Name : Abhishek Guleri
Roll No. : 185509
Lab : Data Mining Lab
Assignment No. : 1 & 2
Branch - CSE DD

Q1. L is a list defined as L= [11, 12, 13, 14].

R

```
> L <- list(11, 12, 13, 14)
> L
[[1]]
[1] 11

[[2]]
[1] 12

[[3]]
[1] 13

[[4]]
[1] 14
```

Python

```
>>> L =  [11, 12, 13, 14]
>>> L
[11, 12, 13, 14]
```

i. WAP to add 50 and 60 to L.

R

```
> L <- c(L, 50, 60)
> L
[[1]]
[1] 11

[[2]]
[1] 12

[[3]]
[1] 13

[[4]]
[1] 14
```

```
[[5]]
[1] 50

[[6]]
[1] 60
```

Python

```
>>> L.append(50)
>>> L.append(60)

OR

>>> L.extend((50, 60))

>>> L
[11, 12, 13, 14, 50, 60]
```

ii. WAP to remove 11 and 13 from L.

```
> L[!L %in% c(11, 13)]
[[1]]
[1] 12

[[2]]
[1] 14

[[3]]
[1] 50

[[4]]
[1] 60
```

Python

```
>>> L.remove(50)
>>> L.remove(60)
>>> L
[11, 12, 13, 14]
```

iii. WAP to sort L in ascending order.

```
>>> L.sort()
>>> L
```

```
[11, 12, 13, 14]
```

iv. WAP to sort L in descending order.

```
>>> L.sort(reverse = True)
>>> L
[14, 13, 12, 11]
```

v. WAP to search for 13 in L.

```
>>> if 13 in L:
...   print ("yes");
...
yes
```

vi. WAP to count the number of elements present in L.

```
>>> print(len(L))
4
```

vii. WAP to sum all the elements in L.

```
>>> print(sum(L))
50
```

viii. WAP to sum all ODD numbers in L.

```
>>> print(sum(num for num in L if num % 2 == 1))
24
```

ix. WAP to sum all EVEN numbers in L.

```
>>> print(sum(num for num in L if num % 2 == 0))
26
```

x. WAP to sum all PRIME numbers in L.

```
>>> def findPrime(n):
...   for i in range(2, int(n/2)+1):
...               if n%i == 0:
...                     return False
...   return True
...
>>> for i in L:
...   if findPrime(i):
```

```
...                 sum += i
...
>>> print(sum)
24
```

xi. WAP to clear all the elements in L.

```
>>> L.clear()
>>> L
[]
```

xii. WAP to delete L.

```
>>> del L
>>> L
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'L' is not defined
```

Q2. D is a dictionary defined as D= {1:5.6, 2:7.8, 3:6.6, 4:8.7, 5:7.7}.

```
>>> D = {1:5.6, 2:7.8, 3:6.6, 4:8.7, 5:7.7}
>>> D
{1: 5.6, 2: 7.8, 3: 6.6, 4: 8.7, 5: 7.7}
```

i. WAP to add new entry in D; key=8 and value is 8.8

```
>>> D[8] = 8.8
>>> D
{1: 5.6, 2: 7.8, 3: 6.6, 4: 8.7, 5: 7.7, 8: 8.8}
```

ii. WAP to remove key=2.

```
>>> del D[2]
>>> D
{1: 5.6, 3: 6.6, 4: 8.7, 5: 7.7, 8: 8.8}
```

iii. WAP to check whether 6 keys are present in D.

```
>>> def checkKey(dict, key):
...   if dict.has_key(key):
...         print("Present, value = ", dict[key])
...   else:
...         print("Not present")

>>> checkKey(D, 6)
Not Present
```

iv. WAP to count the number of elements present in D.

```
>>> print(len(D))
 5
```

v. WAP to add all the values present D.

```
>>> def returnSum(dict):
...   sum = 0
...   for i in dict.values():
...         sum += i
...
...   return sum
...
>>> print("Sum :", returnSum(D))
Sum : 36.4
```

vi. WAP to update the value of 3 to 7.1.

```
>>> D
{1: 5.6, 2: 7.8, 3: 6.6, 4: 8.7, 5: 7.7}
>>> D[3] = 7.1
>>> D
{1: 5.6, 2: 7.8, 3: 7.1, 4: 8.7, 5: 7.7}
```

vii. WAP to clear the dictionary.

```
>>> D.clear()
 >>> D
 {}
```

Q3. S1 is a set defined as S1= [10, 20, 30, 40, 50, 60].
S2 is a set defined as S2= [40, 50, 60, 70, 80, 90].

```
>>> S1= {10, 20, 30, 40, 50, 60}
>>> type(S1)
>>> S2= {40, 50, 60, 70, 80, 90}
>>> type(S2)
```

i. WAP to add 55 and 66 in Set S1.

```
>>> S1.add(55)
>>> S1.add(66)
>>> S1 {50, 66, 20, 55, 40, 10, 60, 30}
```

ii. WAP to remove 10 and 30 from Set S1.

```
>>> S1.remove(10)
>>> S1.remove(30)
>>> S1 {50, 66, 20, 55, 40, 60}
```

iii. WAP to check whether 40 is present in S1.

```
>>> if 40 in S1:
...   print("namaste")
...
namaste
```

iv. WAP to find the union between S1 and S2.

```
>>> U = S1.union(S2)
>>> U
{66, 70, 40, 80, 50, 20, 55, 90, 60}
```

v. WAP to find the intersection between S1 and S2.

```
>>> I = S1.intersection(S2)
 >>> I
 {40, 50, 60}
```

vi. WAP to find the S1 - S2.

```
>>> print (S1.difference(S2))
{66, 20, 55}
```

Q4. Write the following program.

i. WAP to print 100 random strings whose length is between 6 and 8.

```
>>> import string
>>> import random
>>> i = 1
>>> while i < 100:
...   N = random.randrange(6, 8, 1)
...   res = ''.join(random.choices(string.ascii_uppercase + string.digits,
k = N))
...   print(str(res))
...   i += 1
...
3WLCD1
LGI3JQ
GMO8H1
K090S8
4MKLOV
HOGYYDX
0PNPRY3
1TPO99T
T3FMA6L
W9IKJ0
N6BXC6
RKWNGC
4IPDAE
4XKB118
BGZ10W
X8XTO2N
:
:
```

ii. WAP to print all prime numbers between 600 and 800.

```
>>> # Range: 600 - 800
>>> lower = 600
>>> upper = 800
>>> for num in range(lower, upper + 1):
...   if num > 1:
 ...        for i in range(2, num):
...              if (num % i) == 0:
...                    break
...         else:
 ...              print(num)
```

```
...
601 607 613 617 619 631 641 643 647 653 659 661 673 677 683 691 701 709 719
727 733 739 743 751 757 761 769 773 787 797
```

iii. WAP to print all numbers between 100 and 1000 that are divisible by 7 and 9.

```
>>> begin = 100
>>> end = 1000
>>> for cnt in range(begin, end+1):
...    if (cnt%7==0 and cnt%9==0):
...   print(cnt)
...
126 189 252 315 378 441 504 567 630 693 756 819 882 945
```

Q5. WAP to create two lists of 10 random numbers between 10 and 30;

```
>>> def RandList(start, end, num):
...   res = []
...   for j in range(num):
...         res.append(random.randint(start, end))
...   return res
...
>>> L1 = RandList(10, 30, 10)
>>> L1
 [14, 15, 12, 21, 27, 18, 12, 13, 18, 22]
>>> L2 = RandList(10, 30, 10)
>>> L2
[15, 19, 27, 23, 30, 29, 21, 28, 10, 21]
```

Find

i. Common numbers in the two lists

```
>>> def common_numbers(l1, l2):
...   l1_set = set(l1)
...   l2_set = set(l2)
...
...   if (l1_set & l2_set):
 ...        print(l1_set & l2_set)
...   else:
...         print("No common numbers")
...
>>> common_number(L1, L2)
{27, 21, 15}
```

ii. Unique numbers in both the list

```
>>> l1_set = set(L1)
>>> l2_set = set(L2)
 >>> l1_set - l2_set {
12, 13, 14, 18, 22}
```

iii. Minimum in both the list

```
>>> min(L1 + L2)
10
```

iv. Maximum in both the list

```
>>> max(L1 + L2)
30
```

v. Sum of both the lists

```
>>> # Sum of two lists element-wise
>>> zip_list = zip(L1, L2)
>>> sum = [x + y for (x,y) in zip_list]
>>> sum
[29, 34, 39, 44, 57, 47, 33, 41, 28, 43]
```

Q6. WAP to create a list of 100 random numbers between 100 and 900.

```
>>> import random
>>> def RandList(start, end, num):
...   res = []
...   for j in range(num):
...         res.append(random.randint(start, end))
...   return res
...
>>> L = RandList(100, 900, 100)
>>> L
 [415, 329, 566, 215, 518, 460, 635, 293, 826, 190, 501, 570, 862, 183,
492, 176, 724, 455, 551, 393, 440, 732, 345, 848, 350, 457, 652, 675, 380,
717, 839, 360, 429, 206, 149, 753, 378, 687, 824, 827, 416, 138, 858, 799,
509, 333, 888, 421, 222, 482, 483, 648, 889, 848, 123, 418, 563, 231, 284,
560, 730, 854, 171, 680, 338, 352, 809, 351, 191, 423, 361, 644, 669, 383,
840, 732, 775, 361, 794, 105, 678, 430, 366, 467, 163, 366, 383, 852, 302,
380, 809, 520, 526, 241, 512, 516, 721, 715, 107, 413]
```

Count and print the:
i. All odd numbers
ii. All even numbers
iii. All prime numbers

```
>>> def checkPrime(number):
...   for i in range(2, number // 2 + 1):
 ...        if number % i == 0:
...               return 0
...   else:
...         return 1
...
>>> even_num = 0
>>> odd_num = 0
>>> prime_num = 0
>>>
>>> for x in L:
...   if not x % 2:
...         even_num += 1
...   if checkPrime(x):
...         prime_num += 1
...   if x % 2:
 ...        odd_num += 1
 ...
>>> even_num
56
>>> odd_num
44
>>> prime_num
15
```

Q7. D is a dictionary defined as D={1:"One",2:"Two",3:"Three",4:"Four", 5:"Five"}.

```
>>> D={1:"One",2:"Two",3:"Three",4:"Four", 5:"Five"}
>>> D
{1: 'One', 2: 'Two', 3: 'Three', 4: 'Four', 5: 'Five'}
```

WAP to read all the keys and values from dictionary and write to the file in the given below
format.
Key1, Value1
Key2, Value2
Key3, Value3

```
>>> for x in D:
...   print('{0}, {1}'.format(x, D.get(x)))
...
1, One
2, Two
3, Three
4, Four
5, Five
```

Q8. L is a list defined as L={"One","Two","Three","Four","Five"}.

```
>>> L=["One","Two","Three","Four","Five"]
>>> type(L)
 <class 'list'>
```

WAP to count the length of reach element from a list and write to the file in the given below
format:
One, 3
Two, 3
Four, 4


```
>>> file = open('StrLenght.txt', 'a')
>>> for i in L:
...   file.write("%s, %d \n" % (i, len(i)))
...
8
8
10
9
9
>>> file.close()
```

Q9. Write to the file 100 random strings whose length is between 10 and 15.

```
>>> import string
>>> import random
>>> file = open('RandStrings.txt', 'a')
>>> i = 1
>>> while i < 100:
...   N = random.randrange(10, 15, 1)
...   res = ''.join(random.choices(string.ascii_uppercase + string.digits,
k = N))
...   file.write("%s\n" % str(res))
...   i += 1
...
>>> file.close()
```

```
[aanya@fedora temp]$ cat RandStrings.txt
8L8AL7P6TZ7Q3Y
UZE0OG32HRS
P2V97GQANQ23N
QDV7D7ZQPIM
E46L9RJQ4J0DUF
E38RGEG8ZK5DV
ZAT622DB2HY
ERS0EJUKLAHZ
4XELLXT2TM
YOQY5L312U
VEBKUGWDV99E
8FMT8V3X2I
D7RG4U37I421
YF8RL55WMLV
PD85K0D675
WEMGY186LZLIN
8NLGIPACBMLHI
1VJVERO9E1YY7
HRQDT0E9GPZ1T
ZA1ILMEV8V9KY5
```

27 JANUARY 2022

Q10. Write to the file all prime numbers between 600 and 800.

```
>>> # Range: 600 - 800
>>> lower = 600
>>> upper = 800
>>> for num in range(lower, upper + 1):
```

```
...   if num > 1:
 ...         for i in range(2, num):
...                 if (num % i) == 0:
...                         break
...         else:
 ...                 print(num)
...
601 607 613 617 619 631 641 643 647 653 659 661 673 677 683 691 701 709 719
727 733 739 743 751 757 761 769 773 787 797
```

Q11. WAP to calculate the time taken by a program.

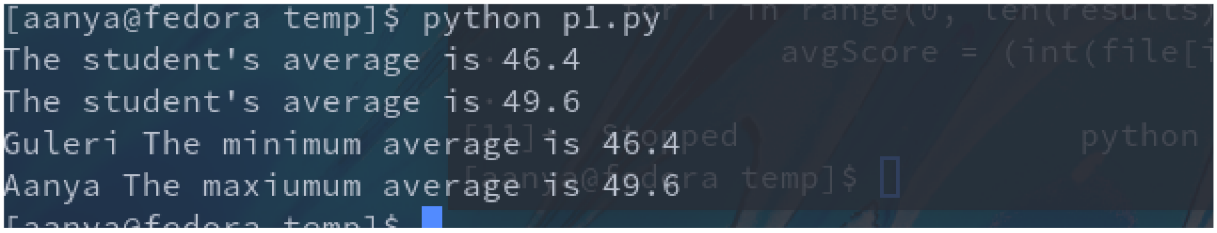
```
>>> import time
>>> start = time.time()
>>> # code here
>>> print("gg bois")
gg bois
>>> end = time.time()
>>> total_time = end - start
>>> print(str(total_time))
17.46066689491272
```

Q12. WAP to create a dictionary of student marks in five subjects and you have to find the student having maximum and minimum average marks.

```
>>> results = { 'Guleri': {'S1' : 45, 'S2' : 50, 'S3' : 46, 'S4' : 48, 'S5'
: 43},
 ...         'Aanya' : {'S1' : 49, 'S2' : 49, 'S3' : 50, 'S4' : 50, 'S5' :
50}}
>>> results
{'Guleri': {'S1': 45, 'S2': 50, 'S3': 46, 'S4': 48, 'S5': 43}, 'Aanya':
{'S1': 49, 'S2': 49, 'S3': 50, 'S4': 50, 'S5': 50}}
```

For sake of simplicity I'm using below format list instead of dictionary

```
>>> file = ["Guleri", "45", "50", "46", "48", "43", "Aanya", "49", "49",
"50", "50", "50"]
>>> file
['Guleri', '45', '50', '46', '48', '43', 'Aanya', '49', '49', '50', '50',
'50']
```




```
file = ["Guleri", "45", "50", "46", "48", "43", "Aanya", "49", "49", "50",
"50", "50"]
class_average = 0
maximum_num = 0
minimum_num = 1000
names_min = [ ]
names_max = [ ]
for i in range(0, len(file), 6):
StudentAverage=(int(file[i + 1]) + int(file[i + 2]) + int(file[i + 3]) +
int(file[i + 4]) + int(file[i + 5]))/5
print("The student's average is", round(StudentAverage,2))

if StudentAverage > maximum_num:
maximum_num = StudentAverage
names_max.clear()
names_max.append(file[i])

if StudentAverage < minimum_num:
minimum_num = StudentAverage
names_min.clear()
names_min.append(file[i])

for i in range(len(names_min)):
print(names_min[i], end = " ")
print("The minimum average is", round(minimum_num,2))

for i in range(len(names_max)):
print(names_max[i], end = " ")
print("The maximum average is", round(maximum_num,2))
```

Q13 WAP to sort the following number of elements in a list and calculate time taken.

Number of elements in list	Time taken to sort
5k	0.04550671577453613
10k	0.12796235084533691

15k	**0.2804598808288574**
20k	**0.4217698574066162**
25k	**0.61993408203125**

```
import random
import time

def RandList(start, end, num):
      res = []
      for j in range(num):
      res.append(random.randint(start, end))
      return res

def partition(arr, low, high):
      i = (low-1)       # index of smaller element
      pivot = arr[high]       # pivot

      for j in range(low, high):

      if arr[j] <= pivot:

            i = i+1
            arr[i], arr[j] = arr[j], arr[i]

      arr[i+1], arr[high] = arr[high], arr[i+1]
      return (i+1)


def quickSort(arr, low, high):
      if len(arr) == 1:
      return arr
      if low < high:

      pi = partition(arr, low, high)

      quickSort(arr, low, pi-1)
      quickSort(arr, pi+1, high)


start = time.time()
```

```
L = RandList(0, 100, 5000)
n = len(L)
quickSort(L, 0, n-1)
print("Sorted array is:")
for i in range(n):
      print("%d" % L[i])

end = time.time()
print("Time Taken : %s" % str(end - start))
```