LAKIREDDY BALI REDDY COLLEGE OF ENGINEERING

(AUTONOMOUS)

L.B. REDDY NAGAR, MYLAVARAM, KRISHNA DIST., A.P.-521 230.

DEPARTMENT

OF

INFORMATION TECHNOLOGY

20CS54-PYTHON PROGRAMMING LAB

.

B.Tech.II SEMESTER

(R20)

A.Y:2020-2021

Python Programming LAB (20CS54)

1. **Pre-requisites: C** Language

2. **Course Educational Objectives (CEOs):**

   In this course student will learn about

Student should be able to have critical understanding to do programming in PYTHON use other higher level language compilers to carryout programming.

Course Outcomes (COs): At the end of the course, the student will be able to :

CO1: Apply building blocks of python in solving computational problems.
CO2: Implement in-built data structures available in python to solve computational problems.
CO3: Implement modular programming, string manipulations and object oriented programming in python.
CO4: Improve individual / teamwork skills, communication & report writing skills with ethical values.

4. Course Articulation Matrix:

Course Code	COs	Program Outcomes												PSOs		
		1	2	3	4	5	6	7	8	9	10	11	12	1	2	3
20CS54	CO1	3	-	-	2	1	-	-	-	-	-	-	-	3	-	-
	CO2	-	3	2	3	2	-	-	-	-	-	-	-	-	3	-
	CO3	-	3	2	3	2	-	-	-	-	-	-	-	-	3	-
	CO4	-	-	-	-	-	-	-	2	2	2	-	-		-	-
1 = Slight (Low) 2 = Moderate (Medium) 3-Substantial(High)																

5. List of Experiments:

SNO	List of Experiments:		Related CO
	Introduction: Language basics and example problems		CO1
1	A	Checking the given number is leap year or not	CO1
2	B	Finding the biggest number among three numbers	CO1
3	C	Displaying reversal of a number	CO1
4	D	To check given number is Armstrong or not	CO1
5	E	To print sum of N natural numbers	CO1
6	F	To check given number is palindrome or not	CO1
7	G	To print factorial of a number	CO1
8	H	To print all the prime numbers with in the given range	CO1
9	I	To calculate the series : S=1+x+x2+x3……x n	CO1
10	J	To print the following pattern : * * * * * *	CO1
	Module-1	**Exercise Programs on Lists.**	
11	1.A	To display elements of the list in reverse order	CO2
12	1.B	Write a Python script to find the minimum and maximum elements without using built-in operations in the lists.	CO2
13	1.C	Write a Python script to remove duplicates from a list.	CO2
14	1.D	Write a Python script to append a list to the second list.	CO2
15	1.E	Write a Python script to create a list with different data types.	CO2

	Module-2	**Exercise Programs on Tuples**	
16	2.A	Write a Python script to find the repeated items of a tuple.	CO2
17	2.B	Write a Python script to create a tuple with different data types	CO2
18	2.C	Write a Python script to replace last value of tuples in a list. Sample list: [(10, 20, 40), (40, 50, 60), (70, 80, 90)] Expected Output: [(10, 20, 100), (40, 50, 100), (70, 80, 100)]	CO2
19	2.D	Write a Python script to sort a tuple by its float element. Sample data: [('item1', '12.20'), ('item2', '15.10'), ('item3', '24.5')] Expected Output: [('item3', '24.5'), ('item2', '15.10'), ('item1', '12.20')]	CO2
	Module-3	**Exercise Programs on Sets and Dictionaries.**	
20	3.A	Write a Python script to add member(s) in a set.	CO2
21	3.B	Write a Python script to perform Union, Intersection, difference and symmetric difference of given two sets.	CO2
22	3.C	Write Python script to test whether every element in S is in T and every element in T is in S.	CO2
23	3.D	Write a Python script to sort (ascending and descending) a dictionary by value.	CO2
24	3.E	Write a Python script to check whether a given key already exists or not in a dictionary.	CO2
25	3.F	Write a Python script to concatenate following dictionaries to create a new one.	CO2
26	3.G	Sample Dictionary : dic1={1:10, 2:20} dic2={3:30, 4:40} dic3={5:50,6:60} Expected Result : {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}	CO2
27	3.H	Write a Python script to print a dictionary where the keys are numbers between 1 and 15 (both included) and the values are square of keys.	CO2
28	3.I	Write a Python program to map two lists into a dictionary.	CO2

	Module-4	**Exercise Programs on functions and recursion.**	
29	4.A	Define a function max_of_three() that takes three numbers as arguments and returns the largest of them.	CO3
30	4.B	Write a program which makes use of function to display all such numbers which are divisible by 7 but are not a multiple of 5, between given range X and Y.	CO3
31	4.C	Define functions to find mean, median, mode for the given numbers in a list.	CO3
32	4.D	Define a function which generates Fibonacci series up to n numbers.	CO3
33	4.E	Implement a python script for factorial of number by using recursion.	CO3
34	4.F	Implement a python script to find GCD of given two numbers using recursion.	CO3
	Module 5	**Exercise programs on Date and Time Modules.**	
35	5.A	Write a Python script to get the current time in Python.	CO3
36	5.B	Write a Python script to get current time in milliseconds in Python	CO3
37	5.C	Write a Python script to print next 5 days starting from today.	CO3
	Module 6	**Exercise programs on Exception Handling.**	
38	6.A	Write a Python script to handle simple errors by using exception handling mechanism	CO3
39	6.B	Write a Python script to handle multiple errors with one except statement.	CO3
	Module 7	**Exercise programs on Strings**	
40	7.A	Implement Python Script to perform various operations on string using string libraries.	CO3
41	7.B	Implement Python Script to check given string is palindrome or not.	CO3

42	7.C	Implement python script to accept line of text and find the number of characters, number of vowels and number of blank spaces in it	CO3
43	7.D	Implement python script that takes a list of words and returns the length of the longest one.	CO3
	Module 8	**Exercise programs on Regular Expressions**	
44	8.A	Write a Python script to check that a string contains only a certain set of characters (in this case a-z, A-Z and 0-9).	CO3
45	8.B	Write a Python script to check whether password is valid or not. Conditions for a valid password are: Should have at least one number. Should have at least one uppercase and one lowercase character. Should have at least one special symbol. Should be between 6 to 20 characters long.	CO3
	Module 9	**Exercise programs on Object Oriented Programming**	
46	9.A	Write a Python script to create and access class variables and methods.	CO3
47	9.B	Write a Python script to implement method overloading.	CO3
48	9.C	Write a Python script to implement single inheritance.	CO3
49	9.D	Write a Python script to implement method overriding.	CO3
	Module 10	**Exercise programs on Python Libraries – Numpy , Pandas , Matplotlib**	
50	10.A	Write a NumPy program to generate a matrix product of two arrays.	CO3
51	10.B	Write a NumPy program to create a random array with 1000 elements and compute the average, variance, standard deviation of the array elements.	CO3
52	10.C	Demonstrate how to download dataset and how to create DataFrame i. Write a Pandas program to get the first 3 rows of a DataFrame ii. Write a Pandas program to select the specified columns	CO3

		and rows from a given data frame . iii. Write a Pandas program to select the rows where the score is missing, i.e. is NaN. iv. Write a Pandas program to insert a new column in existing DataFrame.	
53	10.D	Write a Python programming to display a bar chart using different color for each bar.	CO3
54	10.E	Write a Python programming to create a pie chart with a title.	CO3

BASIC EXAMPLE PROBLEMS

AIM: Implement the python program script for checking the given year is leap year or not

PROGRAM:

```
    y=int(input(" Entre the year :"))
if y%100==0:
    if y%400==0:
        print(y, "is a leap year ")
    else:
        print(y, "is not a leap year ")
else:
    if y%4==0:
        print(y," is a leap year ")
    else:
        print(y, "is not a leap year ")
```

OUTPUT:

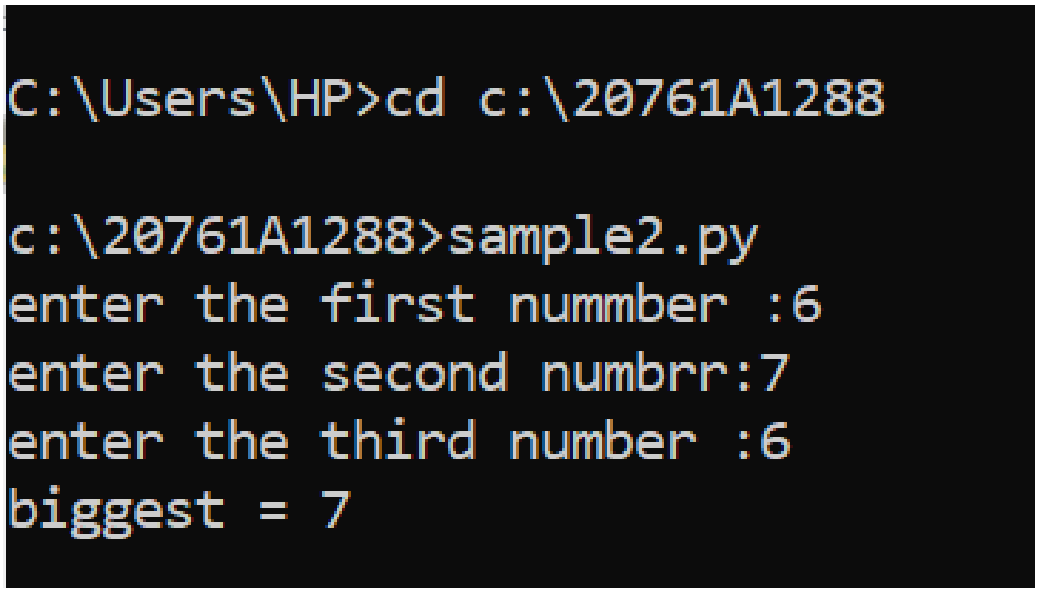
```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>sample1.py
 Entre the year :2000
2000 is a leap year

c:\20761A1288>sample1.py
 Entre the year :1900
1900 is not a leap year

c:\20761A1288>sample1.py
 Entre the year :2010
2010 is not a leap year

c:\20761A1288>sample1.py
 Entre the year :2020
2020  is a leap year
```

AIM: **Implement the python script finding biggest number among three numbers**

PROGRAM:

```
a=int(input("enter the first number :"))
b=int(input("enter the second number:"))
c=int(input("enter the third number :"))
if a>b and a>c :
                    print(" biggest =",a)
else :
                    if b>c :
                        print("biggest =",b)
                    else :
                        print("biggest =",c)
```

OUTPUT:



AIM: Implement python script for displaying the reversal of a given number

PROGRAM:

```
n=int(input("enter the number :"))
print("reverse of the number =",end=" ")
while n!=0:
        r=n%10
        print(r, end=" ")
        n=n//10
```

OUTPUT:

```
c:\20761A1288>sample4.py
enter the number :1234
reverse of the number = 4 3 2 1
c:\20761A1288>sample4.py
enter the number :6543
reverse of the number = 3 4 5 6
```

AIM: Implement the python script to check given number is Armstrong number or not

PROGRAM:

```
n=int(input("enter the number :"))
m=n
s=0
while m!=0:
        r=m%10
        s=s+r**3
        m=m//10
if s==n:
        print(n," is a  armstrong  number ")

else:
        print(n, "is not a armstrong  number ")
```

OUTPUT:

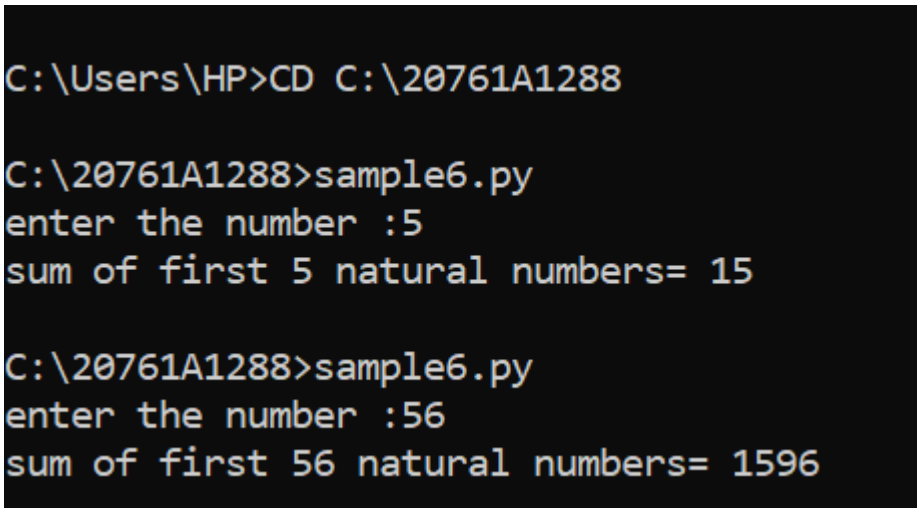
```
c:\20761A1288>sample5.py
enter the number :153
153 is a armstrong number

c:\20761A1288>sample5.py
enter the number :143
143 is not a armstrong number
```

AIM: Implement python script to print sum of N natural numbers

PROGRAM:

```
n=int(input("enter the number :"))
s=0
for i in range(1,n+1):
    s= s+ i
print("sum of first", n, "natural numbers=",s)
```

OUTPUT:



AIM: Implement the python script to check weather the given number is palindrome or not

PROGRAM:

```
n=int(input("enter the number"))
rev=0
m=n
while n!=0:
                    r=n%10
                    rev=rev*10+r
                    n=n//10
if rev==m:
                    print(m, "is a palindrome")
else:
                    print(m, "is not a palindrome")
```

OUTPUT:

```
c:\20761A1288>sample7.py
enter the number123
123 is not a palindrome

c:\20761A1288>sample7.py
enter the number333
333 is a palindrome
```

AIM: Implement python script print factorial of a number

PROGRAM:

```
n=int(input("enter the number :"))
f=1
for i in range(1,n+1):
                    f=f*i
                    print("factorial of ",n ,"is=",f)
```

OUTPUT:

```
c:\20761A1288>sample8.py
enter the number :2
factorial of 2 is= 1
factorial of 2 is= 2

c:\20761A1288>sample8.py
enter the number :3
factorial of 3 is= 1
factorial of 3 is= 2
factorial of 3 is= 6
```

AIM : Implement python script to print all the prime numbers with in the given range

PROGRAM:

```
n1,n2=input("Enter the range:").split()
n1=int(n1)
n2=int(n2)
print("prime numbers in the range",n1,"to",n2,"are")
for n in range(n1,n2+1):
        flag=1
        for i in range(2,n//2+1):
            r=n%i
            if(r==0):
                flag=0
                break
        if flag==1:
            print(n, end=" ")
```

OUTPUT:

```
c:\20761A1288>sample11.py
Enter the range:10 20
prime numbers in the range 10 to 20 are
11 13 17 19
```

AIM: Implement python script to calculate the series:

S=1+x+x2+x3…….xn

PROGRAM:

```
x=int(input("enter the x value:"))
n=int(input("enter the n value:"))
e=1
s=1
while e<=n:
                    s=s+x**e
                    e=e+1
print ("sum of the series=",s)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>sample10.py
enter the x value:5
enter the n value:6
sum of the series= 19531

c:\20761A1288>sample10.py
enter the x value:9
enter the n value:7
sum of the series= 5380840
```

AIM: **Implement python script to print following pattern**

```
                    *

                *       *

            *       *       *
```

PROGRAM:

```
n=int(input("Enter No.of lines : "))
for l in range(1,n+1):
                    print(" "*(n-l),end='')
                    print("* "*l)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>sample13.py
Enter No.of lines : 5
    *
   * *
  * * *
 * * * *
* * * * *
```

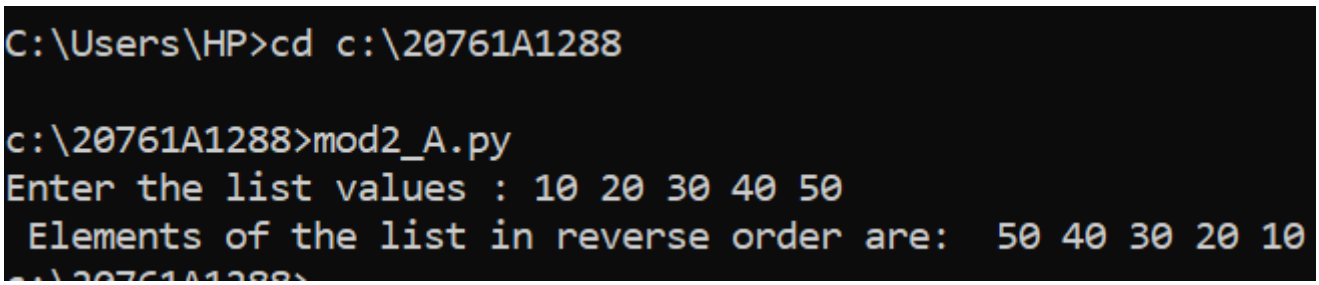
MODULE-1 PROGRAM 1.A

AIM: Implement python script to display elements of list in reverse order.

PROGRAM:

```
l=list(input("Enter the list values : ").split())
l.reverse()
print(" Elements of the list in reverse order are: ",end=' ')
for i in l:
    print(i,end=' ')
```

OUTPUT:



MODULE-1 PROGRAM 1.B

AIM: Implement python script to find minimum and maximum elements without using built-in operations in the lists.

PROGRAM:

```
l=list(map(int,input("Enter list values : ").split()))
minimum=maximum=l[0]
for i in l:
    if i>maximum:
        maximum=i
    if i<minimum:
        minimum=i
print("Minimum Element in the list = ",minimum)
print("Maximum Element in the list = ",maximum)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod2_B.py
Enter list values : 90 89 76 54 43
Minimum Element in the list =  43
Maximum Element in the list =  90
```

MODULE-1 PROGRAM 1.C

AIM: Implement python script to remove duplicates from a list.

PROGRAM:

```
l=list(input("Enter list values : ").split())
l1=[]
for i in l:
    if i not in l1:
        l1.append(i)
print("List after deleting duplicate values is ")
for i in l1:
    print(i,end=' ')
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod2_c.py
Enter list values : 00 sahalya mm 66 55 4
List after deleting duplicate values is
00 sahalya mm 66 55 4
```

MODULE-1 PROGRAM 1.D

AIM: Implement python script to append a list to the second list.

PROGRAM:

```
l1=list(input(" Enter Elements of first list : ").split())
l2=list(input(" Enter Elements of second  list : ").split())
print("Concatenated list is : ")
for i in l2:
    l1.append(i)
print(l1)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod2_D.py
 Enter Elements of first list : sahalya sowjanya ammu shilpi
 Enter Elements of second  list : lbrce 20761A1288 sunny bunty
Concatenated list is :
['sahalya', 'sowjanya', 'ammu', 'shilpi', 'lbrce', '20761A1288', 'sunny', 'bunty']
```

MODULE-1 PROGRAM 1.E

AIM: Implement python script to count the number of strings in a list where the strings in a list where the string length is 2 or more.

PROGRAM:

```
l=list(input("Enter list of strings : ").split())
c=0
for i in l:
    if len(i)>=2:
        c=c+1
print(" No.of strings of length 2 or more =",c)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod2_E.py
Enter list of strings : we but hyderabad it she her 1 4 3
 No.of strings of length 2 or more = 6
```

MODULE-2 PROGRAM 2.A

AIM: Implement python script to create a tuple with different data types.

PROGRAM:

```
n=input(" Enter Your Name : ")
m=int(input("Enter marks in Python Subject : "))
res=bool(input(" Passed in the subject ( Y/N) : "))
a=float(input(" Enter your Average Marks : "))
t=(n,m,res,a)
print("The tuple With Different types is : ",t)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>ex_4prg1.py
 Enter Your Name : sahalya
Enter marks in Python Subject : 96
 Passed in the subject ( Y/N) : 2002
 Enter your Average Marks : 95
The tuple With Different types is :  ('sahalya', 96, True, 95.0)
```

MODULE-2 PROGRAM 2.B

AIM: Implement python script to find the repeated items of a tuple.

PROGRAM:

```
t=tuple(input("Enter the tuple values : ").split())
res=[]
rep=[]
print("Repeated Elements in the Tuple Are : ",end=' ')
for i in t:
    if i in res:
        if i not in rep:
            rep.append(i)
    else:
        res.append(i)
for i in rep:
    print(i,end=' ')
```

OUTPUT:

```
c:\20761A1288>ex_4prg2.py
Enter the tuple values : 4 4 5 5 6 6
Repeated Elements in the Tuple Are :  4 5 6
c:\20761A1288>
```

MODULE-2 PROGRAM 2.C

AIM: Implement python script to replace last value of tuples in a list.

PROGRAM:

```
n=int(input("Enter number of tuple to create : "))
l=[]
for i in range(n):
    print("Enter Elements of %d tuple : "%(i+1),end=' ')
    t=tuple(input().split())
    l.append(t)
r=input(" Enter element to replace : ")
l2=[]
for i in l:
    s=len(i)
    l1=list(i)
    l1[s-1]=r
    l2.append(tuple(l1))
print("original list is : ",l)
print("List after replacing last element of each tuple is : ",l2)
```

OUTPUT:

```
c:\20761A1288>ex_5prg.py
Enter number of tuple to create : 3
Enter Elements of 1 tuple :  lbrce
Enter Elements of 2 tuple :  sahalya
Enter Elements of 3 tuple :  1432
 Enter element to replace : sahalya
original list is :  [('lbrce',), ('sahalya',), ('1432',)]
List after replacing last element of each tuple is :  [('sahalya',), ('sahalya',), ('sahalya',)]
```

MODULE-2 PROGRAM 2.D

AIM: Implement python script to sort a tuple by its float element.

PROGRAM:

```
n=int(input("Enter number of tuple to create : "))
l=[]
for i in range(n):
    print("Enter Elements of %d tuple : "%(i+1),end=' ')
    t=tuple(input().split())
    l.append(t)
print(sorted(l, key = lambda x: float(x[1]), reverse = True))
```

OUTPUT:

```
c:\20761A1288>ex_6prg.py
Enter number of tuple to create : 3
Enter Elements of 1 tuple :  sahalya 99.8
Enter Elements of 2 tuple :  sowjanya 99.8
Enter Elements of 3 tuple :  ammu 99.8
[('sahalya', '99.8'), ('sowjanya', '99.8'), ('ammu', '99.8')]
```

MODULE-3 PROGRAM 3.A

AIM: Implement python script to add members in a set

PROGRAM:

```
S=set()
n=int(input("Enter the number of elments in a set :"))
for i in range(n):
        num=input("enter the element : ")
        S.add(num)
print(" The set of elements are :",S)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>prg90.py
Enter the number of elments in a set :3
enter the element : 2
enter the element : 3
enter the element : 4
 The set of elements are : {'2', '3', '4'}
```

MODULE-3 PROGRAM 3.B

AIM: Implement python script to perform union, Intersection, Difference of given two sets

PROGRAM:

```
s1=set(input("Enter the elements of the first set :").split())
s2=set(input("Enter the elements of the second set :").split())
print("elements of the first set :",s1)
print("elements of the second set :",s2)
print("Union of (S1,S2) is  :",s1|s2)
print("Intersection of s1&s2 is :",s1.intersection(s2))
print("Difference of s1-s2 is :",s1-s2)
print("Symmetric Difference of (s1,s2) is :",s1^s2)
```

OUTPUT :

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>prg20.py
Enter the elements of the first set :2 n3 4
Enter the elements of the second set :3 4 5
elements of the first set : {'2', 'n3', '4'}
elements of the second set : {'4', '5', '3'}
Union of (S1,S2) is  : {'2', 'n3', '4', '3', '5'}
Intersection of s1&s2 is : {'4'}
Difference of s1-s2 is : {'2', 'n3'}
Symmetric Difference of (s1,s2) is : {'5', '2', 'n3', '3'}
```

MODULE-3 PROGRAM 3.C

AIM: Implement the python script to check whether every element in S is in T and every element in T is in S

PROGRAM:

```
S=set(input("Enter the elements of the set S :").split())
T=set(input("Enter the elements of the set T :").split())
for i in S:
        flag=True
        if i not in T:
            flag=False
            break
for i in T:
        flag1=True
        if i not in S:
            flag1=False
            break
if flag==True:
        print("S is a subset of T")
else:
        print("S is not a subset of T")
if flag1==True:
        print("T is a subset of S")
else:
        print("T is not a subset of S")
```

OUTPUT:

```
c:\20761A1288>prg25.py
Enter the elements of the set S :23 4 5 5 6
Enter the elements of the set T :23 4 5 5 6
S is a subset of T
T is a subset of S
```

MODULE -3 PROGRAM 3.D

PROGRAM: Implement python script to sort (ascending and descending) a dictionary by values.

```
n=int(input("Enter No Of Students : "))
markdict={}
for i in range(n):
    sname,marks=input("Enter name and marks of %d student : "%(i+1)).split()
    markdict[sname]=int(marks)

marklist=list(markdict.items())
marklist1 = sorted(markdict.items(), key=lambda x:x[1])
l=len(marklist)
for i in range(l-1):
    for j in range(l-i-1):
        if marklist[j][1]<marklist[j+1][1]:
            t=marklist[j]
            marklist[j]=marklist[j+1]
            marklist[j+1]=t
sortdict=dict(marklist)
print(" Data in sorted ( Descending ) order : ",sortdict)
sortdict=dict(marklist1)
print(" Data in sorted ( Ascending ) order : ",sortdict)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>ex_7prg1.py
Enter No Of Students : 2
Enter name and marks of 1 student : sahalya 98
Enter name and marks of 2 student : sowjanya 98
 Data in sorted ( Descending ) order :  {'sahalya': 98, 'sowjanya': 98}
 Data in sorted ( Ascending ) order :  {'sahalya': 98, 'sowjanya': 98}
```

MODULE-3 PROGRAM 3.E

AIM: Implement python script to check whether a given key already exists or not in a dictionary.

PROGRAM:

```
n=int(input("Enter No Of Objects in Dictionry : "))
d={}
for i in range(n):
    k,v=input("Enter key and value of %d object : "%(i+1)).split()
    d[k]=v
skey=input("Enter the key search : ")
if skey in d.keys():
    print(" Key Exist in the Dictionary the corresponding value of the key is : ",d[skey])
else:
    print(" Key Doesn't exist in the dictionary ")
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>ex_7prg2.py
Enter No Of Objects in Dictionry : 2
Enter key and value of 1 object : sahalya 96
Enter key and value of 2 object : sowjanya 98
Enter the key search : sahalya
 Key Exist in the Dictionary the corresponding value of the key is :  96
```

MODULE-3 PROGRAM 3.F

AIM: Implement python script to concatenate following dictionaries to create new one.

Sample Dictionary: dic1={1:10,2:20} dic2={3:30,4:40} dic3={5:50,6:60}

Expected Result: {1:10, 2:20,3:30, 4:40, 5:50, 6:60}

PROGRAM:

```
d1={1:10, 2:20}

d2={3:30, 4:40}

d3={5:50, 6:60}

d={}

for i in (d1,d2,d3):

        d.update(i)

print(" Dictionary1 : ",d1)

print(" Dictionary2 : ",d2)

print(" Dictionary3 : ",d3)

print(" The Concatenated Dictionary is : ",d)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>ex_7prg3.py
 Dictionary1 :  {1: 10, 2: 20}
 Dictionary2 :  {3: 30, 4: 40}
 Dictionary3 :  {5: 50, 6: 60}
 The Concatenated Dictionary is :  {1: 10, 2: 20, 3: 30, 4: 40, 5: 50, 6: 60}

c:\20761A1288>
```

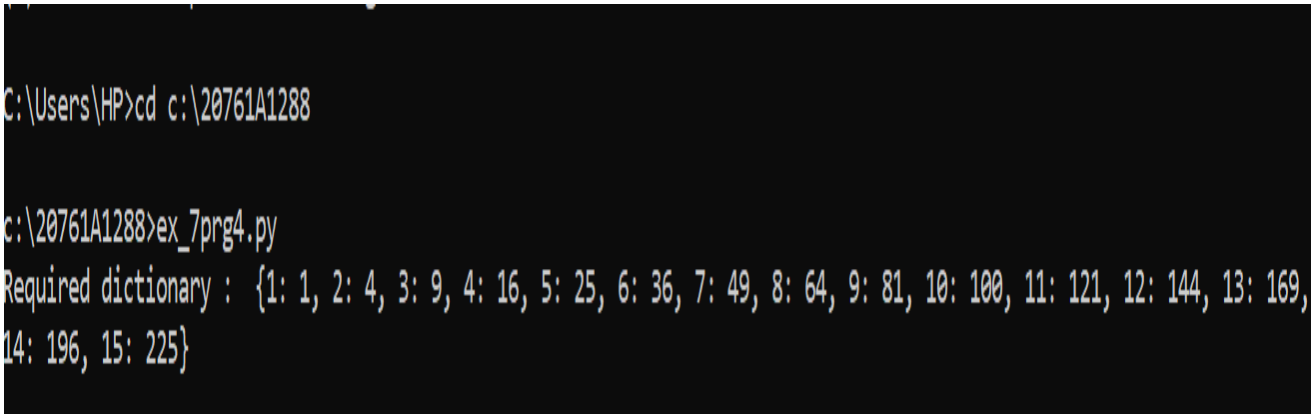
MODULE-3 PROGRAM 3.G

AIM: Implement python script to print a dictionary where the keys are numbers between 1 and 15 and the values are squares of keys.

PROGRAM:

```
d={n:n*n for n in range(1,16)}

print("Required dictionary : ",d)
```

OUTPUT:



MODULE-3 PROGRAM 3.H

AIM: Implement python script to map two lists into a dictionary.

PROGRAM:

```
l=list(input("Enter list of subjects : ").split())
m=list(input("Enter marks in each subject : ").split())
d=dict(zip(l,m))
print(" mapping of two lists into dictionary is : ",d)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>ex_7prg5.py
Enter list of subjects : PC LATT MATLAB SCILAB AP CI DS
Enter marks in each subject : 98 97 96 100 100 98
 mapping of two lists into dictionary is :  {'PC': '98', 'LATT': '97', 'MATLAB': '96', 'SCILAB': '100', 'AP': '100', 'CI
': '98'}
```

MODULE-3 PROGRAM 3.I

AIM: To define a function max_of_three() that takes three numbers as arguments and returns the largest of them.

PROGRAM:

```
def max_of_three(a,b,c):
    if a>b and a>c:
        return a
    elif b>c:
        return b
    else:
        return c
a,b,c=map(int,input("Enter any three values : ").split())
m=max_of_three(a,b,c)
print(" Largest number = ",m)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod6_A.py
Enter any three values : 55 66 77
 Largest number =  77
```

MODULE-4 PROGRAM 4.A

AIM: Which makes use of function to display all such numbers which are divisible by 7 but are not a multiple of 5, between given range X and Y.

PROGRAM:

```
x,y=map(int,input(' Enter Range : ').split())
n=[]
for i in range(x,y+1):
    if (i%7==0) and (i%5!=0):
        n.append(str(i))
print('numbers Divisible by 7 and not a multiple of 5 in the range are
')
print(', '.join(n))
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod6_B.py
 Enter Range : 5 89
numbers Divisible by 7 and not a multiple of 5 in the range are
7, 14, 21, 28, 42, 49, 56, 63, 77, 84

c:\20761A1288>
```

MODULE-4 PROGRAM 4.B

AIM: Define functions to find mean, median, mode for the given numbers in a list.

PROGRAM:

```
def mean(a):
    s=sum(a)
    m1=s/len(a)
    print("Mean is : ",m1)
def median(a):
    a.sort()
    n=len(a)
    if n%2==0:
        m2=(a[n//2]+a[(n-1)//2])/2
    else:
        m2=a[(n-1)//2]
    print ("Median is : ",m2)
def mode_list(a):
    d={}
    for i in a:
        if i in d:
            d[i]=d[i]+1
        else:
            d[i]=1
```

```
    val=max(list(d.values()))
    l=[]
    for k,v in d.items():
        if v==val:
            l.append(k)
    print("Mode : ",l)


lst=list(map(int,input(" Enter List Of Values : ").split()))
mean(lst)
median(lst)
mode_list(lst)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod6_C.py
 Enter List Of Values : 1 2 3 4 55 66 77 8999 8 77 55 66 1 2 3
Mean is :  627.9333333333333
Median is :  8
Mode :  [1, 2, 3, 55, 66, 77]
```

MODULE-4 PROGRAM 4.C

AIM: Define a function, which generates Fibonacci series up to n numbers.

PROGRAM:

```
def fib(n):
    if n==0 or n==1:
        return n
    return fib(n-1)+fib(n-2)
n=int(input('enter the No.Of terms : '))
print(' Fibonacci series upto %d terms'%n)
for i in range(n):
    f=fib(i)
    print(f,end=' ')
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod6_D.py
enter the No.Of terms : 6
 Fibonacci series upto 6 terms
0 1 1 2 3 5
```

MODULE-4 PROGRAM 4.D

AIM: Implement a python script for factorial of number by using recursion.

PROGRAM:

```
def fact(n):
    if n==0:
        return 1
    return n*fact(n-1)

n=int(input(" Enter the number : "))
f=fact(n)
print(" Factorial of %d = %d"%(n,f))
```

OUTPUT:

```
c:\20761A1288>mod6_E.py
 Enter the number : 5
 Factorial of 5 = 120

c:\20761A1288>
```

MODULE-4 PROGRAM 4.F

AIM: Implement a python script to find GCD of given two numbers using recursion.

PROGRAM:

```
def GCD(x,y):
    if x==y:
        return x
    elif x>y:
        return GCD(x-y,y)
    else:
        return GCD(x,y-x)
a,b=map(int,input(" Enter any two numbers : ").split())
g=GCD(a,b)
print("GCD of %d ,%d is %d"%(a,b,g))
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod6_F.py
 Enter any two numbers : 66 7
GCD of 66 ,7 is 1
```

MODULE-5 PROGRAM 5.A

AIM: Implement a python script to get the current time in Python.

PROGRAM:

```
from datetime import datetime
d=datetime.now()
d1=d.strftime('%H:%M:%S')
print(" Current Time = ",d1)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod7_A.py
 Current Time =  16:13:57
```

MODULE-5 PROGRAM 5.B

AIM: Implement a python script to get current time in milliseconds in Python.

PROGRAM:

```
from datetime import datetime

d = datetime.today()

print(" Current Time = ",d.strftime('%H:%M:%S'))

ms=(d.hour*60*60+d.minute*60+d.second)*1000+d.microsecond//1000

print("Current time in milliseconds = ",ms)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod7_B.py
 Current Time =  16:16:31
Current time in milliseconds =  58591217

c:\20761A1288>
```

MODULE-5 PROGRAM 5.C

AIM: Implement a python script to print next 5 days starting from today.

PROGRAM:

```
from datetime import datetime,timedelta

base = datetime.now()

print(" Todays Date is : ",base.strftime('%d-%m-%Y'))

print(" Next FIve Days dates are : ")

for x in range(1,6):

        d=base + timedelta(days=x)

        print(d.strftime('%d-%m-%Y'))
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod7_C.py
 Todays Date is :  17-08-2021
 Next FIve Days dates are :
18-08-2021
19-08-2021
20-08-2021
21-08-2021
22-08-2021

c:\20761A1288>
```

MODULE-6 PROGRAM 6.A

AIM: Implementation of Python script to handle simple errors by using exception-handling mechanism.

PROGRAM:

```
try:
        a=int(input(" Enter First Value : "))
        b=int(input(" Enter the second Value : "))
        print(" Sum = ",a+b)
        print(" Quotient = ",a//b)
        print(" Average = ",(a+b)/2.0)
except ZeroDivisionError:
        print(" Division is not possible because the denominator is Zero
")
except ValueError:
        print(" please provide integer value only")
else:
        print(" Operation Completed Without Exception ")

try:
        print(" Difference = ",a-b)
        print(" Product = ",a*b)
except NameError as msg:
        print(" Error is : ",msg)
finally:
        print(" Possible Arithmetic operations completed ")
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod8_A.py

c:\20761A1288>mod8_a.py
 Enter First Value : 5
 Enter the second Value : 6
 Sum =  11
 Quotient =  0
 Average =  5.5
 Operation Completed Without Exception
 Difference =  -1
 Product =  30
 Possible Arithmetic operations completed
```

MODULE-6 PROGRAM 6.B

AIM: Implementation of Python script handle multiple errors with one except statement.

PROGRAM:

```
try:
        l=['item0','item1','item2']
         a = int(input("Enter a:"))
        b = int(input("Enter b:"))
        c = a//b
        print("a//b = %d"%c)
        print(" list item is : ",l[c])
except Exception as e:
   print("Error is : ",e)
else:
   print("No Exception Occured")
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod8_B2.py
Enter a:10
Enter b:5
a//b = 2
 list item is :  item2
No Exception Occured

c:\20761A1288>mod8_B2.py
Enter a:10
Enter b:0
Error is :  integer division or modulo by zero

c:\20761A1288>mod8_B2.py
Enter a:10
Enter b:
Error is :  invalid literal for int() with base 10: ''

c:\20761A1288>mod8_B2.py
Enter a:100
Enter b:10
a//b = 10
Error is :  list index out of range
```

MODULE-7 PROGRAM 7.A

AIM: Implementation of python Script to perform various operations on string using string libraries.

PROGRAM:

```
# String methods
s=input(" Enter the string : ")
print(" Changing Case of a stirng ")
print(s," In Lower Case : ",s.lower())
print(s," In Upper case : ",s.upper())
print(s," In Change Case : ",s.swapcase())
print(s," In Title() form : ",s.title())
print(s," In Capitalize() form : ",s.capitalize())
ss=input( " Enter the substring : ")
flag=s.startswith(ss)
if flag==True:
        print(s," starts with ",ss)
else:
        print(s, " not starts with ",ss)
flag=s.endswith(ss)
if flag==True:
        print(s," ends with ",ss)
else:
        print(s, " not ends with ",ss)
print("Different words of the string are : ")
l=s.split()
for i in l:
        print(i)
print(" Joining words with --> ")
print('-->'.join(l))
print(" Display 'LBRCE' with justification methods ")
print(" 'LBRCE' after left justify with width 15 is :
","LBRCE".ljust(15,'-'))
print(" 'LBRCE' after right justify with width 15 is
:","LBRCE".rjust(15,'-'))
print(" 'LBRCE' after center justify with width 15 is
:","LBRCE".center(15,'-'))
```

```
s2="   This is Python Class    "
print("Apply strip methods on :",s2)
print(s2," After strip()  :",s2.strip())
print(s2," After lstrip() :",s2.lstrip())
print(s2," After rstrip() :",s2.rstrip())

print(" FIND / REPLACE methods on string ")
s3=input(" Enter Some String with repeted words in it  : ")
s4=input(" ENter the repeted word : ")
p=-1
while True:
        p=s3.find(s4,p+1,len(s3))
        if p==-1:
                break
        print(s4, " found at position ",p," in ",s3)
print("{} Occurs {} in the {}".format(s4,s3.count(s4),s3))
s5=input(" Enter the word to replace : ")
print(s3, " after replace is : ",s3.replace(s4,s5))

ch=input("Enter any character:")
if ch.isalnum():
        print("Alpha Numeric Character")
        if ch.isalpha():
                print("Alphabet character")
                if ch.islower():
                        print("Lower case alphabet character")
                else:
                        print("Upper case alphabet character")
        else:
                print("it is a digit")
elif ch.isspace():
        print("It is space character")
else:
        print("Non space Special Character")
```

OUTPUT:

```
c:\20761A1288>mod8_C.py
 Enter the string : This is lbrce college
 Changing Case of a stirng
This is lbrce college  In Lower Case :  this is lbrce college
This is lbrce college  In Upper case :  THIS IS LBRCE COLLEGE
This is lbrce college  In Change Case :  tHIS IS LBRCE COLLEGE
This is lbrce college  In Title() form :  This Is Lbrce College
This is lbrce college  In Capitalize() form :  This is lbrce college
 Enter the substring : college
This is lbrce college  not starts with  college
This is lbrce college  ends with  college
Different words of the string are :
This
is
lbrce
college
 Joining words with -->
This-->is-->lbrce-->college
 Display 'LBRCE' with justification methods
 'LBRCE' after left justify with width 15 is :  LBRCE----------
 'LBRCE' after right justify with width 15 is : ----------LBRCE
 'LBRCE' after center justify with width 15 is : -----LBRCE-----
Apply strip methods on :    This is Python Class
   This is Python Class      After strip()  : This is Python Class
   This is Python Class      After lstrip() : This is Python Class
   This is Python Class      After rstrip() :    This is Python Class
 FIND / REPLACE methods on string
 Enter Some String with repeted words in it  :
```

```
 Enter Some String with repeted words in it  : This is lbrce college.I am the student in lbrce.lbrce is the best
 ENter the repeted word : lbrce
lbrce  found at position  8  in  This is lbrce college.I am the student in lbrce.lbrce is the best
```

```
lbrce  found at position  8  in  This is lbrce college.I am the student in lbrce.lbrce is the best
lbrce  found at position  42  in  This is lbrce college.I am the student in lbrce.lbrce is the best
lbrce  found at position  48  in  This is lbrce college.I am the student in lbrce.lbrce is the best
lbrce Occurs 3 in the This is lbrce college.I am the student in lbrce.lbrce is the best
 Enter the word to replace : python
This is lbrce college.I am the student in lbrce.lbrce is the best  after replace is :  This is python college.I am the s
tudent in python.python is the best
Enter any character:1
Alpha Numeric Character
Alphabet character
Lower case alphabet character
```

MODULE-7 PROGRAM 7.B

AIM: Implementation of python Script to check given string is palindrome or not.

PROGRAM:

```
s=input("Enter the String : ")
s1=s[::-1]
print(" Reverse of string is : ",s1)
if s==s1:
        print(s," is a Palindrome String")
else:
        print(s," is Not a Palindrome String")
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod9_B.py
Enter the String : sahalya
 Reverse of string is :  aylahas
sahalya  is Not a Palindrome String

c:\20761A1288>mada
'mada' is not recognized as an internal or external command,
operable program or batch file.

c:\20761A1288>mod9_B.py
Enter the String : madam
 Reverse of string is :  madam
madam  is a Palindrome String

c:\20761A1288>_
```

MODULE-7 PROGRAM 7.C

AIM: Implementation of python script to accept line of text and find the number of characters, number of vowels and number of blank spaces in it.

PROGRAM:

```
s=input("Please Enter line of Text : ")
cvol=0
cspace=0
cc=0
for ch in s:
        cc=cc+1
        if (ch=='a' or ch=='e' or ch=='i' or ch=='o' or ch=='u' or ch=='A'
or ch=='E' or ch=='I' or ch=='O' or ch=='U'):
                cvol = cvol + 1
        elif ch==' ':
                cspace = cspace + 1

print("Number of Vowels in the Text = ",cvol)
print("Number of spaces = ",cspace)
print("Number of characters = ",cc)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod9_C.py
Please Enter line of Text : This is my book
Number of Vowels in the Text =  4
Number of spaces =  3
Number of characters =  15

c:\20761A1288>
```

MODULE-7 PROGRAM 7.D

AIM: Implementation of python script that takes a list of words and returns the length of the longest one.

PROGRAM:

```
l=input("Enter List Of Words : ").split()
llw=0
for w in l:
        if len(w) > llw:
                llw = len(w)

print(" Length of Longest word = ",llw)
print(" The Longest words are : ")
for w in l:
        if len(w)==llw:
                print(w)
```

OUTPUT:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod9_D.py
Enter List Of Words : sahalya sowjanya
 Length of Longest word =  8
 The Longest words are :
sowjanya
```

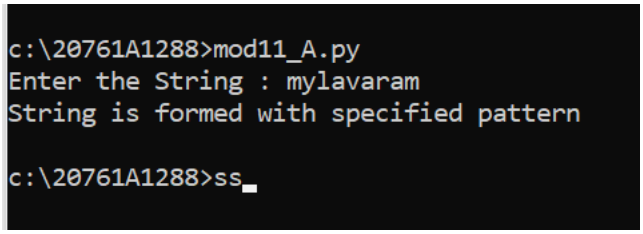
MODULE-8 PROGRAM 8.A

Aim: Write a Python script to check that a string contains only a certain set of characters (in this case a-z, A-Z and 0-9).

Program:

```
Import re
def is_alnum(s):
        m=re.fullmatch("[a-zA-Z0-9]*",s)
        if m!= None:
                print("String is formed with specified pattern")
        else:
                print("String is not formed with specified pattern ")

s=input("Enter the String : ")
is_alnum(s)
```

Output:



MODULE-8 PROGRAM 8.B

Aim: Write a Python script to check whether password is valid or not.

Conditions for a valid password are:

Should have at least one number.
Should have at least one uppercase and one lowercase character.
Should have at least one special symbol.
Should be between 6 to 20 characters long.

Program:

```
import re
password=input(" Enter the Password : ")
flag = 0
while True:
        if len(password)<6 or len(password)>20:
                flag=-1
                break
        elif not re.search("[a-z]", password):
                flag = -1
                break
        elif not re.search("[A-Z]", password):
                flag = -1
                break
        elif not re.search("[0-9]", password):
                flag = -1
                break
        elif not re.search("[^a-zA-Z0-9]", password):
                flag = -1
                break
        elif re.search("\s", password):
                flag = -1
                break
        else:
                flag = 0
                print("Valid Password")
                break
if flag ==-1:
        print("Not a Valid Password")
```

Output:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod11_B.py
 Enter the Password : sahalya@123
Not a Valid Password

c:\20761A1288>mod11_B.py
 Enter the Password : Sahalya@123
Valid Password
```

MODULE-9 PROGRAM 9.A

Aim: Write a python script to Create &access class variables and methods

Program:

```
class Robot:
population=0
what="Humanoid Robot"
version=1.0
speed="1THZ"
memory="1ZB"

def _init_(self, name):
self.name = name
print('\n < Initialising ',self.name,' > ')
Robot.population += 1

def _del_(self):
print(self.name , ' is being destroyed!\n')
Robot.population -= 1
if Robot.population == 0:
print(self.name ,' was the last one.')
else:
print(' \n There are still ', Robot.population ,' robots working.')

def update(cls):
cls.version=2.0
cls.speed = "2THZ"
cls.memory="2ZB"

def sayHi(self):
print("\nHai i am a : ",self.what)
print("My Name is : ",self.name)
print("Version : ",self.version)
print("Speed : ",self.speed)
print("Memory : ",self.memory)

def howMany():
print('\n No of Robots Available curruently : ',Robot.population)


d=Robot('Chitti')
d.sayHi()
Robot.howMany()
d1=Robot('Vennela')
```

```
d1.update()
d1.sayHi()
Robot.howMany()
print("\nRobots can do some work here.\n")
print("\nRobots have finished their work. So let's destroy them.\n")
del d
del d1
Robot.howMany()
```

Output:

```
 < Initialising  Chitti  >

Hai i am a :  Humanoid Robot
My Name is :  Chitti
Version :  1.0
Speed :  1THZ
Memory :  1ZB

 No of Robots Available curruently :  1

 < Initialising  Vennela  >

Hai i am a :  Humanoid Robot
My Name is :  Vennela
Version :  2.0
Speed :  2THZ
Memory :  2ZB

 No of Robots Available curruently :  2
```

```
Robots can do some work here.


Robots have finished their work. So let's destroy them.

Chitti  is being destroyed!


 There are still  1  robots working.
Vennela  is being destroyed!

Vennela  was the last one.

 No of Robots Available curruently :  0
```

MODULE-9 PROGRAM 9.B

Aim: Write a python script to implement method overloading

Program:

```
from multipledispatch import dispatch
```

```
@dispatch(int,int)
def product(first,second):
   result = first*second
   print(" Product of two integers = ",result)
@dispatch(int,int,int)
def product(first,second,third):
   result  = first * second * third
   print(" Product of three integers = ",result)

@dispatch(float,float,float)
def product(first,second,third):
   result  = first * second * third
   print(" Product of three floats = ",result)

product(2,3)
product(2,3,2)
product(2.2,3.4,2.3)
```

Output:

```
Product of two integers =  6
Product of three integers =  12
Product of three floats =  17.204
```

MODULE-9 PROGRAM 9.C

Aim: Write a python script to implement single heritance

Program:

```
class Parent:
        def __init__(self,fname,fage):
                self.firstname=fname
                self.age=fage
        def view(self):
                print(self.firstname , self.age)
class Child(Parent):
        def __init__(self , fname , fage):
                Parent.__init__(self, fname, fage)
                self.lastname="Internet"
        def view(self):
                print("course name" , self.firstname ,"first came",
self.age , " years ago." , self.lastname, " has courses to master
python")
ob = Child("Python" , '28')
ob.view()
```

Output:

```
C:\Users\HP>cd c:\20761A1288

c:\20761A1288>mod_12A.py
course name Python first came 28  years ago. Internet  has courses to master python

c:\20761A1288>
```

MODULE-9 PROGRAM 9.D

Aim: Write a python script to implement method overriding

Program:

```
class Bank:
   def getroi(self):
      return 10
```

```
class SBI(Bank):
    def getroi(self):
        return 7

class ICICI(Bank):
    def getroi(self):
        return 8
class Axis(Bank):
        def display(self):
                print("Axis Bank Rate of interest:",self.getroi())

b1 = Bank()
b2 = SBI()
b3 = ICICI()
b4 = Axis()

print("Bank Rate of interest:",b1.getroi())
print("SBI Rate of interest:",b2.getroi())
print("ICICI Rate of interest:",b3.getroi())
b4.display()
```

Output:

```
c:\20761A1288>m_12D.PY
Bank Rate of interest: 10
SBI Rate of interest: 7
ICICI Rate of interest: 8
Axis Bank Rate of interest: 10

c:\20761A1288>sS
```

MODULE-10 PROGRAM 10.A

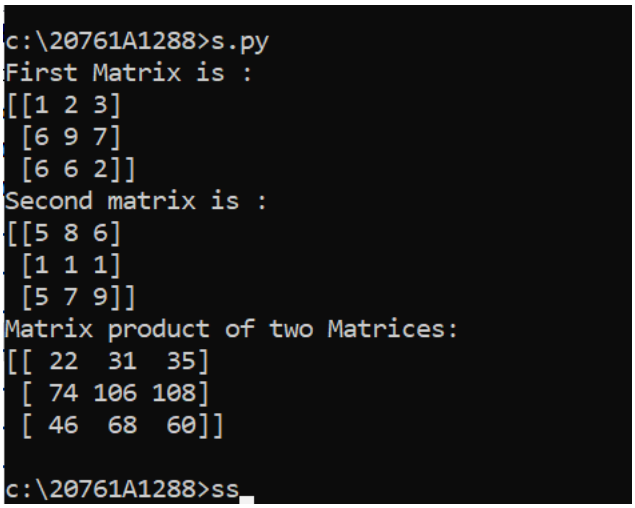
AIM : Write a NumPy program to generate a matrix product of two arrays.

PROGRAM 12.1 :

```
import numpy as np
x =np.random.randint(1,10,(3,3))
```

```
y =np.random.randint(1,10,(3,3))
print("First Matrix is :")
print(x)
print("Second matrix is :")
print(y)
print("Matrix product of two Matrices:")
#print(np.matmul(x, y))
print(x @ y)
```

OUTPUT :



MODULE-10 PROGRAM 10.B

AIM : Write a NumPy program to create a random array with 1000 elements and compute the average, variance, standard deviation of the array elements.

Mean is average of element. Where 0 <= i < n

Mean of arr[0..n-1] = ∑(arr[i]) / n

Variance is the sum of squared differences from the mean divided by a number of elements.

Variance = $\sum(arr[i] - mean)^2 / n$

Standard Deviation is the square root of the variance.

Standard Deviation = $\sqrt{variance}$

Example :

Array = [6 5 3 4 2] sum=20 average or Mean = 20/5 = 4

variance=$((6-4)^2+(5-4)^2+(3-4)^2+(4-4)^2+(4-2)^2)/5$ = (4+1+1+0+4)/5=10/5=2

std=sqrt(2)=1.414

PROGRAM :

```
import numpy as np
x = np.random.randint(1,100,1000)
print(" Elements are : ")
print(x)
print("Average of the array elements:")
mean = x.mean()
print(mean)
print("Standard deviation of the array elements:")
std = x.std()
print(std)
print("Variance of the array elements:")
var = x.var()
print(var)
```

OUTPUT :

```
E:\AY20_21\IT_PYTHON\lab>python Ex13_2.py
 Elements are :
[73 38 91 93 95 39 91  7 73 55 57 47 71 41 35  3 88  6 70 84 13  1 73 59
 30 64 33 97 27 70 81 76 39 42 38 21 26 42 91 19 94 70 42 80 54 23 51 45
 32 21 60 73 34 67  2 45 81  1 16 65 68 62 37 37 87 70 82  5 20  8 81 66
 94 93 33 29 32 80 24  4 51 82 17 32 91 76 44 88  5 18 46 30 35  1 94 55
 83 44 53 95 15 81 23 21 91 81 68 48 98 87 61 81 78 96 12 91 40 97 36 43
 34 29 55 23 72 44 77 71 29 62 24 93 92 63 85 81 82 49  1 42 52 22  6 45
 38 43 92  5 29 58 53 95 81 83  2 43 84 43 46 95 55 20 81 11 87  7 34 78
 47 43 52 68 93 48 92 68 43  4 58 78 27 93 69 30 31 26 37 36  5 61 62  8
 45 13 57 13 67 50 12 15 13 63 37  7 68 66 89 35 41 92 15 20 45 43 95 46
 96  8 34 59 25 90 16 61 14 61 37 22 49 19 46 58 46 14 59 42 53 24 74 10
 31 34 50 53 75 40 18 66 31 18 45 79  1 13 48 56 25 10 29 16 73 43 82 60
 46 61 23 88  3 45 40 41 69 70 57 42 57 72 42 38 52  3 17 82 26 61 54  1
 37 79 61 59 21 83 81  9 10 31 98 81 88 79  8 47 13 13 38 36 54 14 79 49
  1 10  1 98 12 62 75 54 99 57 21  4 44 88 82 24  5 25 14 34 58 14 71 19
 32 74  9  6 70 37 72 81 44  3 59 39 90 35 58  9 41 61  3 71 26 39 87 92
 12 46 36 97 86 99  2 75 13 34  8 65 10 95 93 88 65 38 43 50 77 74 13 87
 77 30  9 58 85 99 28 75 74 67 15  2 90 31 16  6 96 51 78 15 71  2 50 71
 47 64 69 42 66 63 47 10 68 15 77 16 31 43 42 31 53  9 96 50 54 51 21 61
 22 10 18 68 61 22 16 42 19 62 99 71 14 20 59 33 63 26 95 14 40 10 31 35
 83  7  1 80 55 56  9 37 19 23 94 25 55 17 40 19 74 68 89 52 60 34 78 19
 23 58 18 12 97 66 91 49 71  9 43 17 62 63 76 32 81 68 67 18 77 52 33 76
 33  5 58 71 36 99 72 15 24 47 75 48 47 96 54 51 97 53 12 13 85 55 49  3
 38 12  2 23 85 80 10 50 32  4 26 58 92 85 62 56 10 64 38 50  2 68 24 86
 19 82 77 62 87 24 99 13 60  7 18 96 31 26 40 94 29 84 43 88 52  9 67  3
 19 36  2 40 68 21 68 40 98 47 90 27 14 45 20 38 75 89 65 95 69 22 64  3
  9 56 58 91 26 26 62 53 21 13 20  8 23 53 24 33 81 86 72 36 48 19 38 76
 69 85  3  9 62 11 55 54 24 20 87 98 37 45 21 32 57 51 10 40 84 52 19 55
 44 18 86 30 72 21 26  5 24 45 34 22 78 44 53  7 76 68 80  6 74  9 90 67
 98 73 68 68 89 57 16 67 80 21  3 96 96 81 91 52 55 65 36 15 79  4 10 83
 14 94 51 27 32 30 51 93 41 14 98 66 78 51 74 76 52 71 42 53 20 52 68 16
 61 85 24 18  9 33  6 76 46 26 15 57 70 57 49 84 73 90 71 57 32  8 30 85
 75 93 83 78  8 47 58 76 13  2 14 85 55 95 66 65 51 37 40 17 85  6 50 94
 20 32 65 28 50  5 32 22 16  4 62 84 32 78 24 74 69 65  3 50 35 52 83  5
 65 11  3 98 51 83 45 78 58 58 16  3 98 44  1 69 14  7  6 54 98 18 73 19
 53 85 10 53 69 95 60 34 73 82 14 29 70 75 16 95 30 67 18  4 51 40 83 71
 26 84 51 24 36  7 25 27 86 19  5 97 42 61 55 15 69 33 39 63 97 14 68 28
 38  1 27 39 27  1 53 98 31 50  6 43 19  6 87 68  1 47 59 58 46 95 40 74
 28 12 47 42 93 16 16 60 11 59 44 86 85 41  8 66 57 32 25 12 28 31 68 87
 83 13 97 61 29 77 33 19 32 86 25 46 38 47  3 88 92 12 62 49 57 53 18 33
  7 25 17 33 51 94 94 51 92 55 79 44 55 12 77 86 65 42 92 37 89 29 18 58
 46  9 29 95 95 28 55 81 23 31 68 65  7 27 68 63 37 52 62 87  7 98 36 55
 50 49 36 77 62 90 59 56  4 81  1 29  2 22 10 50]
Average of the array elements:
48.155
Standard deviation of the array elements:
28.302313951336206
Variance of the array elements:
801.0209750000001

E:\AY20_21\IT_PYTHON\lab>
```

MODULE-10 PROGRAM 10.C

AIM : Demonstrate how to download dataset and how to create DataFrame

i. Write a Pandas program to get the first 3 rows of a DataFrame
ii. Write a Pandas program to select the specified columns and rows from a given data frame.
iii. Write a Pandas program to select the rows where the score is missing, i.e. is NaN.
iv. Write a Pandas program to insert a new column in existing DataFrame.

PROGRAM :

I. Write a Pandas program to get the first 3 rows of a DataFrame

```
import pandas as pd
import numpy as np

exam_data  = {'name': ['Hema', 'Latha', 'Reddy', 'Jay', 'Madhu'],
                    'score': [90, 70, np.nan, np.nan, 30 ],
                    'attempts': [1,2,1,1,1],
                    'qualify': ['yes', 'yes', 'no', 'no', 'no']}
labels = ['a', 'b', 'c', 'd', 'e']
df = pd.DataFrame(exam_data , index=labels)
print("First three rows of the data frame:")
print(df.iloc[:3])
```

II. Write a Pandas program to select the specified columns and rows from a given data frame.

```
print("Select specific columns and rows:")
print(df.iloc[[1, 3, 5, 6], [1, 3]])
```

III. Write a Pandas program to select the rows where the score is missing, i.e. is NaN.

```
print("Rows where score is missing:")
print(df[df['score'].isnull()])
```

IV. Write a Pandas program to insert a new column in existing DataFrame.

```
color =
['Red','Blue','Orange','Red','White','White','Blue','Green','Green','Red']
df['color'] = color
```

```
print("\nNew DataFrame after inserting the 'color' column")
print(df)
```

OUTPUT :

```
E:\AY20_21\IT_PYTHON\lab>python Ex13_3.py

 First three rows of the data frame:
    name  score  attempts qualify
a   Hema   90.0         1     yes
b  Latha   70.0         2     yes
c  Reddy    NaN         1      no

 Select specific rows and columns:
    name  score
a   Hema   90.0
e  Madhu   30.0

 Rows where score is missing:
    name  score  attempts qualify
c  Reddy    NaN         1      no
d    Jay    NaN         1      no

New DataFrame after inserting the 'color' column
    name  score  attempts qualify   color
a   Hema   90.0         1     yes   Green
b  Latha   70.0         2     yes    Blue
c  Reddy    NaN         1      no  Orange
d    Jay    NaN         1      no     Red
e  Madhu   30.0         1      no   White
```

MODULE-10 PROGRAM 10.D

AIM : Write a Python program to display a bar chart using different color for each bar.

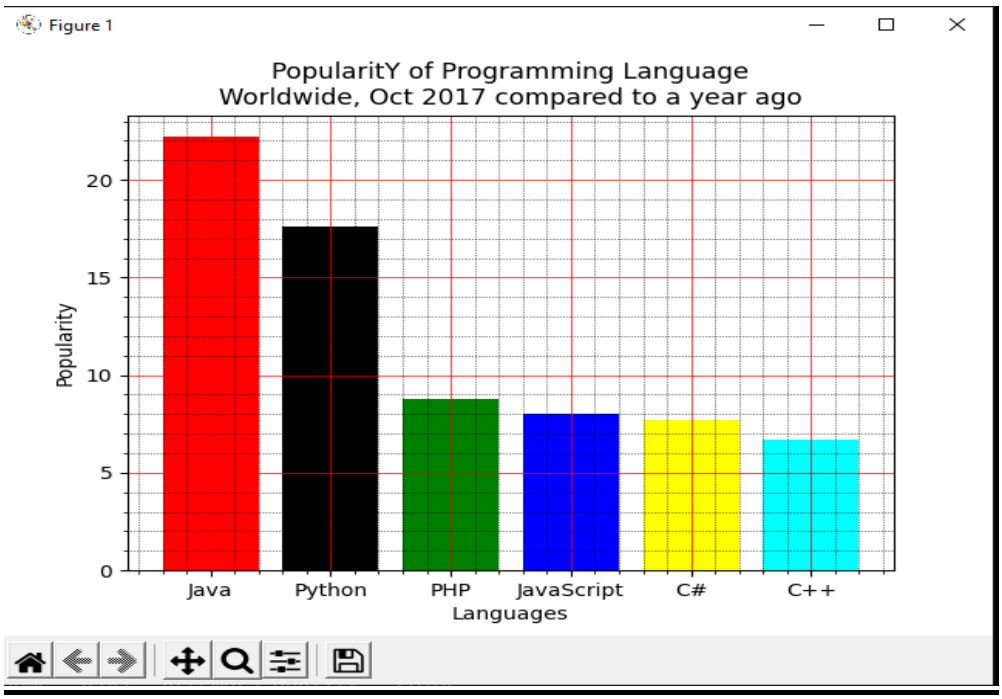
PROGRAM :

```
import matplotlib.pyplot as plt
```

```
x = ['Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++']
popularity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]
x_pos = [i for i, _ in enumerate(x)]
plt.bar(x_pos, popularity, color=['red', 'black', 'green', 'blue', 'yellow',
'cyan'])
plt.xlabel("Languages")
plt.ylabel("Popularity")
plt.title("PopularitY of Programming Language\n" + "Worldwide, Oct
2017 compared to a year ago")
plt.xticks(x_pos, x)
plt.minorticks_on()
plt.grid(which='major', linestyle='-', linewidth='0.5', color='red')
plt.grid(which='minor', linestyle=':', linewidth='0.5', color='black')
plt.show()
```

OUTPUT :

```
E:\AY20_21\IT_PYTHON\lab>python Ex13_4.py
Matplotlib is building the font cache; this may take a moment.
```



MODULE-10 PROGRAM 10.E

AIM : Write a Python programming to create a pie chart with a title.

PROGRAM 12.5 :

```
import matplotlib.pyplot as plt
# Plot data
languages = 'Java', 'Python', 'PHP', 'JavaScript', 'C#', 'C++'
popuratity = [22.2, 17.6, 8.8, 8, 7.7, 6.7]
```

```
#colors = ['red', 'gold', 'yellowgreen', 'blue', 'lightcoral', 'lightskyblue']
colors = ["#1f77b4", "#ff7f0e", "#2ca02c", "#d62728", "#9467bd",
"#8c564b"]
# explode 1st slice
explode = (0, 0.1, 0, 0, 0, 0)
# Plot
plt.pie(popuratity, explode=explode, labels=languages, colors=colors,
autopct='%1.1f%%', shadow=True, startangle=140)
plt.title("PopularitY of Programming Language\n" + "Worldwide, Oct
2017 compared to a year ago", bbox={'facecolor':'0.8', 'pad':5})
plt.show()
```

OUTPUT :

```
E:\AY20_21\IT_PYTHON\lab>python Ex13_5.py
```