

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)TM

HORMIS NAGAR, MOOKKANNOOR, ANGAMALY-683577



FOCUS ON EXCELLENCE

20MCA131 PROGRAMMING LAB

LABORATORY RECORD

Name: ADHEENA JOY

Branch: MASTER OF COMPUTER APPLICATIONS

Semester: 1 Batch: A Roll No: 6

MARCH 2022

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)TM

HORMIS NAGAR, MOOKKANNOOR, ANGAMALY-683577



FOCUS ON EXCELLENCE

CERTIFICATE

*This is to certify that this is a Bonafide record of the Practical work done by
ADHEENA JOY (FIT21MCA-2006) in the 20MCA131 PROGRAMMING LAB
Laboratory towards the partial fulfilment for the award of the Master Of Computer
Applications during the academic year 2021-2022.*

Signature of Staff in Charge

Name:

Signature of H O D

Name:

Date of University practical examination

Signature of
Internal Examiner

Signature of
External Examiner

CONTENT

SI No:	Date :	Name of Experiment:	Page No:	Signature of Staff –In – Charge:
1		Display future leap years from current year to a final year entered by user.	5	
2		List comprehensions: (a) Generate positive list of numbers from a given list of integers (b) Square of N numbers (c) Form a list of vowels selected from a given word (d) List ordinal value of each element of a word (Hint: use ord() to get ordinal values)	6	
3		Count the occurrences of each word in a line of text.	10	
4		Prompt the user for a list of integers. For all values greater than 100, store 'over' instead	11	
5		Store a list of first names. Count the occurrences of 'a' within the list	12	
6		Enter 2 lists of integers. Check (a) Whether list are of same length (b) whether list sums to same value (c) whether any value occur in both.	13	
7		Get a string from an input string where all occurrences of first character replaced with '\$', except first character	15	
8		Create a string from given string where first and last characters exchanged. [eg: python -> nythop]	16	
9		Accept the radius from user and find area of circle.	17	
10		Find biggest of 3 numbers entered.	18	
11		Accept a file name from user and print extension of that.	20	

12		Create a list of colors from comma-separated color names entered by user. Display first and last colors.	21	
13		Accept an integer n and compute n+nn+nnn.	22	
14		Print out all colors from color-list1 not contained in color-list2.	23	
15		Create a single string separated with space from two strings by swapping the character at position 1.	24	
16		Sort dictionary in ascending and descending order.	25	
17		Merge two dictionaries.	26	
18		Find gcd of 2 numbers.	27	
19		From a list of integers, create a list removing even numbers.	28	
20		Program to find the factorial of a number	29	
21		Generate Fibonacci series of N terms	30	
22		Find the sum of all items in a list	31	
23		Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square.	32	
24		Display the given pyramid with step number accepted from user. Eg:N=4 1 2 4 3 6 9 4 8 12 16	34	

25		Count the number of characters (character frequency) in a string.	36	
26		Add 'ing' at the end of a given string. If it already ends with 'ing', then add 'ly'.	38	
27		Accept a list of words and return length of longest word.	39	
28		Construct following pattern using nested loop. *	41	
29		Generate all factors of a number	43	
30		Create a package graphics with modules rectangle, circle and sub-package 3D-graphics with modules cuboid and sphere. Include methods to find area and perimeter of respective figures in each module. Write programs that finds area and perimeter of figures by different importing statements. (Include selective import of modules and import * statements)	44	
31		Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two Rectangle objects by their area	47	

AIM

1.Display future leap years from current year to a final year entered by User.

PROGRAM CODE

```
print("Enter the two years:")

print("Enter the start year:")

startyear=int(input("startyear"))

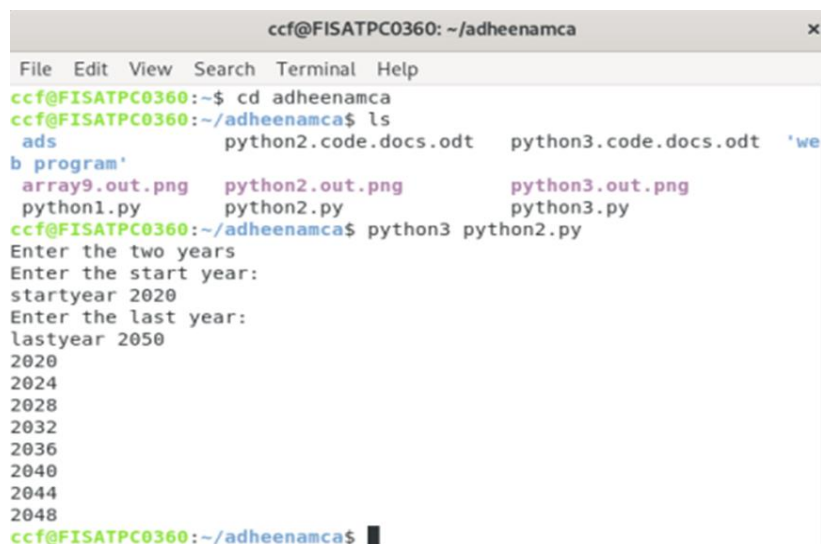
print("Enter the last year:")

lastyear=int(input("lastyear"))

for year in range(startyear,lastyear):

    if (year%4==0) and (year%100!=0) or (year%400==0): print (year)
```

OUTPUT



```
ccf@FISATPC0360: ~/adheenamca
File Edit View Search Terminal Help
ccf@FISATPC0360:~$ cd adheenamca
ccf@FISATPC0360:~/adheenamca$ ls
ads          python2.code.docs.odt  python3.code.docs.odt  'we
b program'
array9.out.png  python2.out.png        python3.out.png
python1.py      python2.py              python3.py
ccf@FISATPC0360:~/adheenamca$ python3 python2.py
Enter the two years
Enter the start year:
startyear 2020
Enter the last year:
lastyear 2050
2020
2024
2028
2032
2036
2040
2044
2048
ccf@FISATPC0360:~/adheenamca$
```

AIM

2.List comprehensions:


- a) Generate positive list of numbers from a given list of integers.
- b) Squares of N numbers
- c) Form a list of vowels selected from a given word
- d) List ordinal value of each element of a word

PROGRAM CODE

a)Generate positive list of numbers from a given list of integers.

```
list1=[12,-1,-2,0,45,67]
for num in list1:
    if (num>=0):
        print(num)
```

OUTPUT

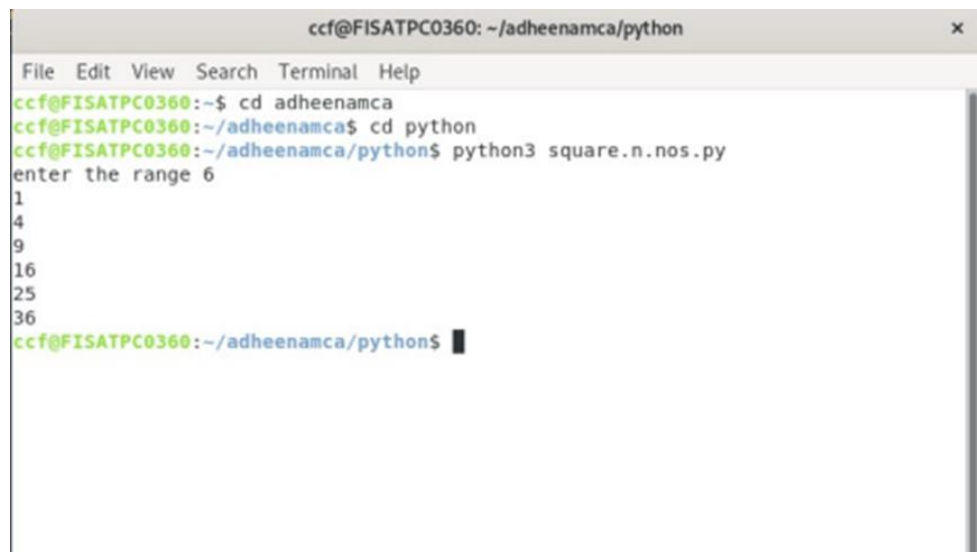


```
ccf@FISATPC0360: ~/adheenamca
File Edit View Search Terminal Help
ccf@FISATPC0360:~$ cd adheenamca
ccf@FISATPC0360:~/adheenamca$ ls
ads          python1.py      python2.out.png  python3.py
array9.out.png  python2.code.docs.odt  python2.py      'web program'
ccf@FISATPC0360:~/adheenamca$ python3 python3.py
12
0
45
67
ccf@FISATPC0360:~/adheenamca$
```


b) Square of N numbers

```
n=int(input("enter the range"))  
for num in range(1,n+1):  
    num=num*num  
    print(num)
```

OUTPUT



```
ccf@FISATPC0360: ~/adheenamca/python  
File Edit View Search Terminal Help  
ccf@FISATPC0360:~$ cd adheenamca  
ccf@FISATPC0360:~/adheenamca$ cd python  
ccf@FISATPC0360:~/adheenamca/python$ python3 square.n.nos.py  
enter the range 6  
1  
4  
9  
16  
25  
36  
ccf@FISATPC0360:~/adheenamca/python$
```

c)Form a list of vowels selected from a given word.

```
s=input("enter any statement:")  
L=[]  
for i in s:  
    if i in "aeiouAEIOU":  
        L.append(i)  
print(L)
```

OUTPUT



```
ccf@FISATPC0360: ~/adheenamca/python  
File Edit View Search Terminal Help  
ccf@FISATPC0360:~$ cd adheenamca  
ccf@FISATPC0360:~/adheenamca$ cd python  
ccf@FISATPC0360:~/adheenamca/python$ python3 list.vowels.py  
enter any statement:water  
['a', 'e']  
ccf@FISATPC0360:~/adheenamca/python$
```

d)List ordinal values of each element of a word.

```
list=['f','i','s','a','t']  
for i in range(0,5):  
    value=ord(list[i])  
    print(value)
```

OUTPUT



The screenshot shows a terminal window titled "ccf@FISATPC0360: ~/adheenamca". The terminal contains the following text:

```
File Edit View Search Terminal Help  
ccf@FISATPC0360:~$ cd adheenamca  
ccf@FISATPC0360:~/adheenamca$ python3 ordinal.value.py  
102  
105  
115  
97  
116  
ccf@FISATPC0360:~/adheenamca$
```

AIM

3.Count the occurrences of each word in a line of text.

PROGRAM CODE

```
list1=[]

list2=[]

x=input("Enter a string:")

for i in x.split(" "):

    list1.append(i)

    if i not in list2:

        list2.append(i)

for i in list2:

    print(i,"\t",list1.count(i))
```

OUTPUT



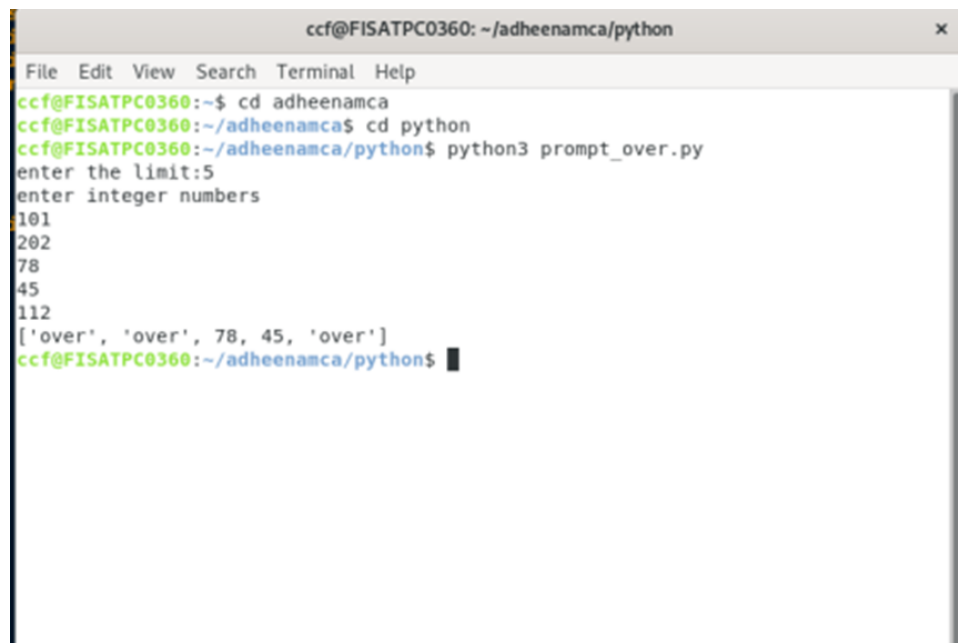
```
stud@debian: ~/adheenamca
stud@debian:~$ cd adheenamca
stud@debian:~/adheenamca$ python3 program4_c01.py
Enter a string:sweet sweet sugary pumpkin
sweet      2
sugary     1
pumpkin    1
stud@debian:~/adheenamca$
```

AIM

4. Prompt the user for a list of integers. For all values greater than 100, store 'over' instead.

PROGRAM CODE

```
list=[]  
n=int(input("enter the limit:"))  
print("enter integer numbers")  
for i in range(0,n):  
j=int(input())  
if(j> 100):  
list.append("over")  
else:  
list.append(j)  
print(list)
```

OUTPUT

```
ccf@FISATPC0360: ~/adheenamca/python  
File Edit View Search Terminal Help  
ccf@FISATPC0360:~$ cd adheenamca  
ccf@FISATPC0360:~/adheenamca$ cd python  
ccf@FISATPC0360:~/adheenamca/python$ python3 prompt_over.py  
enter the limit:5  
enter integer numbers  
101  
202  
78  
45  
112  
['over', 'over', 78, 45, 'over']  
ccf@FISATPC0360:~/adheenamca/python$
```

AIM

5.Store a list of first names. Count the occurrences of 'a' within the list

PROGRAM CODE

```
list=["anna","deepa","lali"]  
count=0  
for word in list:  
    for i in word:  
        if (i=='a'):  
            count+=1  
print("count of a is",count)
```

OUTPUT

A screenshot of a terminal window titled "ccf@FISATPC0360: ~/adheenamca". The terminal shows the following commands and output:
ccf@FISATPC0360:~\$ cd adheenamca
ccf@FISATPC0360:~/adheenamca\$ python3 count.a.py
count of a is 4
ccf@FISATPC0360:~/adheenamca\$
The terminal window has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help".

```
ccf@FISATPC0360: ~/adheenamca  
File Edit View Search Terminal Help  
ccf@FISATPC0360:~$ cd adheenamca  
ccf@FISATPC0360:~/adheenamca$ python3 count.a.py  
count of a is 4  
ccf@FISATPC0360:~/adheenamca$
```

AIM

6. Enter 2 lists of integers. Check

- a. whether list are of same length
- b. whether list sums of same value
- c. whether any value occur in both.

PROGRAM CODE

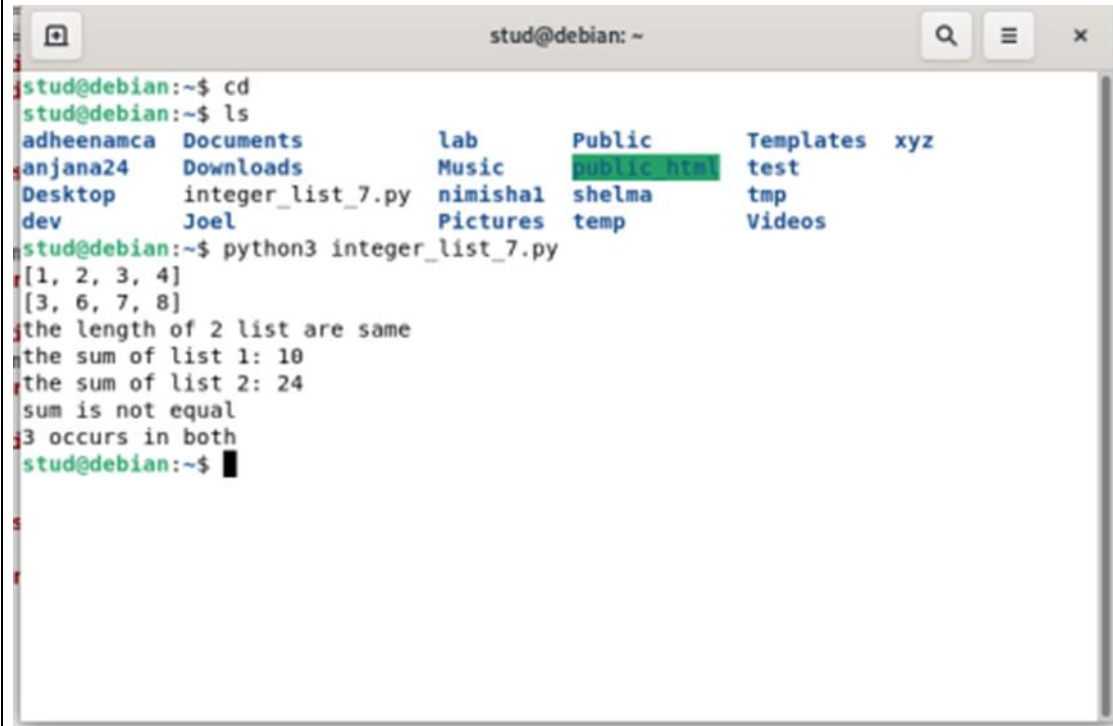
```

l1=[1,2,3,4]
l2=[5,6,7,8]
print(l1)
print(l2)
if len(l1)==len(l2):
    print("the length of 2 list are same") else:
    print("the length of 2 list are not same")
sum1=0
for i in range(len(l1)):
    sum1=sum1+l1[i]
print("the sum of list 1:",sum1)
sum2=0
for j in range(len(l2)):
    sum2=sum2+l2[j]
print("the sum of list 2:",sum2)
if sum1==sum2:
    print("sum is equal")
else:
    print("sum is not equal")
flag=0
for i in l1:
    if i in l2:

```

```
print(i,"occurs in both")  
  
flag=1  
  
if(flag==0):  
  
print("there is no common")
```

OUTPUT



The screenshot shows a terminal window titled 'stud@debian: ~'. The user enters the following commands and receives the following output:

```
stud@debian:~$ cd  
stud@debian:~$ ls  
adheenamca Documents lab Public Templates xyz  
anjana24 Downloads Music public.html test  
Desktop integer_list_7.py nimishal shelma tmp  
dev Joel Pictures temp Videos  
stud@debian:~$ python3 integer_list_7.py  
[1, 2, 3, 4]  
[3, 6, 7, 8]  
the length of 2 list are same  
the sum of list 1: 10  
the sum of list 2: 24  
sum is not equal  
3 occurs in both  
stud@debian:~$
```

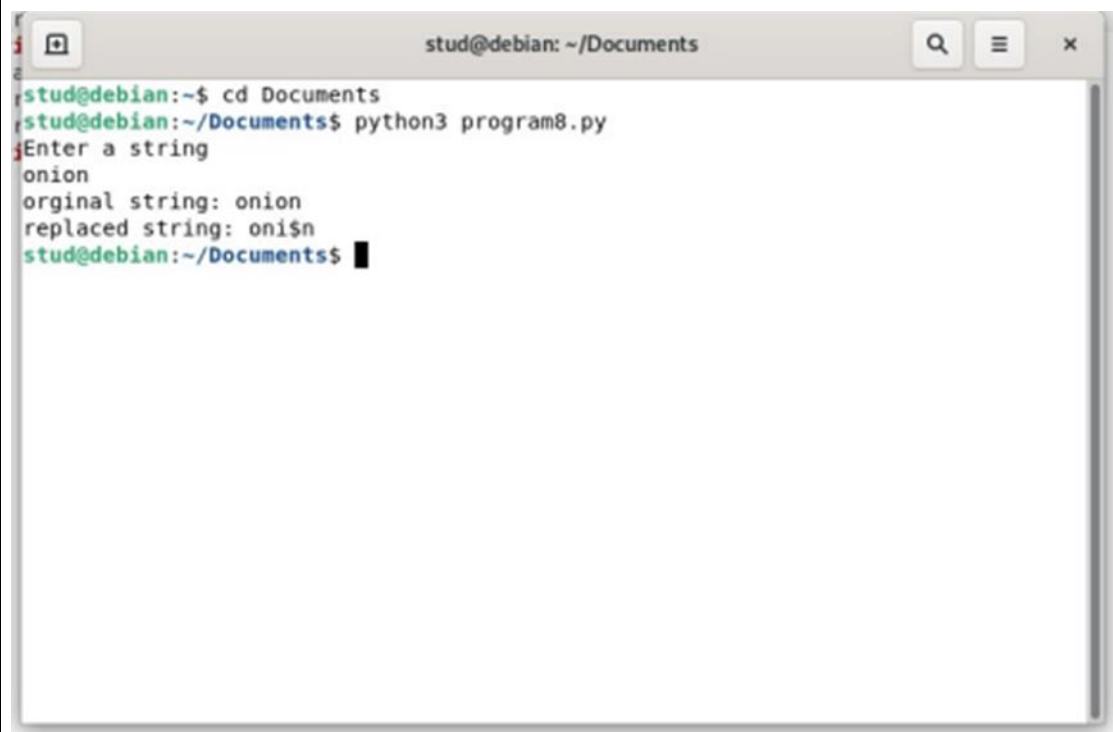

AIM

7. Get a string from an input string where all occurrences of first character replaced with '\$', except first character. [eg: onion -> oni\$n]

PROGRAM CODE

```
str1=input("Enter a string \n")
print("original string:",str1)
char=str1[0]
str1=str1.replace(char,'$')
str1=char+str1[1:]
print("replaced string:",str1)
```

OUTPUT

A screenshot of a terminal window titled 'stud@debian: ~/Documents'. The terminal shows the following sequence of commands and output:

```
stud@debian:~$ cd Documents
stud@debian:~/Documents$ python3 program8.py
Enter a string
onion
original string: onion
replaced string: oni$n
stud@debian:~/Documents$
```

AIM


8.Create a string from given string where first and last characters exchanged.

[eg:python->nythop]

PROGRAM CODE

```
s="paris"  
t=s[0]  
t1=s[-1]  
n=len(s)  
ns=t1+s[1:n-1]+t  
print(ns)
```

OUTPUT

A screenshot of a terminal window titled 'stud@debian: ~/Documents'. The terminal shows the following commands and output:
stud@debian:~\$ cd Documents
stud@debian:~/Documents\$ python3 program9_python.py
sarip
stud@debian:~/Documents\$
The output 'sarip' is the result of the Python program, which swaps the first and last characters of the string 'paris'.

AIM

9. Accept the radius from the user and find the area of the circle.

PROGRAM CODE

```
X=input("enter the radius :")
```

```
Y=int(x)
```

```
A=3.14*y*y
```

```
Print("Area is:")
```

OUTPUT

```
stud@debian:~$ cd adheena
stud@debian:~/adheena$ ls
areacircle.py  factorial.py  fibonacci.png  fibonacci.py  largestthreenumber.py
stud@debian:~/adheena$ python3 areacircle.py
enter the radius 6
113.03999999999999
stud@debian:~/adheena$ █
```

AIM

10.Find the biggest of 3 numbers

PROGRAM CODE

```
X=input("enter the first number:") Y=input("enter the second number:")
Z=input("enter the third number:") a=int(x)
b=int(y)
c=int(z)
if a>b:
if a>c:
print(a)
else:
print(c)
else:
if b>c:
print(b)
else:
print(c)
```

OUTPUT

```
stud@debian:~$ cd adheena
stud@debian:~/adheena$ ls
areacircle.py  areaofcircle.png  factorial.png  factorial.py  fibonacci.png  fibonacci.py  largestthreenumber.py
stud@debian:~/adheena$ python3 largestthreenumber.py
enter the first number 10
enter the second number 70
enter the third number 50
70
stud@debian:~/adheena$
```

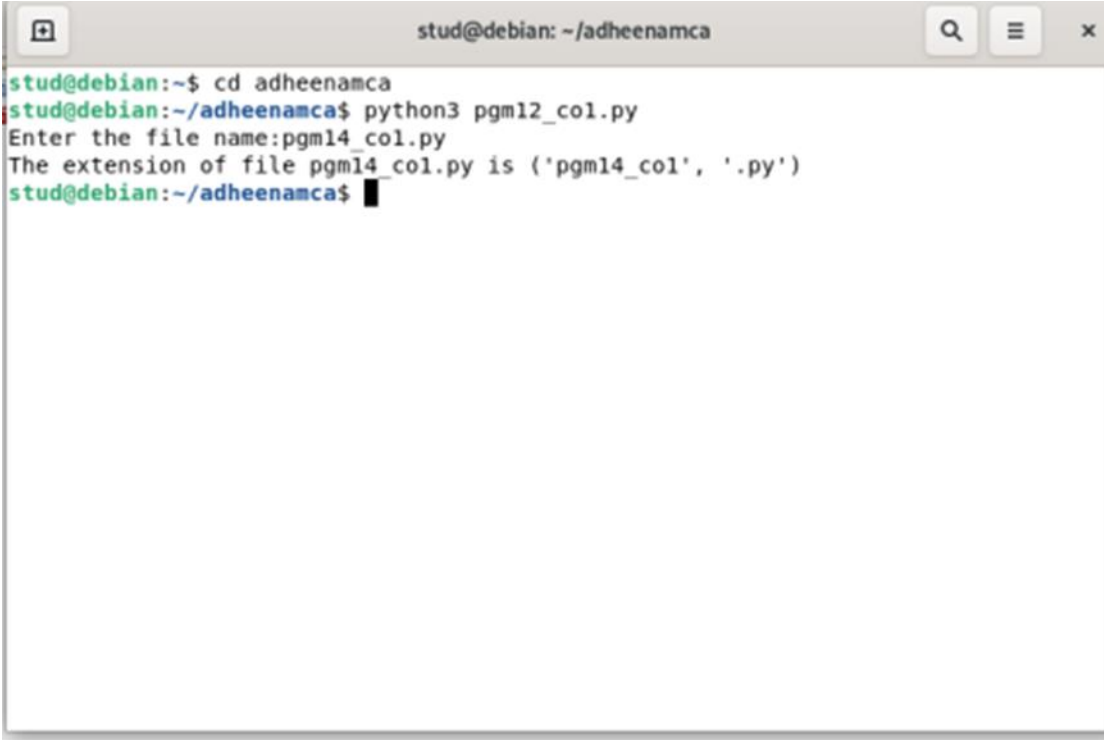
AIM

11. Accept a file name from user and print extension of that.

PROGRAM CODE

```
import os  
  
a=input("Enter the file name:")  
  
print("The extension of file",a,"is",os.path.splitext(a))
```

OUTPUT



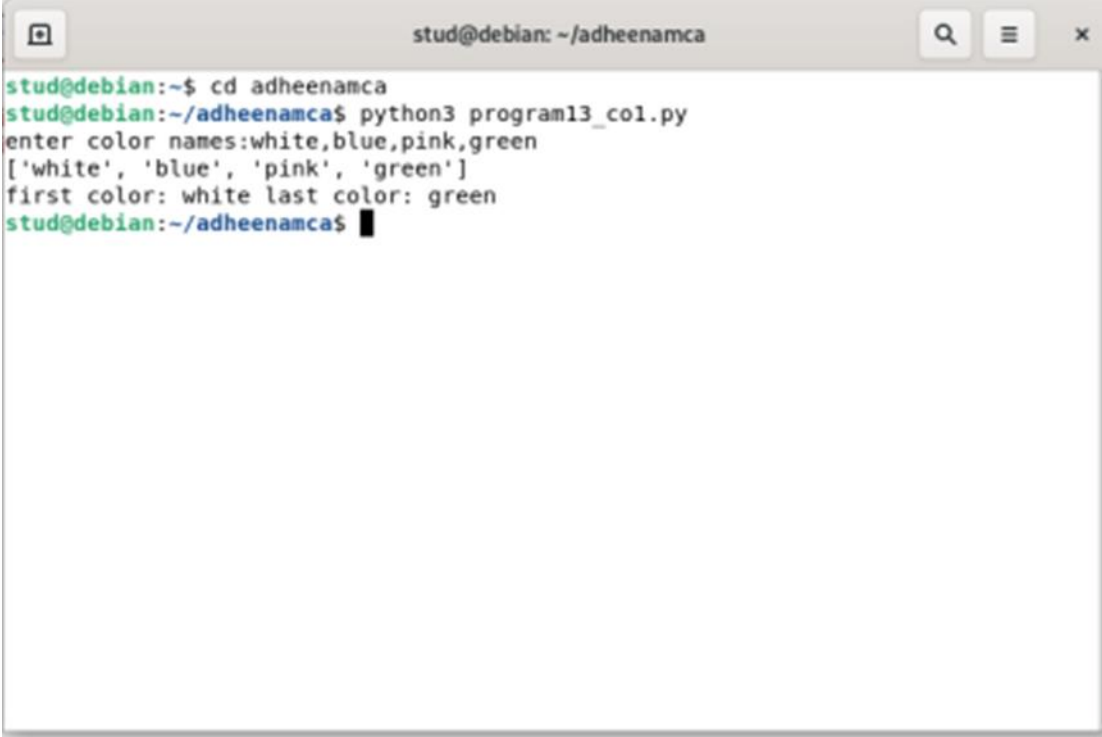
```
stud@debian: ~/adheenamca  
stud@debian:~$ cd adheenamca  
stud@debian:~/adheenamca$ python3 pgml2_col.py  
Enter the file name:pgml4_col.py  
The extension of file pgml4_col.py is ('pgml4_col', '.py')  
stud@debian:~/adheenamca$
```

AIM

12.Create a list of colors from comma-separated color names entered by user. Display first and last colors.

PROGRAM CODE

```
Colors=[]
str=(input("enter color names:"))
for I in str.split(','):
    colors.append(i)
print(colors)
print("first color:",colors[0],"last
color:",colors[-1])
```

OUTPUT

```
stud@debian: ~/adheenamca
stud@debian:~$ cd adheenamca
stud@debian:~/adheenamca$ python3 program13_col.py
enter color names:white,blue,pink,green
['white', 'blue', 'pink', 'green']
first color: white last color: green
stud@debian:~/adheenamca$
```

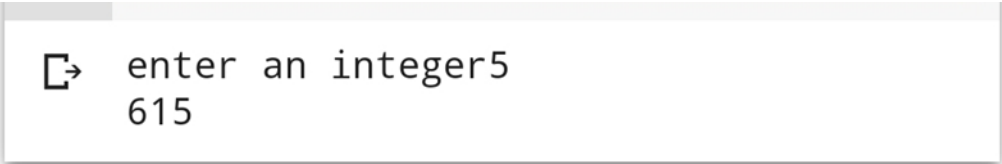
AIM

13. Accept an integer n and compute $n+nn+nnn$.

PROGRAM CODE

```
x=int(input("Enter an integer"))  
n1=str(x)  
n2=n1+n1  
n3=n2+n1  
result=int(n1)+int(n2)+int(n3)  
print(result)
```

OUTPUT



enter an integer5
615

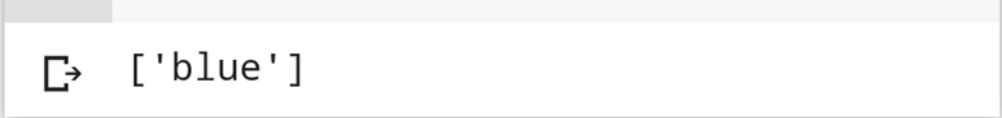
AIM

14. Print out all color from color-list1 not contained in color-list2

PROGRAM CODE

```
l1=["red","orange","pink","blue"]  
l2=["pink","orange","red"]  
l3=[]  
for i in l1:  
    if i not in l2:  
        l3.append(i)  
print(l3)
```

OUTPUT



```
➞ ['blue']
```

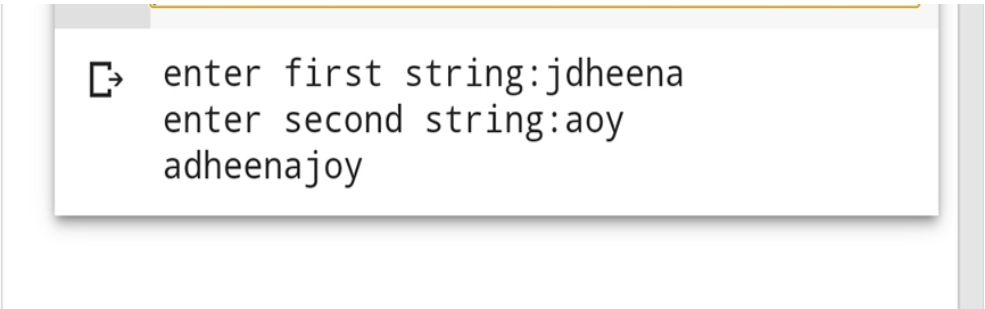
AIM

15.Create a single string separated with space from two strings by swapping the character at position 1.

PROGRAM CODE

```
str1=input("Enter first string:")  
str2=input("Enter second string:")  
str3=str2[0]+str1[1:]+ " "+str1[0]+str2[1:]  
print(str3)
```

OUTPUT



```
enter first string:jdheena  
enter second string:aoy  
adheenajoy
```

AIM

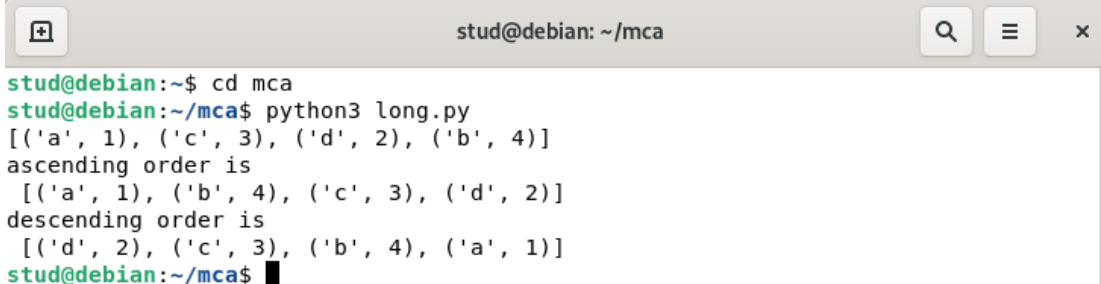
16.Sort dictionary in ascending and descending order.

PROGRAM CODE

```
dict1={"a":1,"c":3,"d":2,"b":4}
l=list(dict1.items())
print(l)
l.sort()
print("ascending order is\n",l)
l=list(dict1.items())
l.sort(reverse=True)

print("descending order is\n",l)
```

OUTPUT



The screenshot shows a terminal window titled 'stud@debian: ~/mca'. The user enters the command 'cd mca' and then 'python3 long.py'. The script outputs the initial dictionary items as a list of tuples: [('a', 1), ('c', 3), ('d', 2), ('b', 4)]. It then prints 'ascending order is' followed by the sorted list: [('a', 1), ('b', 4), ('c', 3), ('d', 2)]. Next, it prints 'descending order is' followed by the reverse-sorted list: [('d', 2), ('c', 3), ('b', 4), ('a', 1)]. The prompt returns to 'stud@debian: ~/mca\$'.

```
stud@debian:~$ cd mca
stud@debian:~/mca$ python3 long.py
[('a', 1), ('c', 3), ('d', 2), ('b', 4)]
ascending order is
[('a', 1), ('b', 4), ('c', 3), ('d', 2)]
descending order is
[('d', 2), ('c', 3), ('b', 4), ('a', 1)]
stud@debian:~/mca$
```

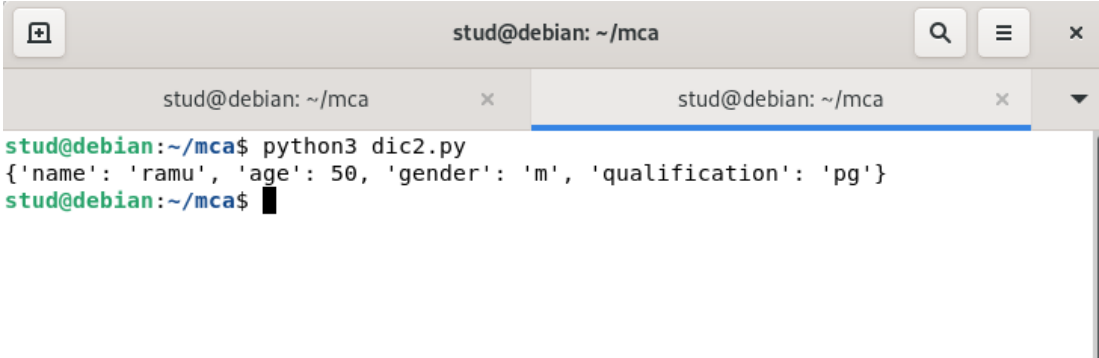
AIM

17.Merge two dictionaries

PROGRAM CODE

```
dict1={"name":"ramu","age":50}  
dict2={"gender":"m","qualification":"pg"}  
dict1.update(dict2)  
print(dict1)
```

OUTPUT



The screenshot shows a terminal window titled 'stud@debian: ~/mca'. It contains two tabs, both with the same title. The active tab shows the following commands and output:

```
stud@debian:~/mca$ python3 dic2.py  
{'name': 'ramu', 'age': 50, 'gender': 'm', 'qualification': 'pg'}  
stud@debian:~/mca$
```

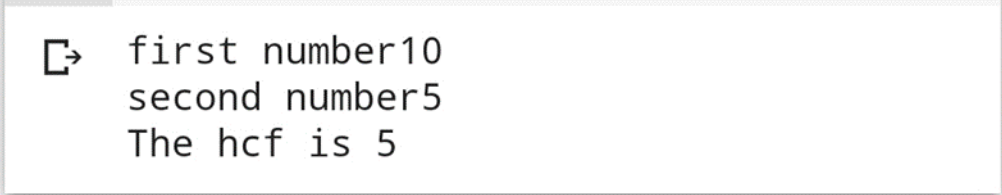
AIM

18.Find gcd of 2 numbers

PROGRAM CODE

```
x=int(input("first number"))
y=int(input("second number"))
if(x>y):
    smaller=y
else:
    smaller=x
for i in range(1,smaller+1):
    if((x%i==0)and(y%i==0)):
        hcf=i
print("The HCF is",hcf)
```

OUTPUT

A screenshot of a terminal window with a light gray background. It shows the execution of a Python program. The first line is a prompt character followed by 'first number' and the input '10'. The second line is a prompt character followed by 'second number' and the input '5'. The third line shows the output 'The hcf is 5'.

```
➞ first number10
second number5
The hcf is 5
```

AIM

19.From a list of integers, create a list removing even numbers.

PROGRAM CODE

```
l1=[1,2,3,4,5,6]
```

```
l2=[]
```

```
for i in l1:
```

```
    If(i%2!=0):
```

```
        l2.append(i)
```

```
print(l2)
```

OUTPUT

```
[1, 3, 5]
```

AIM

20. Program to find the factorial of a number.

PROGRAM CODE

```
n=int(input("Enter a number:"))  
fact=1  
for i in range(1,n+1):  
    fact=fact*i  
print(fact)
```

OUTPUT

```
stud@debian:~$ cd adheena  
stud@debian:~/adheena$ ls  
areacircle.py  factorial.png  factorial.py  fibonacci.png  fibonacci.py  largestthreenumber.py  
stud@debian:~/adheena$ python3 factorial.py  
enter the number 6  
720  
stud@debian:~/adheena$
```

AIM

21. Generate fibonacci series of N terms.

PROGRAM CODE

```
n=int(input("Enter a number:"))
f1=0
f2=1
print(f1)
print(f2)
for i in range(0,n-2):
    f3=f1+f2
    print(f3)
    f1=f2
    f2=f3
```

OUTPUT

```
stud@debian:~$ cd adheena
stud@debian:~/adheena$ ls
areacircle.py  areaofcircle.png  factorial.png  factorial.py  fibonacci.png  fibonacci.py  largestthreenumber.py
stud@debian:~/adheena$ python3 fibonacci.py
enter the number 7
0
1
1
2
3
5
8
13
21
stud@debian:~/adheena$
```


AIM

22.Find the sum of all items in a list.

PROGRAM CODE

```
list1=[1,2,3,4,5]
```

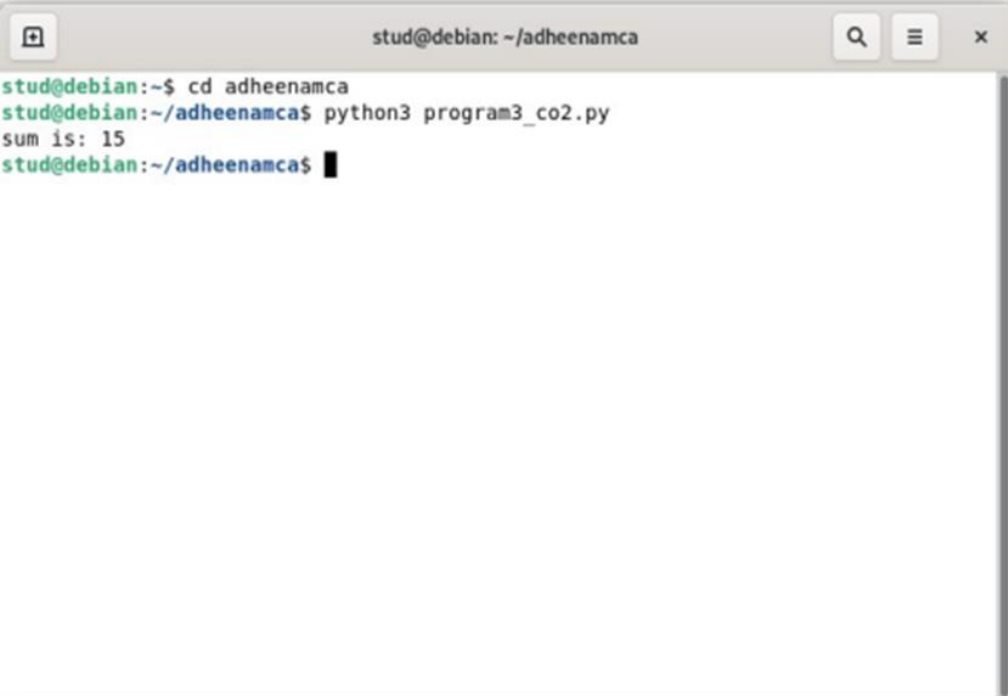
```
sum=0
```

```
for i in list1:
```

```
    sum=sum+i
```

```
print(sum)
```

OUTPUT

A screenshot of a terminal window titled 'stud@debian: ~/adheenamca'. The terminal shows the following commands and output:

```
stud@debian:~$ cd adheenamca
stud@debian:~/adheenamca$ python3 program3_co2.py
sum is: 15
stud@debian:~/adheenamca$
```

AIM

23. Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square.

PROGRAM CODE

```
limit1=1000

limit2=9999

list1=[]

for i in range(limit1,limit2):

    j=i

    digit=[]

    while(i!=0):

        digit.append(i%10)

        i=int(i/10)

        count=0

        for n in digit:

            if n%2==0:

                count=count+1

            if count==4:

                for k in range(31,100):

                    if((k**2)==j):

                        list1.append(j)
```

```
print(k)
```

```
print(list1)
```

OUTPUT

A screenshot of a terminal window titled 'stud@debian: ~/adheenamca'. The terminal shows the following commands and output:

```
stud@debian:~$ cd adheenamca
stud@debian:~/adheenamca$ python3 pgm4_co2.py
68
78
80
92
[4624, 6084, 6400, 8464]
stud@debian:~/adheenamca$
```

AIM

24.Display the given pyramid with step number accepted from user

Eg: N=4

```
1
2 4
3 6 9
4 8 12 16
```

PROGRAM CODE

```
n=int(input("Enter a number:"))

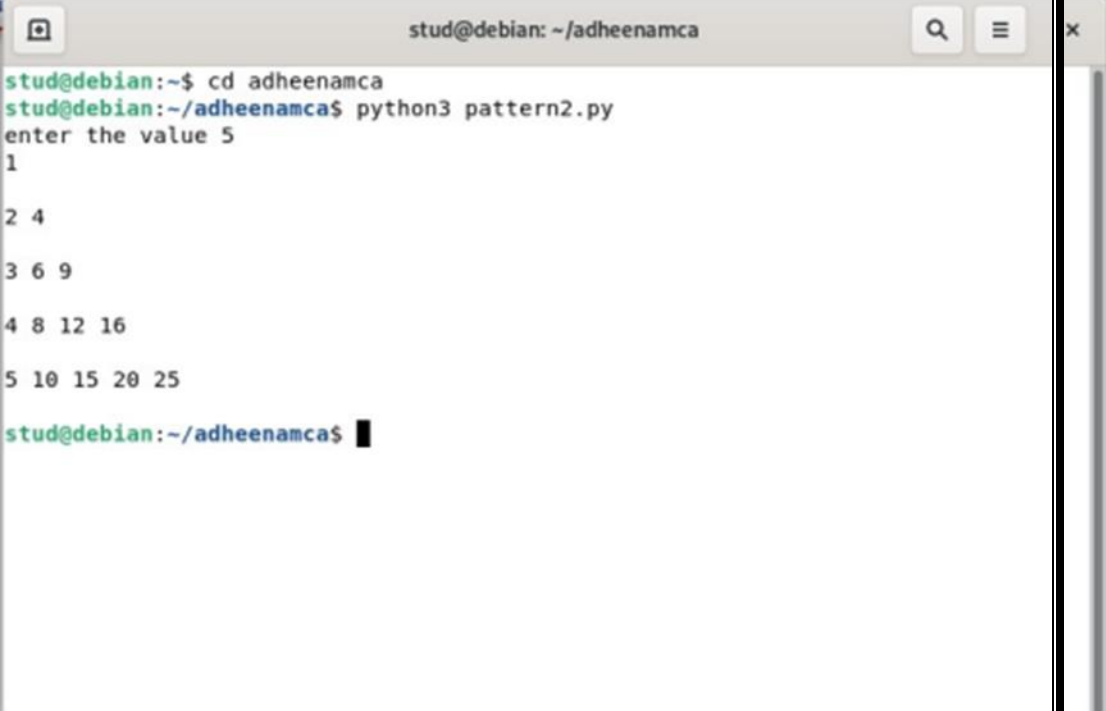
for i in range(1,n+1):

    for j in range(i,(i*i)+1,i):

        print(j,"\\t",end="")

    print("\\n")
```

OUTPUT



```
stud@debian: ~/adheenamca
stud@debian:~$ cd adheenamca
stud@debian:~/adheenamca$ python3 pattern2.py
enter the value 5
1
2 4
3 6 9
4 8 12 16
5 10 15 20 25
stud@debian:~/adheenamca$
```

The image shows a terminal window titled 'stud@debian: ~/adheenamca'. The user enters the command 'cd adheenamca' to change the directory. Then, they run 'python3 pattern2.py'. The program prompts 'enter the value 5'. The user enters '5', and the program outputs a pattern of numbers: '1', '2 4', '3 6 9', '4 8 12 16', and '5 10 15 20 25'. The terminal ends with the prompt 'stud@debian:~/adheenamca\$'.

AIM

25.Count the number of characters (character frequency) in a string.

PROGRAM CODE

```
string=input("Enter a string:")
```

```
list1=[]
```

```
for i in string:
```

```
    if i not in list1:
```

```
        list1.append(i)
```

```
for i in list1:
```

```
    count=0
```

```
    for j in string:
```

```
        if(i==j):
```

```
            count=count+1
```

```
    print(i,"\t:",count)
```

OUTPUT



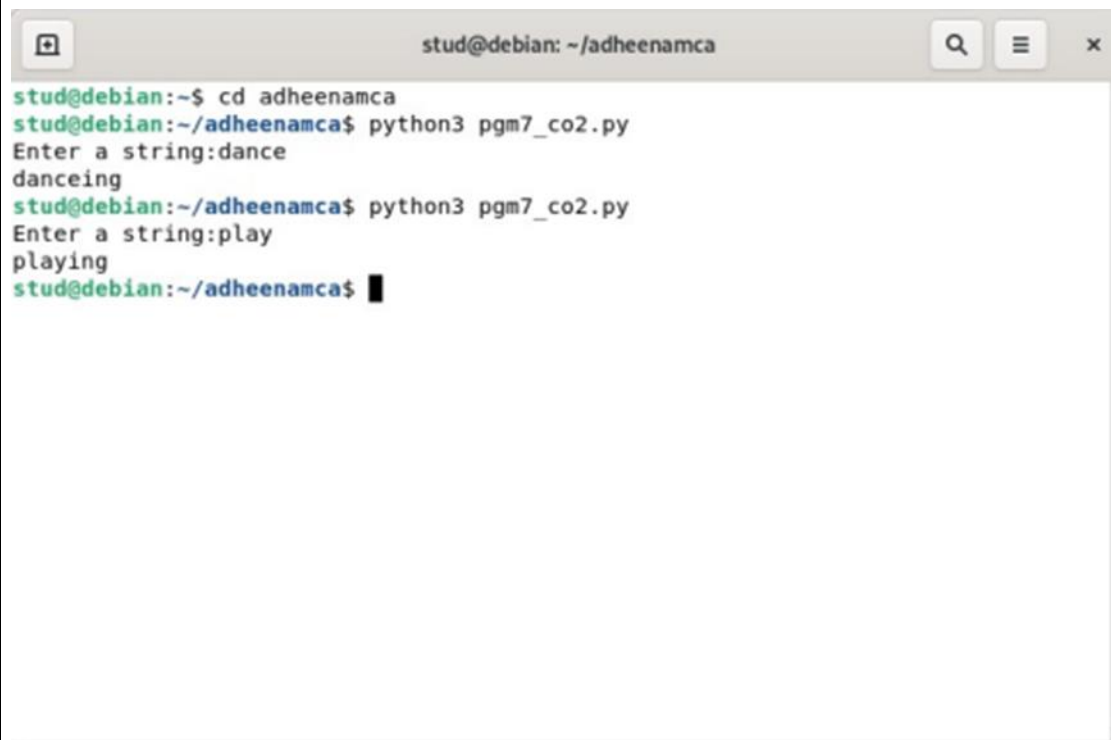
```
stud@debian: ~/adheenamca
stud@debian:~$ cd adheenamca
stud@debian:~/adheenamca$ python3 pgm6_co2.py
Enter a string:adheenajoy
a      : 2
d      : 1
h      : 1
e      : 2
n      : 1
j      : 1
o      : 1
y      : 1
stud@debian:~/adheenamca$
```

AIM

26.Add 'ing' at the end of a given string. If it already ends with 'ing', then add 'ly'.

PROGRAM CODE

```
string=input("Enter a string:")  
  
if(string[-3:]=="ing"):  
  
    string+="ly"  
  
else:  
  
    string+="ing"  
  
print(string)
```

OUTPUT

```
stud@debian: ~/adheenamca  
stud@debian:~$ cd adheenamca  
stud@debian:~/adheenamca$ python3 pgm7_co2.py  
Enter a string:dance  
danceing  
stud@debian:~/adheenamca$ python3 pgm7_co2.py  
Enter a string:play  
playing  
stud@debian:~/adheenamca$
```

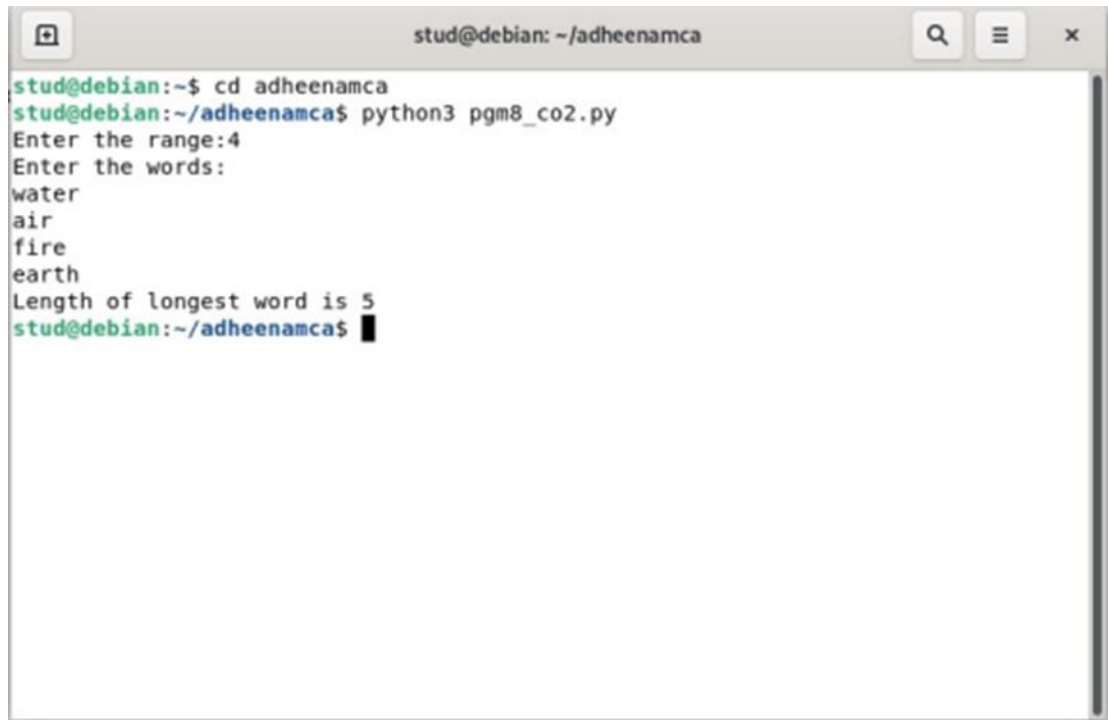

AIM

27. Accept a list of words and return length of longest word.

PROGRAM CODE

```
lis=[]  
  
n=int(input("Enter the range:"))  
  
print("Enter the words:")  
  
for i in range(0,n):  
  
    lis.append(input(""))  
  
longest=lis[0]  
  
for i in range(1,n):  
  
    if(len(lis[i])>len(longest)):  
  
        longest=lis[i]  
  
print("Length of longest word is",len(longest))
```

OUTPUT



A terminal window titled "stud@debian: ~/adheenamca" with search, menu, and close buttons. The terminal shows the following commands and output:

```
stud@debian:~$ cd adheenamca
stud@debian:~/adheenamca$ python3 pgm8_co2.py
Enter the range:4
Enter the words:
water
air
fire
earth
Length of longest word is 5
stud@debian:~/adheenamca$
```

AIM

28. Construct following pattern using nested loop.

```
*  
* *  
* * *  
* * * *  
* * * * *  
  
* * * *  
  
* * *  
  
* *  
  
*
```

PROGRAM CODE

```
k='*'  
for i in range(1,6):  
    for j in range(1,i+1):  
        print(k,end=" ")  
    print("\n")  
for i in range(4,0,-1):  
    for j in range(1,i+1):  
        print(k,end=" ")  
    print("\n")
```

OUTPUT



```
stud@debian: ~/adheenamca
stud@debian:~$ cd adheenamca
stud@debian:~/adheenamca$ python3 program9_co2.py

*
* *
* * *
* * * *
* * * * *
* * * *
* * *
* *
*

stud@debian:~/adheenamca$
```

AIM

29.Generate all factors of a number.

PROGRAM CODE

```
n=int(input("Enter a number:"))  
print("Factors are")  
for i in range(1,n+1):  
    if(n%i==0):  
        print(i)
```

OUTPUT



```
stud@debian: ~/adheenamca  
stud@debian:~$ cd adheenamca  
stud@debian:~/adheenamca$ python3 pgm10_co2.py  
Enter a number:12  
Factors are  
1  
2  
3  
4  
6  
12  
stud@debian:~/adheenamca$
```

AIM

30. Create a package graphics with modules rectangle, circle and sub-package 3D-graphics with modules cuboid and sphere. Include methods to find area and perimeter of respective figures in each module. Write programs that find area and perimeter of figures by different importing statements. (Include selective import of modules and import * statements)

PROGRAM CODE**graphics\circle.py**

```
from math import pi
def area_circle(radius):
    return pi*radius*radius
def perimeter_circle(radius):
    return 2*pi*radius
```

graphics\rectangle.py

```
def area_rec(length,width):
    return length*width
def perimeter_rec(length,width):
    return 2*(length+width)
```

graphics\tdgraphics\cuboid.py

```
def area_cuboid(l,b,h):
    return 2*(l*h + b*h + l*b)
def volume_cuboid(l,b,h):
    return l*b*h
```

graphics\tdgraphics\sphere.py

```
from math import pi
def area_sphere(radius):
    return 4*(pi*radius*radius)
def perimeter_sphere(radius):
```

```
return 2*pi*radius
```

graphics.py (driver code)

```
import graphics

from graphics import circle,rectangle
from graphics.tdgraphics import cuboid,sphere
from graphics.circle import *

print("Area of a circle with radius 10 is :",circle.area_circle(10))
print("Perimeter of a circle with radius 10 is ",circle.perimeter_circle(10))
print("\n")

print("Area of a Rectangle with length and width 10 is :
      ",rectangle.area_rec(10,10))

print("Perimeter of a Rectangle with length and width 10 is :
      ",rectangle.perimeter_rec(10,10))
print("\n")

print("Area of a cuboid with length,width,height 10 is :
      ",cuboid.area_cuboid(10,10,10))

print("Volume of a cuboid with length,width,height 10 is :
      ",cuboid.volume_cuboid(10,10,10))
print("\n")

print("Area of a sphere with radius 10 is :",sphere.area_sphere(10))
print("Perimeter of a sphere with radius 10 is ",sphere.perimeter_sphere(10))
```

OUTPUT

```
Command Prompt
Microsoft Windows [Version 10.0.18362.1316]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\adhee>cd OneDrive
C:\Users\adhee\OneDrive>cd desktop
C:\Users\adhee\OneDrive\Desktop>cd python
C:\Users\adhee\OneDrive\Desktop\python>md graphics
C:\Users\adhee\OneDrive\Desktop\python>cd graphics
C:\Users\adhee\OneDrive\Desktop\python\graphics>notepad circle.py
C:\Users\adhee\OneDrive\Desktop\python\graphics>notepad rectangle.py
C:\Users\adhee\OneDrive\Desktop\python\graphics>md tdgraphics
C:\Users\adhee\OneDrive\Desktop\python\graphics>cd tdgraphics
C:\Users\adhee\OneDrive\Desktop\python\graphics\tdgraphics>notepad cuboid.py
C:\Users\adhee\OneDrive\Desktop\python\graphics\tdgraphics>notepad sphere.py
C:\Users\adhee\OneDrive\Desktop\python\graphics\tdgraphics>cd ..
C:\Users\adhee\OneDrive\Desktop\python\graphics>cd ..
C:\Users\adhee\OneDrive\Desktop\python>notepad driver1.py
```

```
Command Prompt

C:\Users\adhee\OneDrive\Desktop\python>python driver1.py
Area of a circle with radius 10 is : 314.1592653589793
Perimeter of a circle with radius 10 is 62.83185307179586

Area of a Rectangle with length and width 10 is : 100
Perimeter of a Rectangle with length and width 10 is : 40

Area of a cuboid with length,width,height 10 is : 600
Volume of a cuboid with length,width,height 10 is : 1000

Area of a sphere with radius 10 is : 1256.6370614359173
Perimeter of a sphere with radius 10 is 62.83185307179586

C:\Users\adhee\OneDrive\Desktop\python>
```


AIM

31. Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two Rectangle objects by their area.

PROGRAM CODE

```
class Rectangle:

    def __init__(self,length,breadth):

        self.length=length

        self.breadth=breadth

    def area(self):

        return self.length*self.breadth

    def perimeter(self):

        return 2*(self.length+self.breadth)

r1=Rectangle(10,2)

r2=Rectangle(5,9)

x=r1.area()

y=r2.area()

z=r1.perimeter()

w=r2.perimeter()

print("Area of rectangle 1:",x);

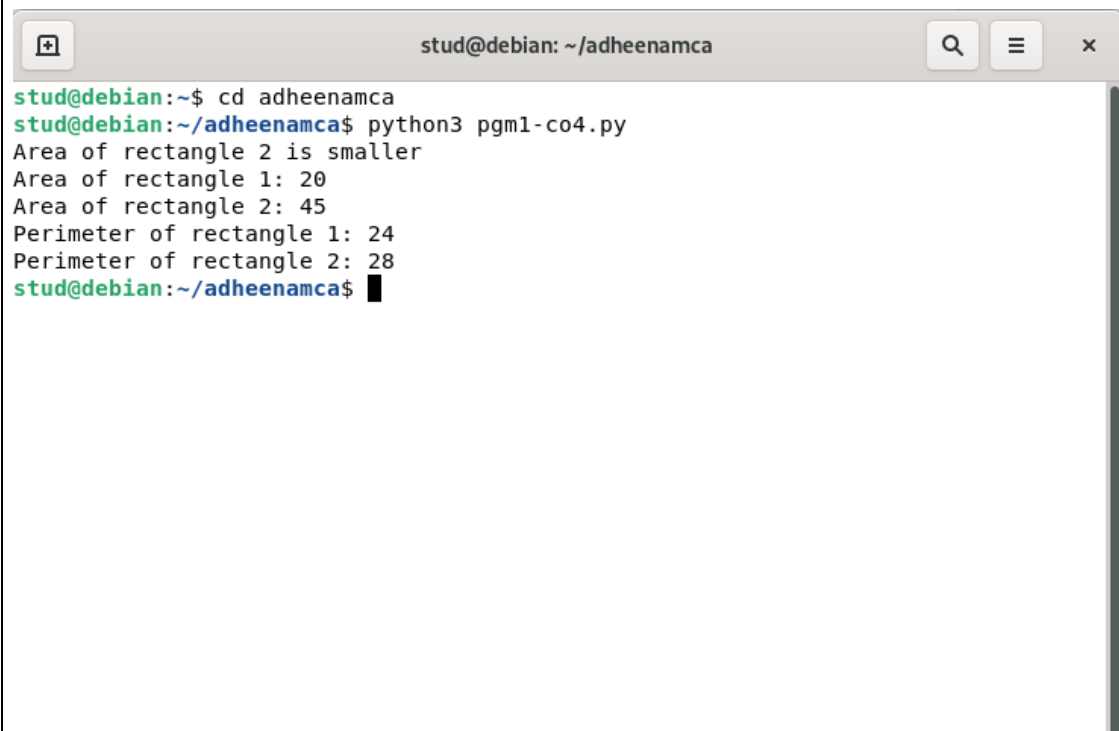
print("Area of rectangle 2:",y);

print("Perimeter of rectangle 1:",z);

print("Perimeter of rectangle 2:",w);
```

```
if(x>y):  
  
    print("Area of rectangle 1 is larger")  
  
else:  
  
    print("Area of rectangle 2 is smaller")
```

OUTPUT

A terminal window titled 'stud@debian: ~/adheenamca' with search, menu, and close buttons. The terminal shows the following commands and output:

```
stud@debian:~$ cd adheenamca  
stud@debian:~/adheenamca$ python3 pgm1-co4.py  
Area of rectangle 2 is smaller  
Area of rectangle 1: 20  
Area of rectangle 2: 45  
Perimeter of rectangle 1: 24  
Perimeter of rectangle 2: 28  
stud@debian:~/adheenamca$
```

AIM

32.Create a Bank account with members account number, name, type of account and balance. Write constructor and methods to deposit at the bank and withdraw an amount from the bank.

PROGRAM CODE

```
class Bank:
```

```
    def __init__(self,acc_no,name,acc_type,balance):
```

```
        self.acc_no=acc_no
```

```
        self.name=name
```

```
        self.acc_type=acc_type
```

```
        self.balance=balance
```

```
    def withdraw(self,x):
```

```
        self.balance=self.balance-x
```

```
        print("Balance amount of  after withdrawal",self.balance)
```

```
    def deposit(self,y):
```

```
        self.balance=self.balance+y
```

```
        print("Balance amount of  after deposit",self.balance)
```

```
    def displ(self):
```

```
        print("account no.:",self.acc_no)
```

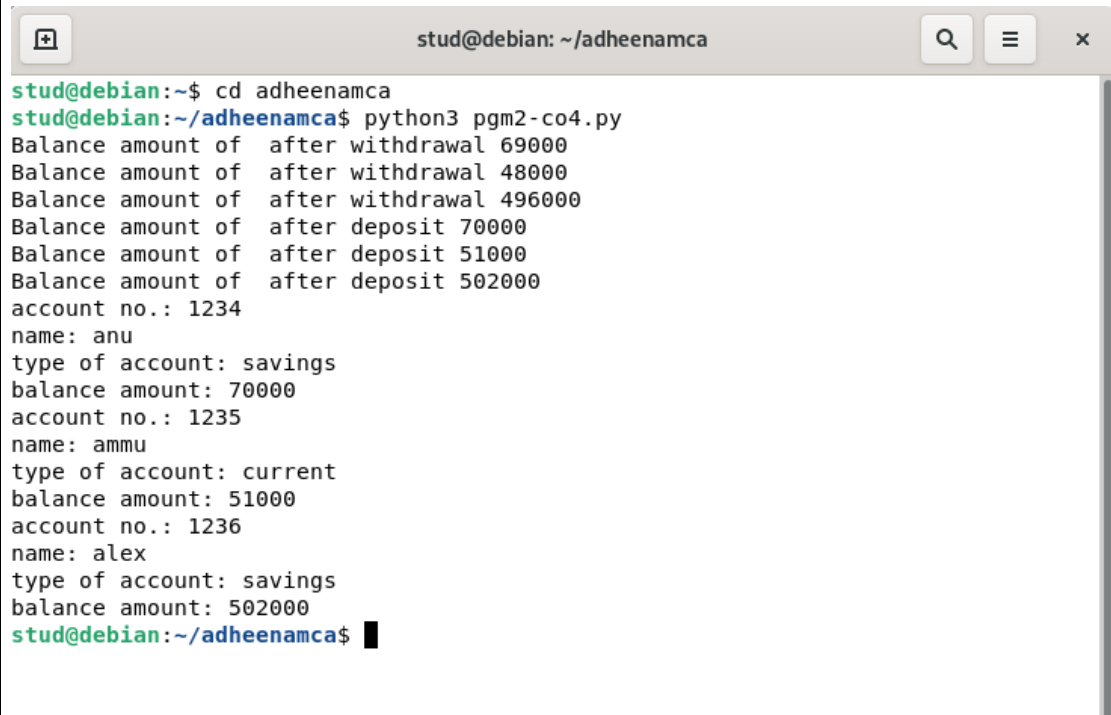
```
        print("name:",self.name)
```

```
        print("type of account:",self.acc_type)
```

```
        print("balance amount:",self.balance)
```

```
per1=Bank(1234,"anu","savings",70000)
per2=Bank(1235,"ammu","current",50000)
per3=Bank(1236,"alex","savings",500000)
per1.withdraw(1000)
per2.withdraw(2000)
per3.withdraw(4000)
per1.deposit(1000)
per2.deposit(3000)
per3.deposit(6000)
per1.displ()
per2.displ()
per3.displ()
```

OUTPUT



```
stud@debian: ~$ cd adheenamca
stud@debian:~/adheenamca$ python3 pgm2-co4.py
Balance amount of  after withdrawal 69000
Balance amount of  after withdrawal 48000
Balance amount of  after withdrawal 496000
Balance amount of  after deposit 70000
Balance amount of  after deposit 51000
Balance amount of  after deposit 502000
account no.: 1234
name: anu
type of account: savings
balance amount: 70000
account no.: 1235
name: ammu
type of account: current
balance amount: 51000
account no.: 1236
name: alex
type of account: savings
balance amount: 502000
stud@debian:~/adheenamca$
```

AIM

33.Create a class Rectangle with private attributes length and width. Overload '<' operator to compare the area of 2 rectangles.

PROGRAM CODE

```
class Rectangle:
```

```
    def __init__(self,length,breadth):
```

```
        self.length=length
```

```
        self.breadth=breadth
```

```
    def area(self):
```

```
        return self.length*self.breadth
```

```
    def perimeter(self):
```

```
        return 2*(self.length+self.breadth)
```

```
    def __lt__(self,r2):
```

```
        if(self.length*self.breadth<r2.length*r2.breadth):
```

```
            return True
```

```
        else:
```

```
            return False
```

```
r1=Rectangle(10,2)
```

```
r2=Rectangle(5,9)
```

```
x=r1.area()
```

```
y=r2.area()
```

```
z=r1.perimeter()

w=r2.perimeter()

print("Area of rectangle 1:",x);

print("Area of rectangle 2:",y);

print("Perimeter of rectangle 1:",z);

print("Perimeter of rectangle 2:",w);

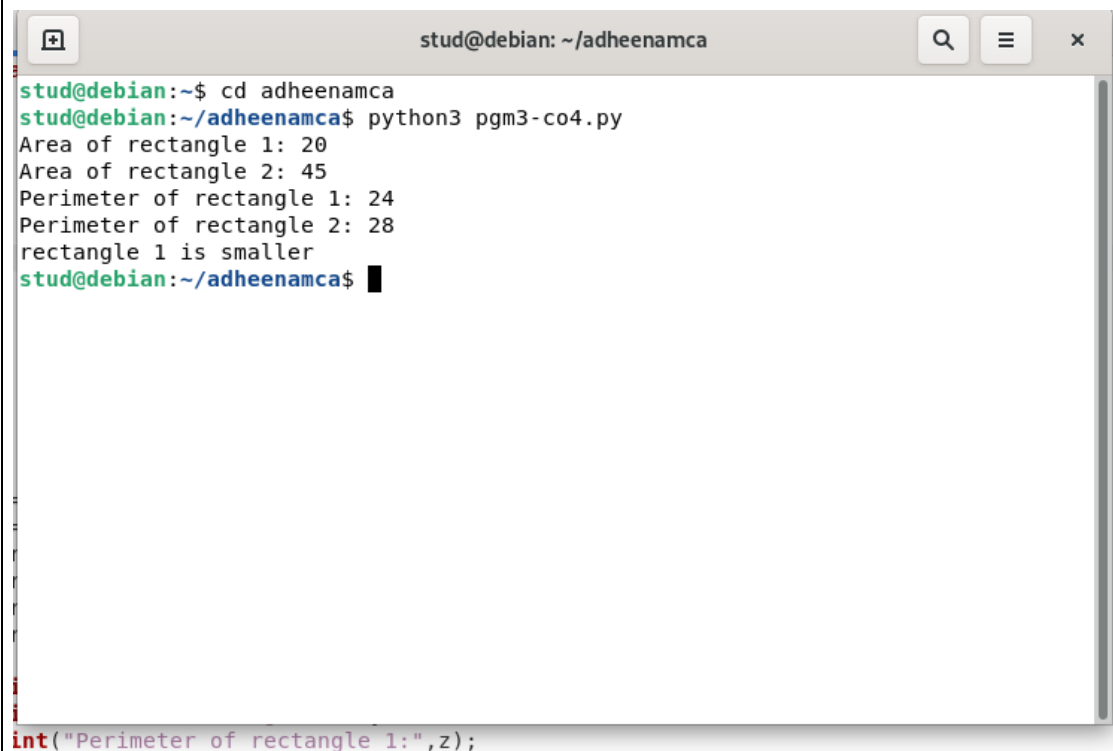
if(r1<r2):

    print("rectangle 1 is smaller")

else:

    print("rectangle 2 is smaller")
```

OUTPUT



```
stud@debian: ~/adheenamca
stud@debian:~$ cd adheenamca
stud@debian:~/adheenamca$ python3 pgm3-co4.py
Area of rectangle 1: 20
Area of rectangle 2: 45
Perimeter of rectangle 1: 24
Perimeter of rectangle 2: 28
rectangle 1 is smaller
stud@debian:~/adheenamca$
```

AIM

34. Create a class Time with private attributes hour, minute and second. Overload '+' operator to find sum of 2 time.

PROGRAM CODE

Source code

```
class Time:

    def __init__(self,hr,min,sec):

        self.__hr=hr

        self.__min=min

        self.__sec=sec

    def __add__(t1,t2):

        hr=t1.__hr+t2.__hr

        min=t1.__min+t2.__min

        sec=t1.__sec+t2.__sec

        print(hr,":",min,":",sec)

t1=Time(3,45,56)

t2=Time(4,20,3)

t1+t2
```


OUTPUT

A terminal window titled 'stud@debian: ~/mca' with search, menu, and close buttons. It shows the execution of 'cd mca' and 'python3 co45.py', which outputs '7 : 65 : 59'.

```
stud@debian:~$ cd mca
stud@debian:~/mca$ python3 co45.py
7 : 65 : 59
stud@debian:~/mca$
```

AIM

35. Create a class Publisher (name). Derive class Book from Publisher with attributes title and author. Derive class Python from Book with attributes price and no_of_pages. Write a program that displays information about a Python book. Use base class constructor invocation and method overriding.

PROGRAM CODE

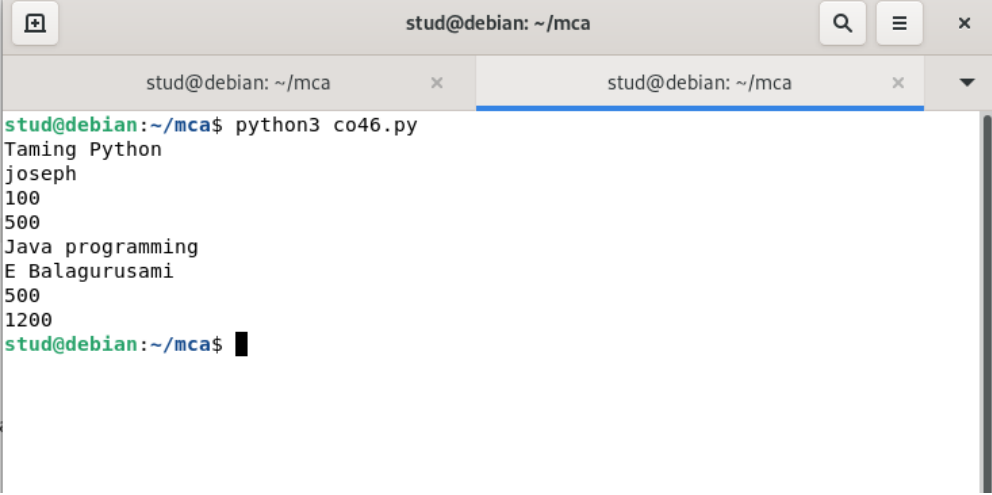
```
class Publisher(object):
    def __init__(self,name):
        self.name=name
    def display1(self):
        print(self.title)
        print(self.author)

class Book(Publisher):
    def __init__(self,name,title,author):
        super().__init__(name)
        self.title=title
        self.author=author
    def display2(self):
        #super().display1()
        print(self.title)
        print(self.author)

class Python(Book):
    def __init__(self,name,title,author,price,no_of_pages):
        super().__init__(name,title,author)
        self.price=price
        self.no_of_pages=no_of_pages
    def display3(self):
        super().display2()
        print(self.price)
        print(self.no_of_pages)
```

```
p=Python("ABC Publications","Taming Python","jeeva jose",100,500)
p.display3()
q=Python("XYZ Publications","Java programming","E
Balagurusami",500,1200)
q.display3()
```

OUTPUT

A screenshot of a terminal window titled 'stud@debian: ~/mca'. The terminal shows the execution of a Python script 'co46.py' using 'python3'. The output of the script is displayed line by line: 'Taming Python', 'joseph', '100', '500', 'Java programming', 'E Balagurusami', '500', and '1200'. The prompt 'stud@debian:~/mca\$' is visible at the bottom of the terminal.

```
stud@debian: ~/mca
stud@debian:~/mca$ python3 co46.py
Taming Python
joseph
100
500
Java programming
E Balagurusami
500
1200
stud@debian:~/mca$
```

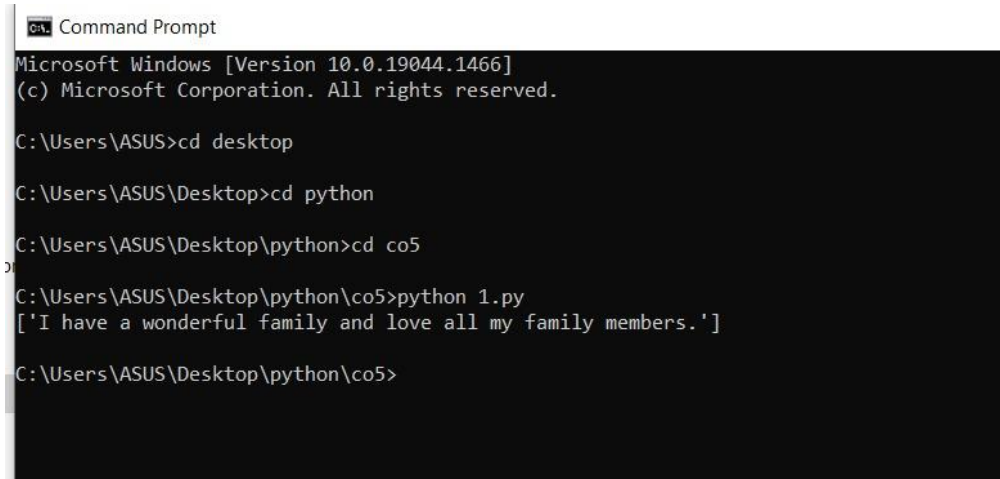
AIM

36. Write a Python program to read a file line by line and store it into a list.

PROGRAM CODE

```
fp=open("text_file.txt",'r')
lines=[]
for line in fp:
    lines.append(line.strip())
print(lines)
```

OUTPUT



```
C:\> Command Prompt
Microsoft Windows [Version 10.0.19044.1466]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ASUS>cd desktop

C:\Users\ASUS\Desktop>cd python

C:\Users\ASUS\Desktop\python>cd co5

C:\Users\ASUS\Desktop\python\co5>python 1.py
['I have a wonderful family and love all my family members.']

C:\Users\ASUS\Desktop\python\co5>
```

AIM

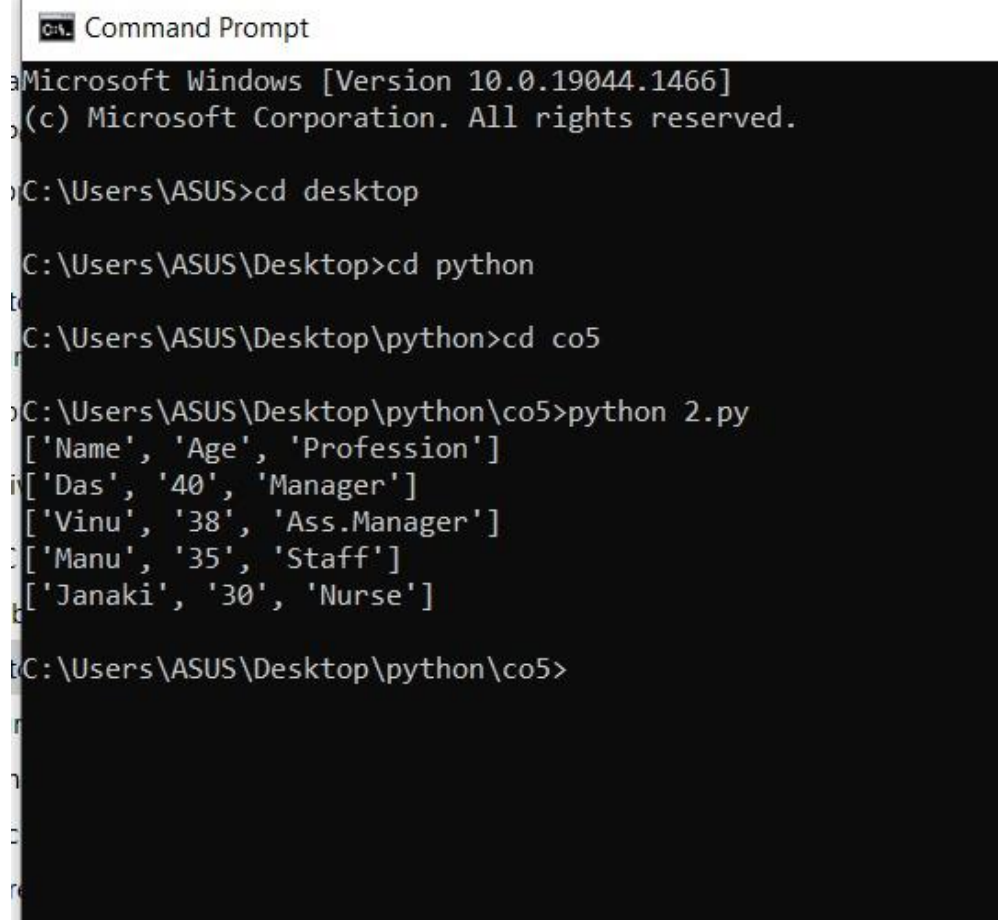
37. Write a Python program to read each row from a given csv file and print a list of strings.

PROGRAM CODE

```
import csv

with open('people.csv', 'r') as file:

    reader = csv.reader(file)
    for row in reader:
        print(row)
```

OUTPUT

```
Command Prompt
Microsoft Windows [Version 10.0.19044.1466]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ASUS>cd desktop

C:\Users\ASUS\Desktop>cd python

C:\Users\ASUS\Desktop\python>cd co5

C:\Users\ASUS\Desktop\python\co5>python 2.py
['Name', 'Age', 'Profession']
['Das', '40', 'Manager']
['Vinu', '38', 'Ass.Manager']
['Manu', '35', 'Staff']
['Janaki', '30', 'Nurse']

C:\Users\ASUS\Desktop\python\co5>
```