6,2) = 4 ; d(6,3) = 3 ; d(6,6) = 0
  - So, the nearest neighbors are (2,3,6) whose respective classes are (0,1,1)
- Test Sample 7:
  - d(7,1) = 6 ; d(7,2) = 5 ; d(7,3) = 4 ; d(7,6) = 1
  - So, the nearest neighbors are (2,3,6) whose respective classes are (0,1,1)
- Test Sample 10:
  - d(10,1) = 9 ; d(10,2) = 8 ; d(10,3) = 7; d(10,6) = 4
  - So, nearest neighbors are (2,3,6) whose respective classes are (0,1,1)

- Test Sample 11:
  - d(11,1) = 10 ; d(11,2) = 9 ; d(11,3) = 8; d(11,6) = 5
  - So, nearest neighbors are (2,3,6) whose respective classes are (0,1,1)

So the predicted classes of (6,7,10,11) are (1,1,1,1) respectively

Test Sample	Actual	Predicted
6	1	1
7	0	1
10	0	1
11	0	1

- Confusion Matrix

Test Sample	Predicted	
	Class 0	Class 1
Actual Class 0	TN 0	FP 3
Class 1	FN 0	TP 1

- Accuracy = TP + TN/P+N = 1 + 0 / 1 + 3 = 1 / 4

  Therefore, Accuracy = 0.25

- Sensitivity = TP / TP + FN = 1 / 1 + 0 = 1

- Specificity = TN / FP + TN = 0 / 3 + 0 = 0