FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)™

HORMIS NAGAR, MOOKKANNOOR, ANGAMALY-683577

FOCUS ON EXCELLENCE

20MCA131 PROGRAMMING LAB

LABORATORY RECORD

Name: ABIMA YUGESH M

Branch: MASTER OF COMPUTER APPLICATIONS

Semester: 1 Batch: A Roll No: 05

MARCH 2022

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)™

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 ***ABIMA YUGESH M*** *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

12		Create a list of colors from comma-separated color names entered by user. Display first and last colors.	**15**	
13		Accept an integer n and compute n+nn+nnn.	**16**	
14		Print out all colors from color-list1 not contained in color-list2.	**16**	
15		Create a single string separated with space from two strings by swapping the character at position 1.	**17**	
16		Merge two dictionaries.	**18**	
17		Find gcd of 2 numbers.	**19**	
18		From a list of integers, create a list removing even numbers.	**20**	
19		Program to find the factorial of a number	**21**	
20		Generate Fibonacci series of N terms	**21**	
21		Find the sum of all items in a list	**22**	
22		Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square.	**23**	
23		Display the given pyramid with step number accepted from user.	**24**	
24		Count the number of characters (character frequency) in a string.	**25**	
25		Add ‘ing’ at the end of a given string. If it already ends with ‘ing’, then add ‘ly’.	**26**	
26		Accept a list of words and return length of longest word.	**26**	
27		Construct following pattern using nested loop.	**27**	

		* *		
28		Generate all factors of a number.	**28**	
29		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)	**29**	
30		Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two Rectangle objects by their area.	**32**	
31		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.	**33**	
32		Create a class Rectangle with private attributes length and width. Overload '<' operator to compare the area of 2 rectangles.	**35**	
33		Create a class Time with private attributes hour, minute and second. Overload '+' operator to find sum of 2 time	**36**	
34		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	**37**	

		displays information about a Python book. Use base class constructor invocation and method overriding.		
35		Write a Python program to read a file line by line and store it into a list.	**39**	
36		Write a Python program to read each row from a given csv file and print a list of strings.	**40**	

COURSE OUTCOME 1

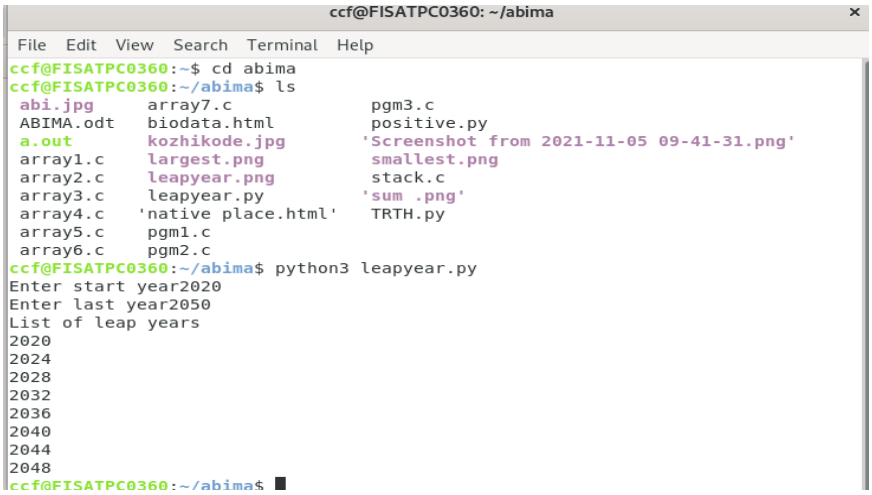
1. Display future leap years from current year to a final year entered by

User.

SOURCE CODE

```
startyear=int(input("Enter start year"))
lastyear=int(input("Enter last year"))
print("List of leap years")
for year in range(startyear,lastyear):
        if(year%4==0) and (year%100!=0) or (year%400==0):
                print(year)
```

OUTPUT



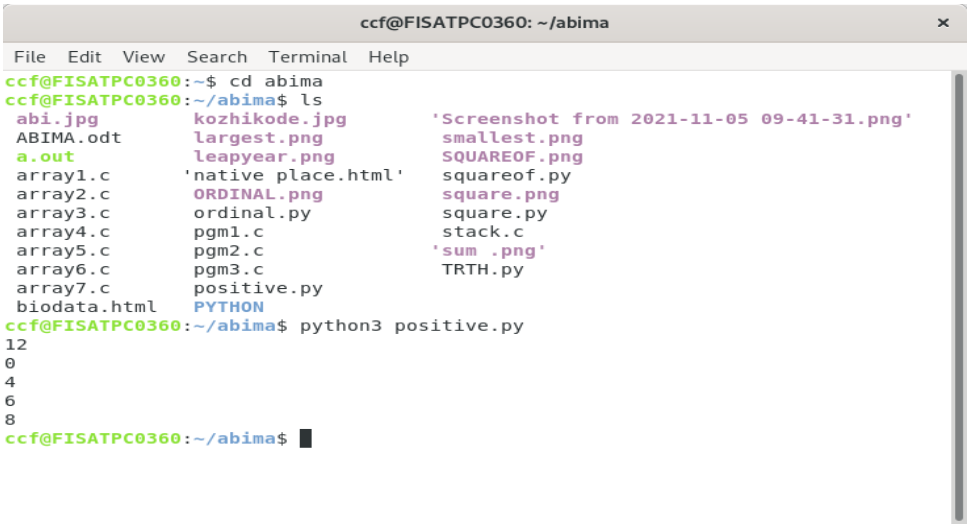
2.List comprehensions:

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

SOURCE CODE

```
list1=[12,-1,-2,0,4,6,8]
for num in list1:
      if(num>=0):
              print(num)
```

OUTPUT

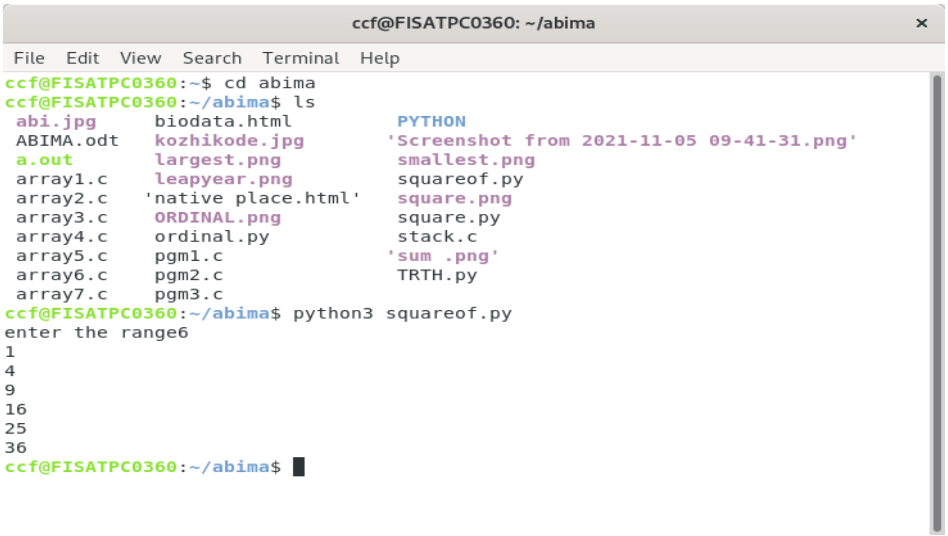


b .Square of N numbers

SOURCE CODE

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

OUTPUT

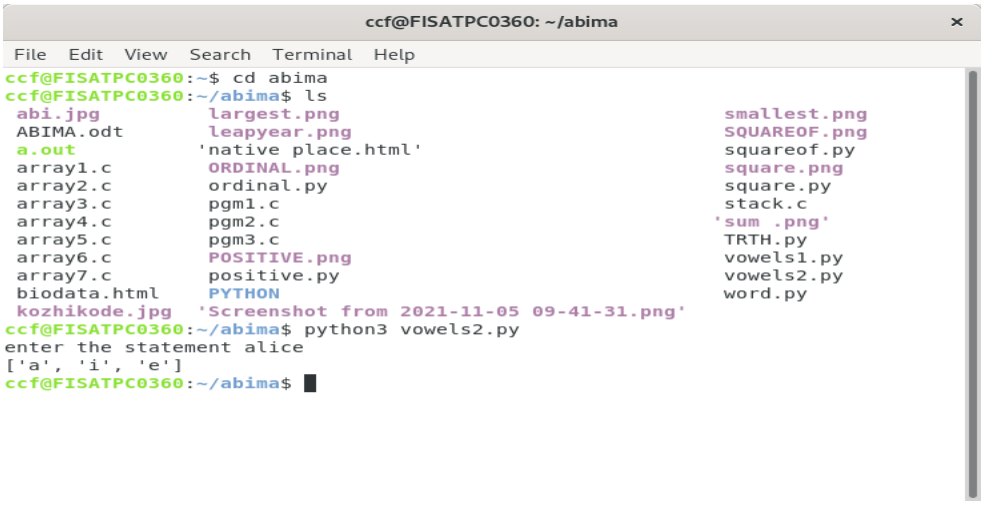


2c .Form a list of vowels selected from a given word.

SOURCE CODE

```
s=input("Enter a string: ")
list=[]
for i in s:
   if i in "aeiouAEIOU":
            list.append(i)
print("vowels in the list are:")
print(list)
```

OUTPUT

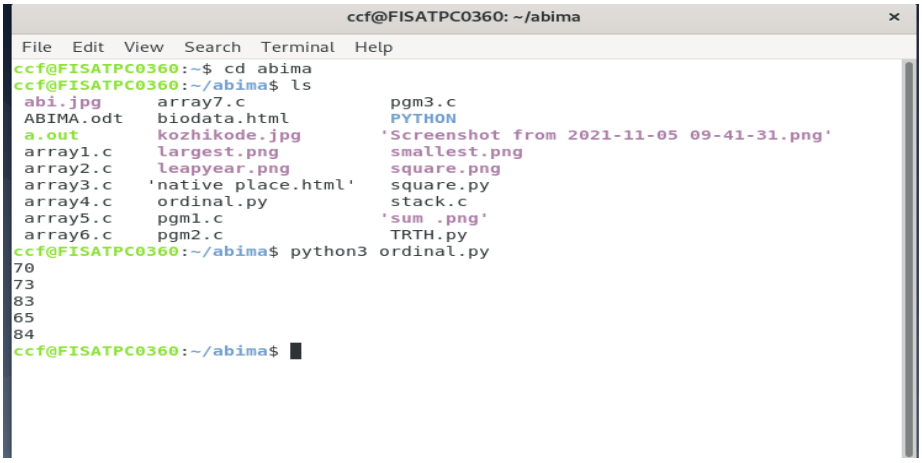


d .List ordinal values of each element of a word.

SOURCE CODE

```
list=['F','I','S','A','T']
for i in range(0,5):
   value=ord(list[i])
   print(value)
```

OUTPUT

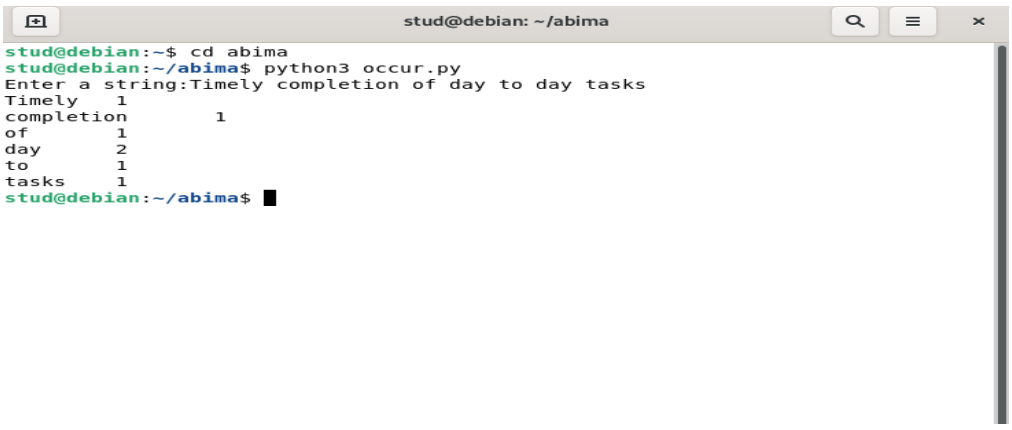


3. Count the occurrences of each word in a line of text.

SOURCE 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

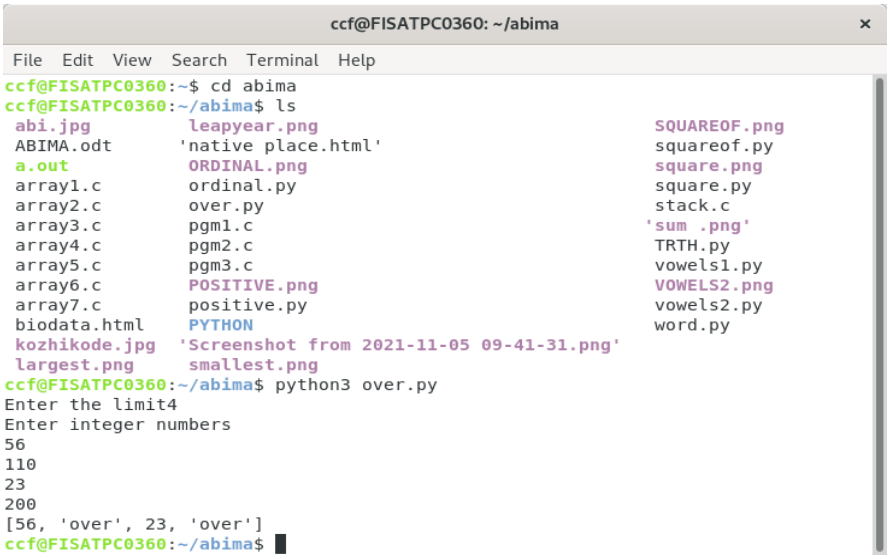


4.Prompt the user for a list of integers. For all values greater than 100, store 'over' instead.

SOURCE 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

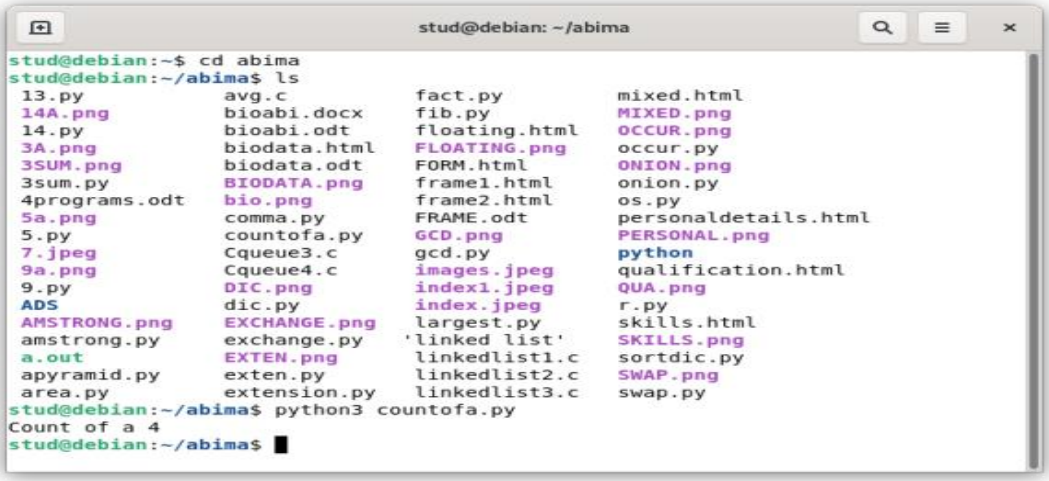


5.Store a list of first names. Count the occurrences of 'a' within the list.

SOURCE CODE

```
list=["abi","adheena","lal"]
count=0
for word in list:
        for i in word:
                if i=='a':
                        count+=1
print("Count of a")
```

OUTPUT



6. Enter 2 lists of integers.Check

a. whether list are of same length

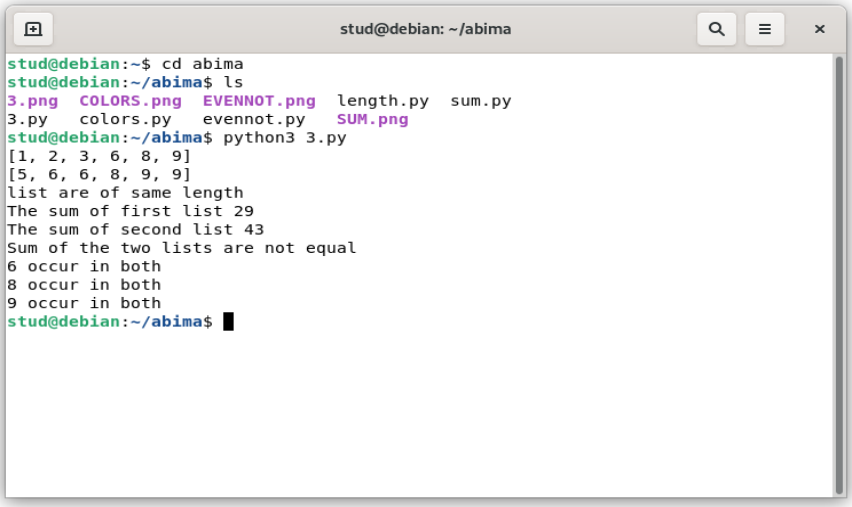
b. whetherlist sums of same value

c. whether any value occur in both.

SOURCE CODE

```
l1=[1,2,3,6,8,9]
l2=[5,6,6,8,9,9]
print(l1)
print(l2)
if (len(l1)==len(l2)):
        print("list are of same length")
else:
        print("list are of different length")
sum1=0
sum2=0
for i in range(len(l1)):
        sum1=sum1+l1[i]
print("The sum of first list",sum1)
for j in range(len(l2)):
        sum2=sum2+l2[j]
print("The sum of second list",sum2)
if(sum1==sum2):
        print("Sum of the two lists are equal")
else:
        print("Sum of the two lists are not equal")
flag=0
for i in l1:
        if i in l2:
                print(i,"occur in both")
                flag=1
if(flag==0):
        print("no common")
```

OUTPUT

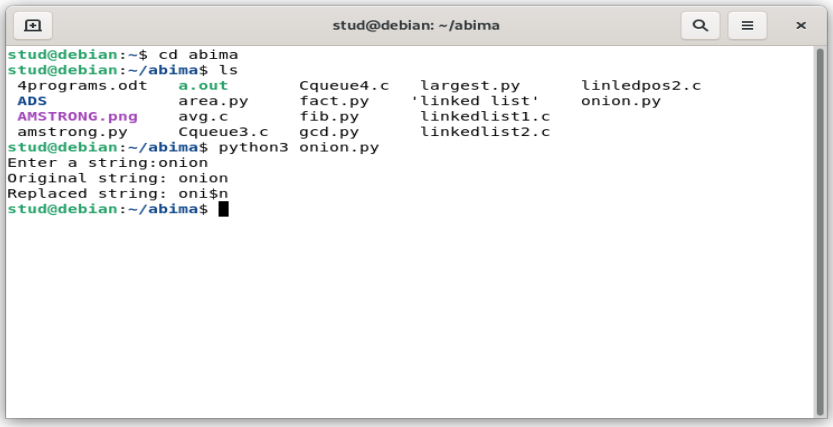


7.Get a string from an input string where all occurrences of first character replaced with '$',except first character.[eg:onion->oni$n]

SOURCE CODE

```
str1=input("Enter a string:")
print("Original string:",str1)
char=str1[0]
str1=str1.replace(char,'$')
str1=char+str1[1:]
print("Replaced string:",str1)
```

OUTPUT



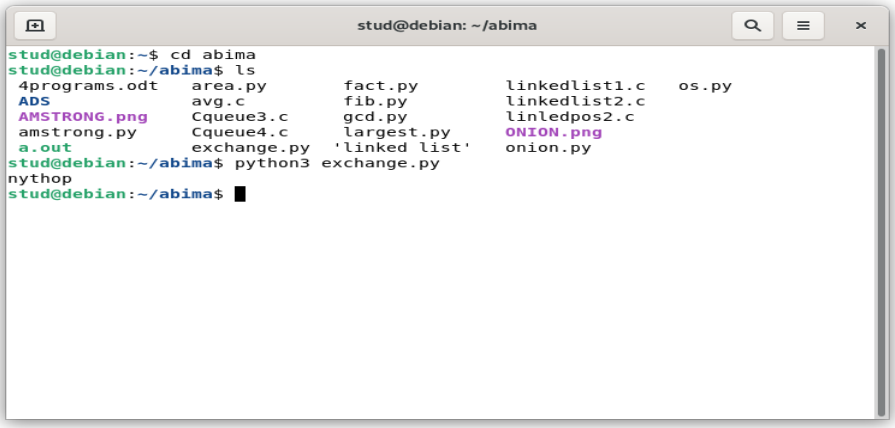
8.Create a string from given string where first and last characters exchanged.

[eg:python->nythop]

SOURCE CODE

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

OUTPUT



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

SOURCE CODE

```
x=input("Enter the radius")
x=int(x)
a=3.14*x*x
print(a)
```

OUTPUT

```
stud@debian:~$ cd abima
stud@debian:~/abima$ ls
area.py  fact.py  fib.py  kk  largest.py
stud@debian:~/abima$ python3 area.py
Enter the radius6
113.03999999999999
stud@debian:~/abima$
```

10.Find the biggest of 3 numbers

SOURCE CODE

```
a=input("Enter the number")
b=input("Enter the second number")
c=input("Enter the third number")
a=int(a)
b=int(b)
c=int(c)
if a>b:
        if a>c:
                print (a)
        else:
                print (c)
else:
        if b>c:
                print (b)
        else:
                print (c)
```

OUTPUT

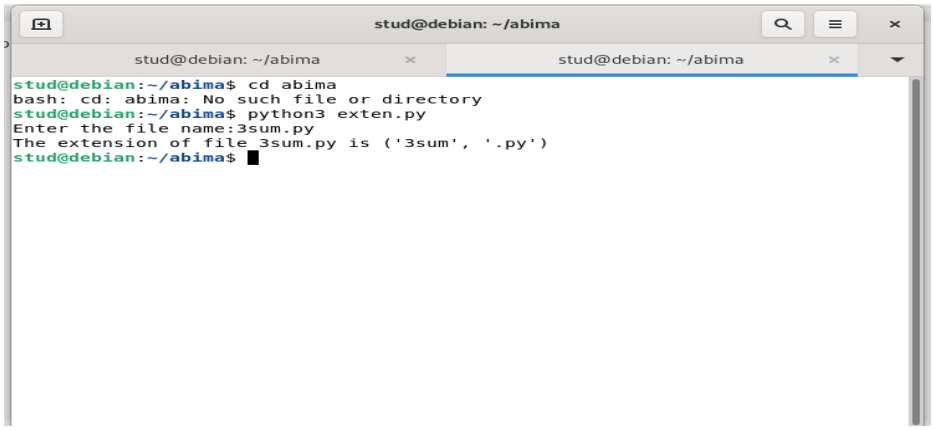
```
stud@debian:~$ cd abima
stud@debian:~/abima$ ls
area.py  fact.py  fib.py  kk  largest.py
stud@debian:~/abima$ python3 largest.py
Enter the number5
Enter the second number7
Enter the third number10
10
stud@debian:~/abima$
```

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

SOURCE CODE

```
import os

a=input("Enter the file name:")

print("The extension of file",a,"is",os.path.splitext(a))
```

OUTPUT



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

SOURCE 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])
```

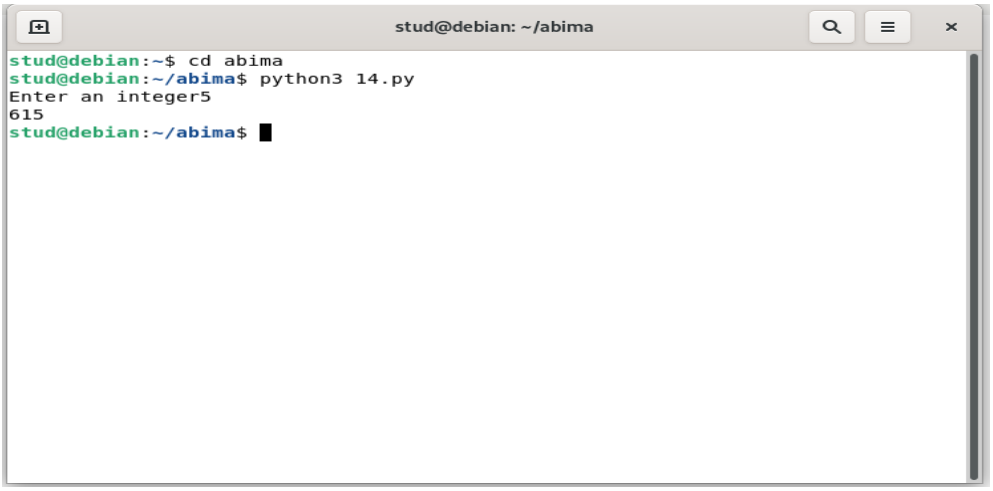
OUTPUT

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

SOURCE 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

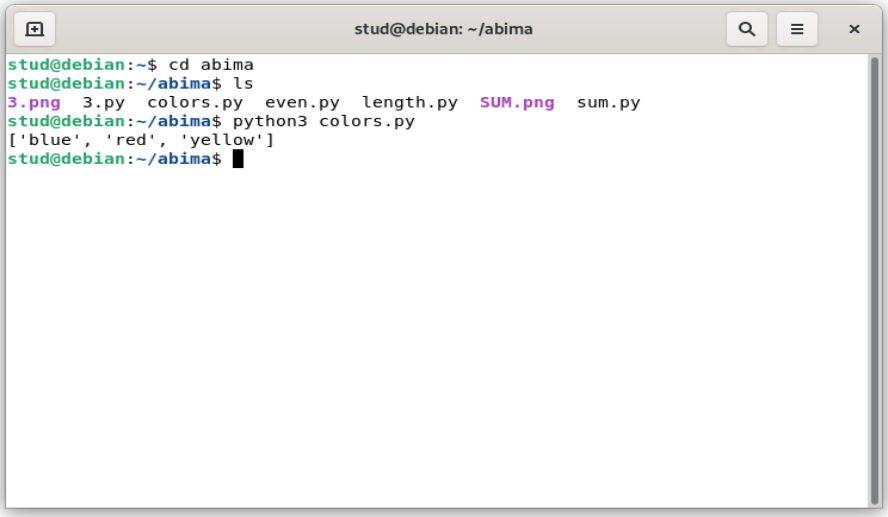


14.Print out all color from color-list1 not contained in color-list2

SOURCE CODE

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

OUTPUT

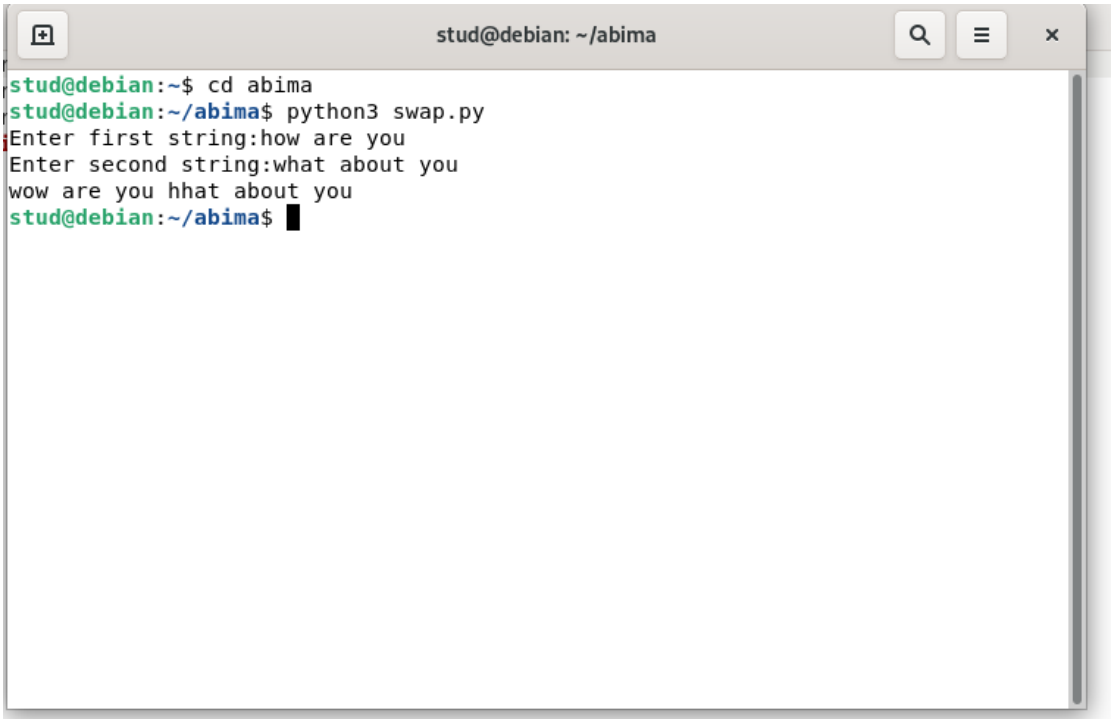


15.Create a single string separated with space from two strings by swapping the character at position 1.

SOURCE CODE

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

OUTPUT

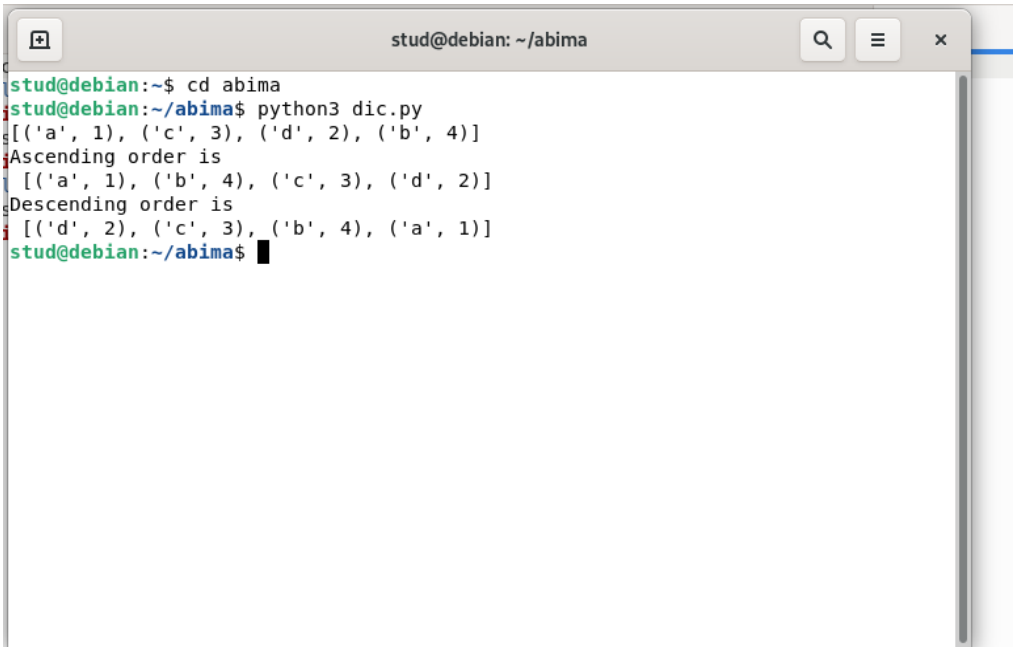


16.Merge two dictionaries.

SOURCE 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

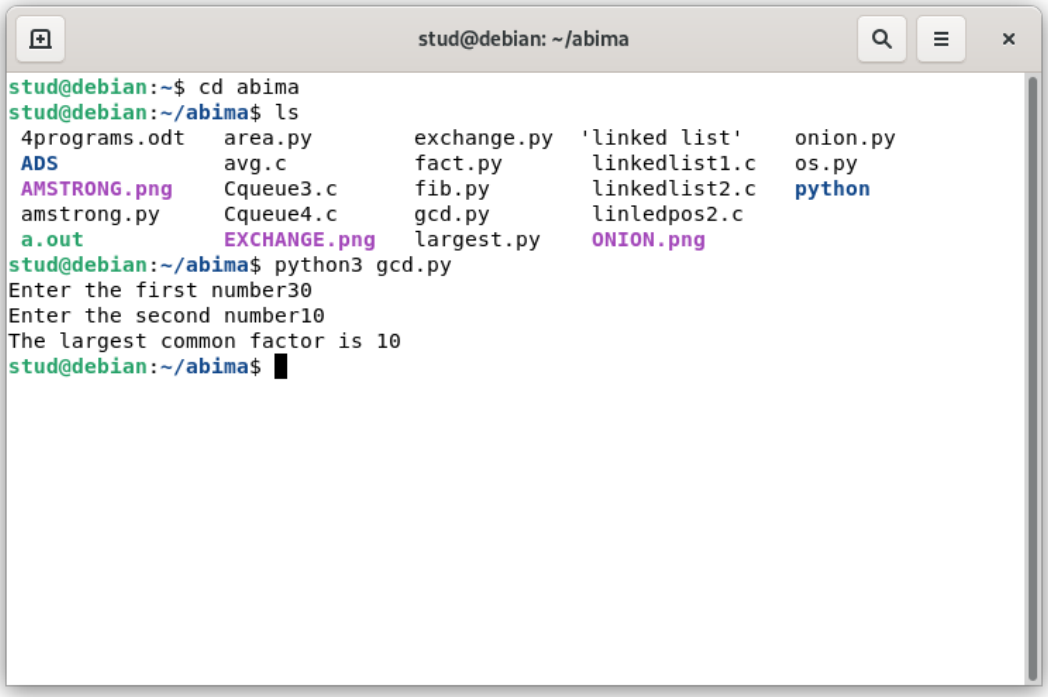


17.Find gcd of 2 numbers

SOURCE CODE

```
a=int(input("Enter the first number"))
b=int(input("Enter the second number"))
if(a>b):
        x1=b
else:
        x1=a
for i in range(1,x1+1):
        if((a%i==0) and (b%i==0)):
                gcd=i
print("The largest common factor is",gcd)
```

OUTPUT



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

SOURCE CODE

```
l1=[1,2,3,4,5,6]
l2=[]
for i in l1:
        if(i%2!=0):
                l2.append(i)
print(l2)
```

OUTPUT

COURSE OUTCOME 2

19.Program to find the factorial of a number.

SOURCE CODE

```
n=int(input('Enter the number:'))

fact=1

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

        fact=fact*i

print(fact)
```

```
stud@debian:~$ cd abima
stud@debian:~/abima$ ls
area.py  fact.py  fib.py  kk  largest.py
stud@debian:~/abima$ python3 fact.py
Enter the number7
5040
stud@debian:~/abima$
```

20.Generate fibonacci series of N terms.

SOURCE CODE

```
n=int(input('Enter a limit:'))

a=0

b=1

print(a)

print(b)

for i in range (2,n):

            c=a+b

            print(c)

            a=b
            b=c
```

OUTPUT

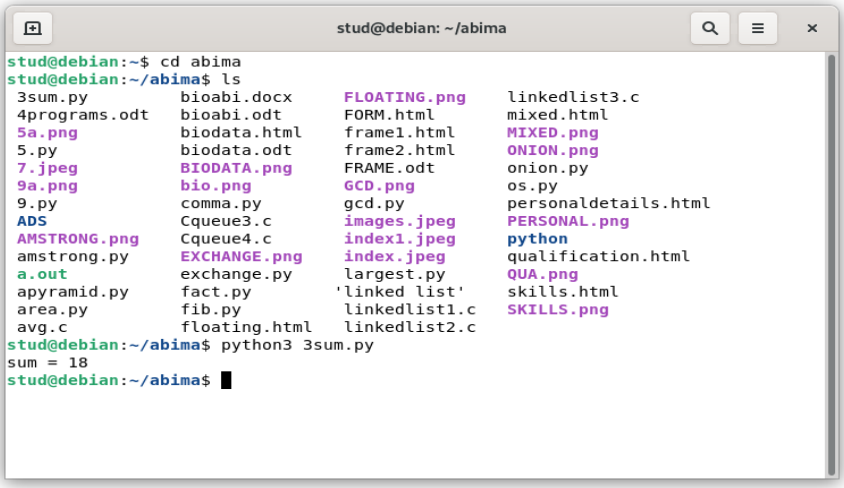
```
stud@debian:~$ cd abima
stud@debian:~/abima$ ls
area.py  fact.py  fib.py  kk  largest.py
stud@debian:~/abima$ python3 fib.py
Enter the number5
0
1
1
2
3
5
8
stud@debian:~/abima$
```

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

SOURCE CODE

```
l1=[1,2,3,4,5,3]
sum=0
for i in range(len(l1)):
        sum=sum+l1[i]
print("sum =",sum)
```

OUTPUT

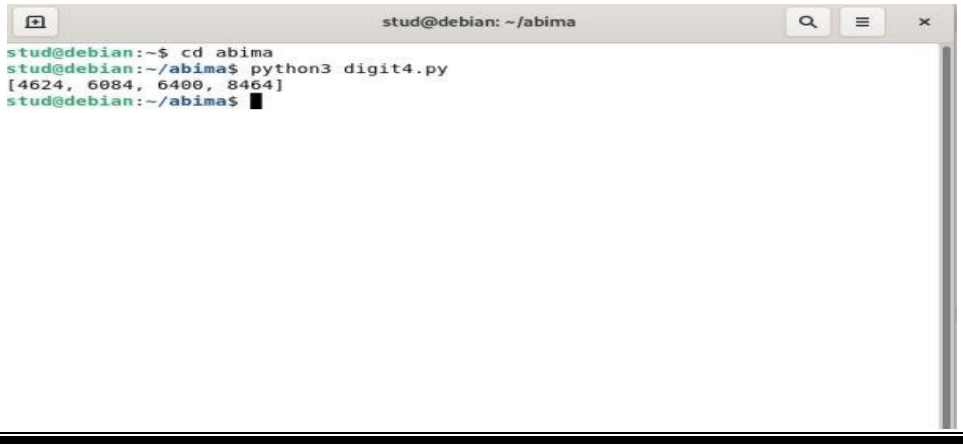


22.Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square.

SOURCE 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

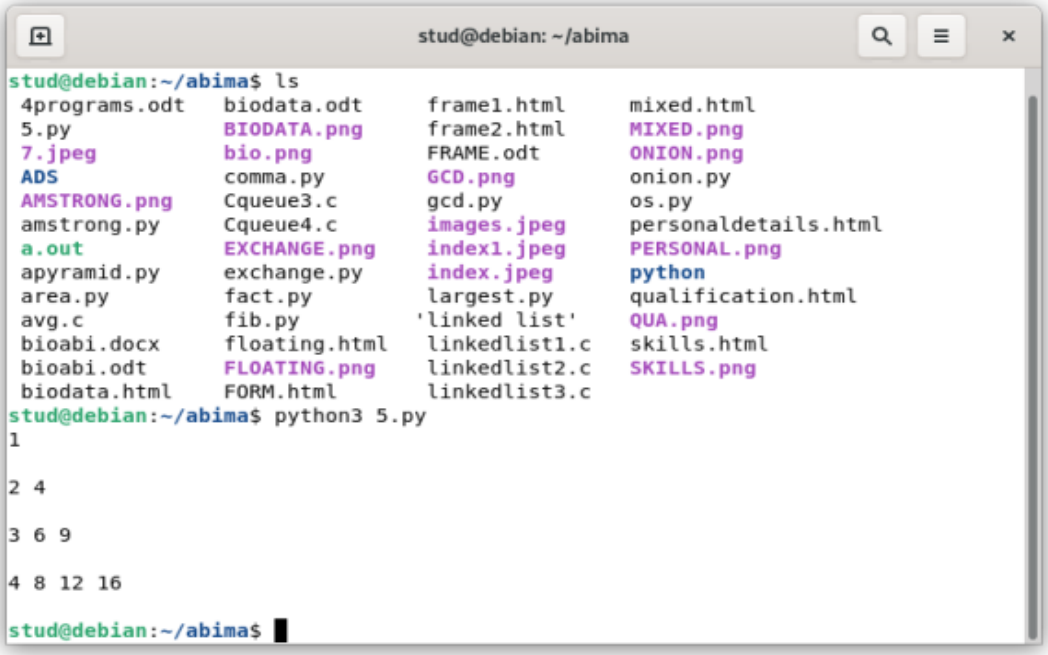


23.Display the given pyramid with step number accepted from user.

SOURCE CODE

```
for j in range(1,5):
    for i in range(1,j+1):
        i=j*i
        print(i,end=" ")
    print("\n")
```

OUTPUT



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

SOURCE 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)
```

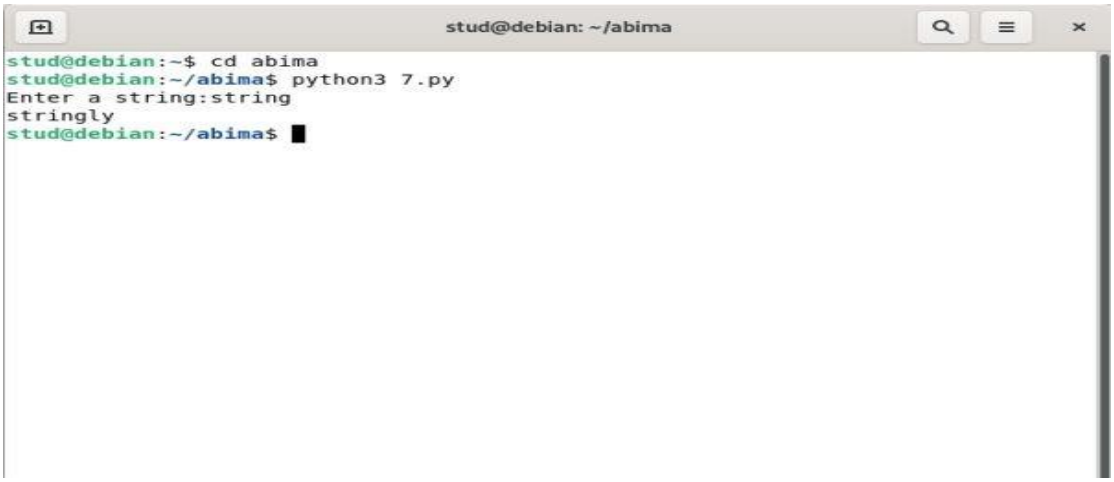
OUTPUT

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

SOURCE CODE

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

OUTPUT

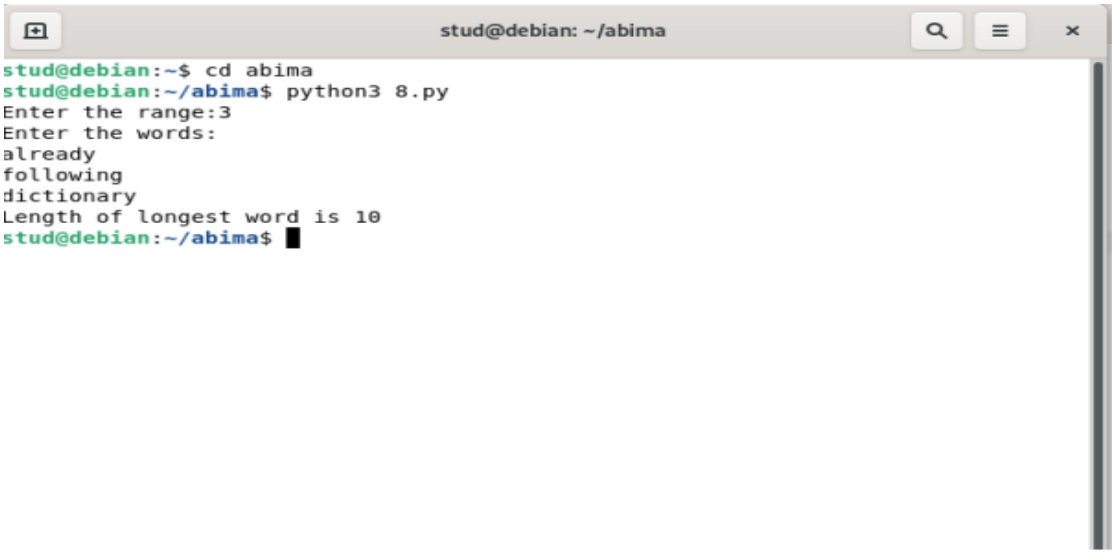


26.Accept a list of words and return length of longest word.

SOURCE 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



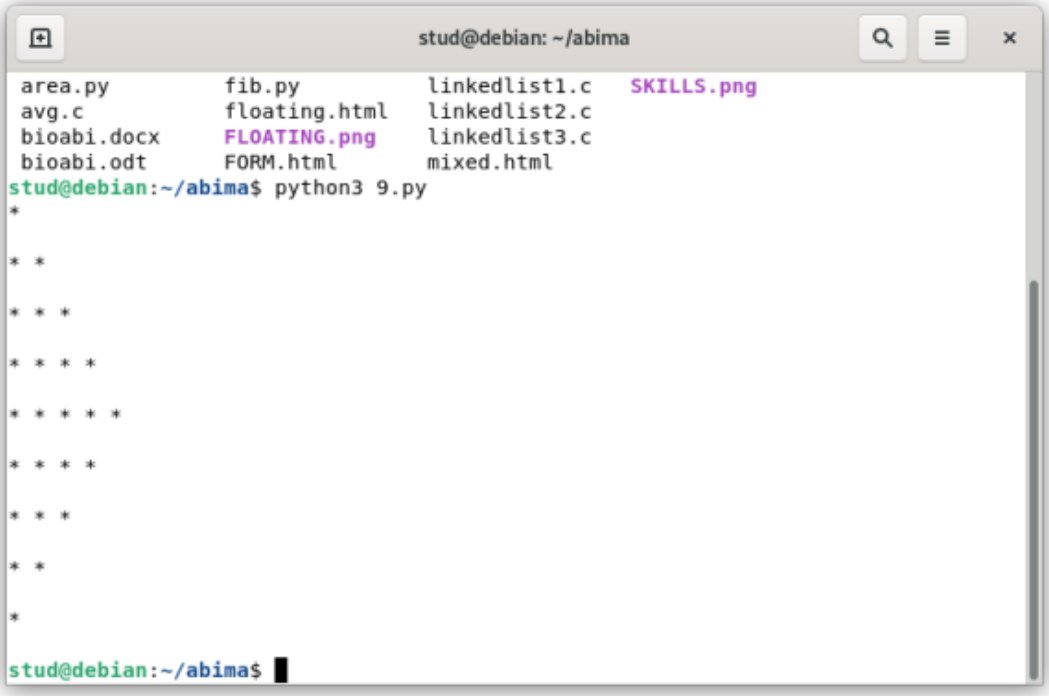
27.Construct following pattern using nested loop.

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

SOURCE CODE

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

OUTPUT

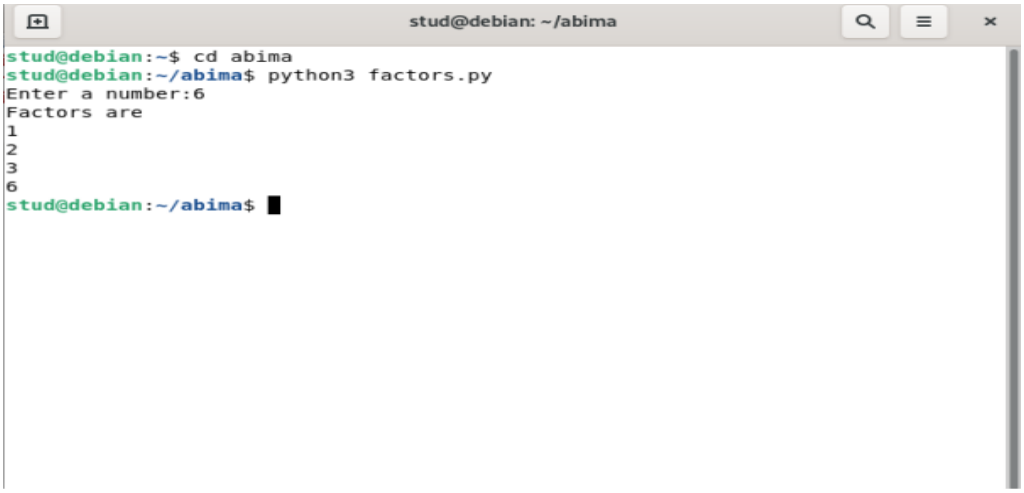


28.Generate all factors of a number.

SOURCE CODE

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

OUTPUT



COURSE OUTCOME 3

29.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)

SOURCE CODE

Graphice\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
```

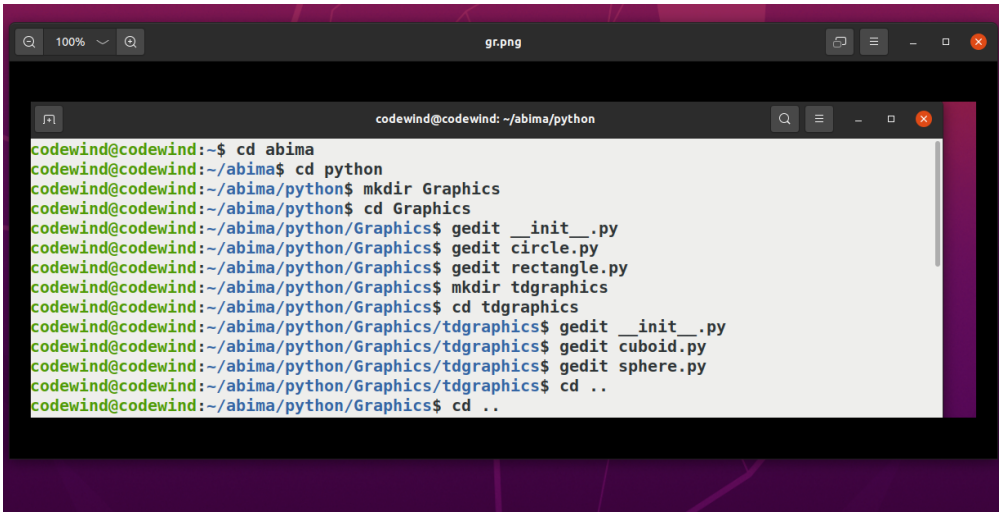
Graphics\tdgraphics\sphere.py

```
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 15 is : ",circle.area_circle(15))
print("Permeter of a circle with radius 15 is ",circle.perimeter_circle(15))
print("\n")

print("Area of a Rectangle with length and width 15 is :
      ",rectangle.area_rec(15,15))
print("Permeter of a Rectangle with length and width 15 is :
      ",rectangle.perimeter_rec(15,15))
print("\n")

print("Area of a  cuboid with length,width,height 15 is :
      ",cuboid.area_cuboid(15,15,15))
print("Volume of a cuboid with length,width,height 15 is :
      ",cuboid.volume_cuboid(15,15,15))
print("\n")

print("Area of a spere with radius 15 is : ",sphere.area_sphere(15))
print("Permeter of a spere with radius 15 is ",sphere.perimeter_sphere(15))
```

OUTPUT


100%
gr.png
codewind@codewind: ~/abima/python
codewind@codewind:~$ cd abima
codewind@codewind:~/abima$ cd python
codewind@codewind:~/abima/python$ mkdir Graphics
codewind@codewind:~/abima/python$ cd Graphics
codewind@codewind:~/abima/python/Graphics$ gedit __init__.py
codewind@codewind:~/abima/python/Graphics$ gedit circle.py
codewind@codewind:~/abima/python/Graphics$ gedit rectangle.py
codewind@codewind:~/abima/python/Graphics$ mkdir tdgraphics
codewind@codewind:~/abima/python/Graphics$ cd tdgraphics
codewind@codewind:~/abima/python/Graphics/tdgraphics$ gedit __init__.py
codewind@codewind:~/abima/python/Graphics/tdgraphics$ gedit cuboid.py
codewind@codewind:~/abima/python/Graphics/tdgraphics$ gedit sphere.py
codewind@codewind:~/abima/python/Graphics/tdgraphics$ cd ..
codewind@codewind:~/abima/python/Graphics$ cd ..


100%
gr1.png
codewind@codewind: ~/abima/python
codewind@codewind:~$ cd abima
codewind@codewind:~/abima$ cd python
codewind@codewind:~/abima/python$ python3 graphics.py
Area of a circle with radius 15 is : 706.8583470577034
Permeter of a circle with radius 15 is 94.24777960769379
Area of a Rectangle with length and width 15 is : 225
Permeter of a Rectangle with length and width 15 is : 60
Area of a cuboid with length,width,height 15 is : 1350
Volume of a cuboid with length,width,height 15 is : 3375
Area of a spere with radius 15 is : 2827.4333882308138
Permeter of a spere with radius 15 is 94.24777960769379
codewind@codewind:~/abima/python$

COURSE OUTCOME 4

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

SOURCE CODE

```
class Rectangle:
        def __init__(self,l,b):
                self.l=l
                self.b=b
        def area(self):
                return self.l*self.b


        def perimeter(self):
                return 2*(self.l+self.b)



y1=Rectangle(15,5)
y2=Rectangle(25,5)
x=y1.area()
y=y2.area()
r=y1.perimeter()
v=y2.perimeter()
print("Area of first rectangle",x);
print("Area of second rectangle",y);
print("Perimeter of first rectangle",r);
print("Perimeter of second rectangle",v);
if(x>y):
        print("Area of first rectangle is greater")
else:
        print("Area of second rectangle is greater")
```

OUTPUT

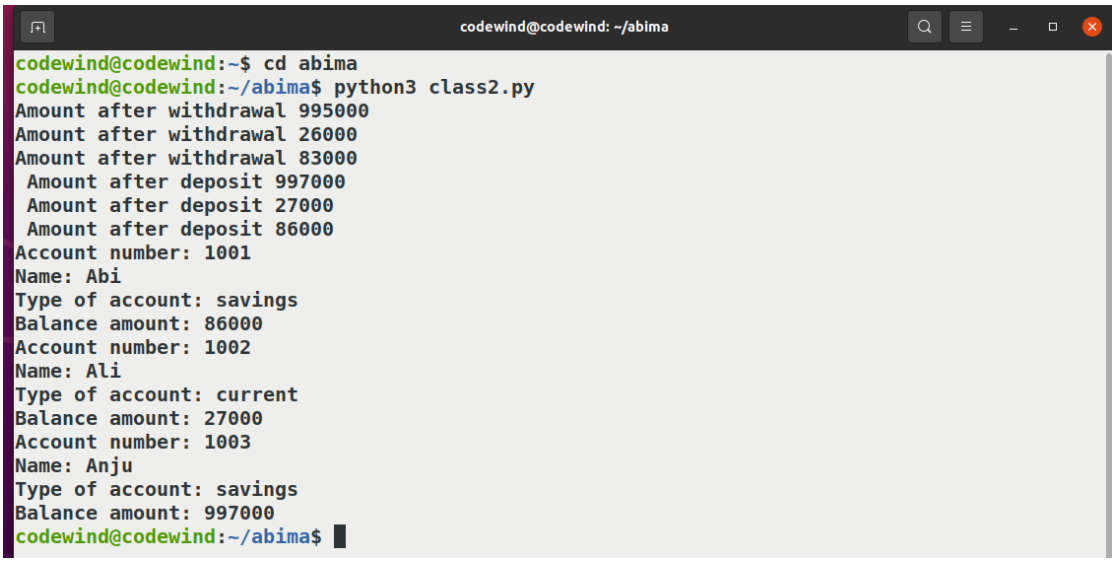
31.Create a Bank account with members account number, name, type of account and balance. Write constructor and methods to deposit at the bank and withdraw an amount from the bank.

SOURCE CODE

```
class Bank:
        def __init__(self,accountno,name,accounttype,bal):
                self.accountno=accountno
                self.name=name
                self.accounttype=accounttype
                self.bal=bal
        def withdraw(self,x):
                self.bal=self.bal-x
                print("Amount after withdrawal",self.bal)
        def deposit(self,y):
                self.bal=self.bal+y
                print(" Amount after deposit",self.bal)
        def display(self):
                print("Account number:",self.accountno)
                print("Name:",self.name)
                print("Type of account:",self.accounttype)
                print("Balance amount:",self.bal)
cus1=Bank(1001,"Abi","savings",90000)
cus2=Bank(1002,"Ali","current",30000)
cus3=Bank(1003,"Anju","savings",1000000)
```

```
cus3.withdraw(5000)
cus2.withdraw(4000)
cus1.withdraw(7000)
cus3.deposit(2000)
cus2.deposit(1000)
cus1.deposit(3000)
cus1.display()
cus2.display()
cus3.display()
```

OUTPUT



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

SOURCE CODE

```
class Rectangle:
        def __init__(self,length,breadth):
                self.__length = length
                self.__breadth = breadth

        def __lt__ (self,rect2):
                if self.__length*self.__breadth < rect2.__length*rect2.__breadth:
                        return True
                else:
                        return False

l=int(input("Enter length of rectangle1: "))
b=int(input("Enter breadth of rectangle1: "))
rect1 = Rectangle(l,b)

l=int(input("Enter length of rectangle2: "))
b=int(input("Enter breadth of rectangle2: "))
rect2 = Rectangle(l,b)

if rect1 < rect2:
        print("Second rectangle is larger")
else:
        print("First rectangle is larger")
```

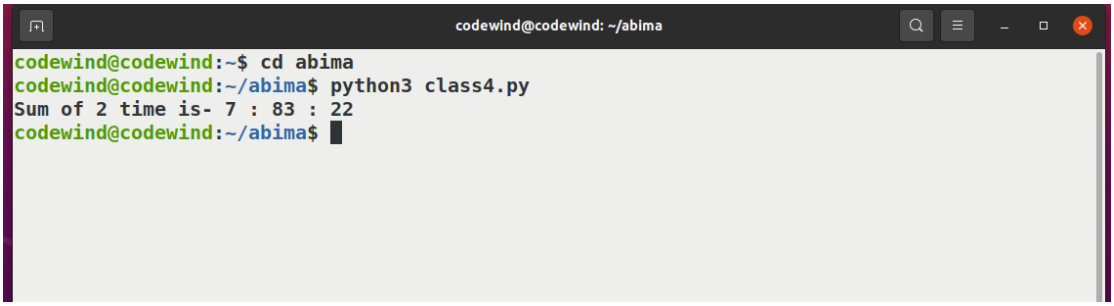
OUTPUT

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

SOURCE CODE

```
class Time:
        def __init__(self,hr,min,sec):
                self.__hr=hr
                self.__min=min
                self.__sec=sec
        def __add__(self,t2):
                a=t1.__hr+t2.__hr
                b=t1.__min+t2.__min
                c=t1.__sec+t2.__sec
                print("Sum of 2 time is""-",a,":",b,":",c)
t1=Time(3,50,12)
t2=Time(4,33,10)
t1+t2
```

OUTPUT



34.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.

SOURCE 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)
d=Python("DC Publications","C programming","Anu Krishna",200,1000)
d.display3()
e=Python("Victoria Publications","Java programming","R Raj",500,2000)
e.display3()
f=Python("Navodhaya Publications","Mathematics","S.Santhosh",700,3000)
f.display3()
```

OUTPUT

COURSE OUTCOME 5

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

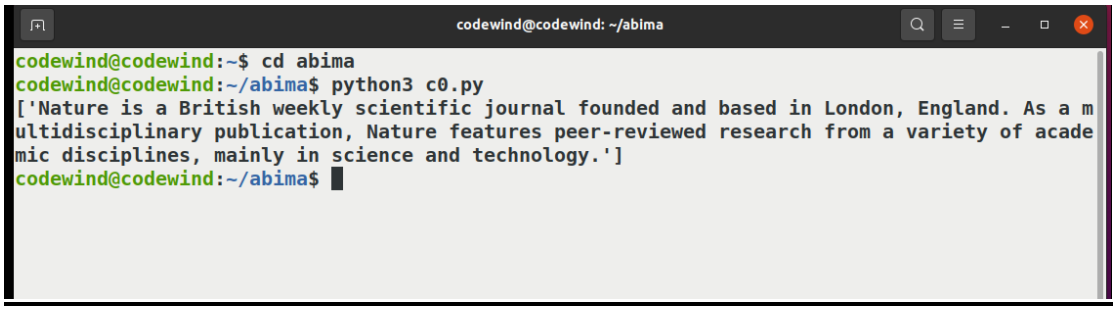
SOURCE CODE

```
fp=open("text.txt",'r')

lines=[]

for line in fp:

        lines.append(line.strip())

print(lines)
```

text.txt

Nature is a British weekly scientific journal founded and based in London, England. As a multidisciplinary publication, Nature features peer-reviewed research from a variety of academic disciplines, mainly in science and technology.

OUTPUT



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

SOURCE CODE

```
import csv

with open('m.csv', 'r') as file:
    reader = csv.reader(file)
    for row in reader:
        print(row)
```

m.csv

```
Name, Subject, Marks
Alen, Maths,78
Abhiraj, English, 90
Anu, Science, 89
Adarsh, IT, 94
```

OUTPUT