

# **FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)<sup>TM</sup>**

**HORMIS NAGAR, MOOKKANNOOR, ANGAMALY-683577**



**FOCUS ON EXCELLENCE**

**20MCA131 PROGRAMMING LAB LABORATORY RECORD**

**Name: CHRISTY P BABY**

**Branch: MASTER OF COMPUTER APPLICATIONS**

**Semester: 1 Batch: A Roll No: 49 Reg No:(FITMCA21-2049)**

**MARCH 2022**

# **FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)<sup>TM</sup>**

**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 **CHRISTY P BABY(FITMCA21-2049)** 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**

<b>Sl No</b>	<b>Date of Experiment</b>	<b>Title of the Experiment</b>	<b>Page No:</b>	<b>Signature of Staff –In – Charge</b>
1		Display future leap years from current year to a final year entered by user.		
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)		
3		Count the occurrences of each word in a line of text.		
4		Prompt the user for a list of integers. For all values greater than 100, store 'over' instead.		
5		Store a list of first names. Count the occurrences of 'a' within the list		
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		
7		Get a string from an input string where all occurrences of first character replaced with '\$', except first character. [eg: onion ->oni\$n]		
8		Create a string from given string where first and last characters exchanged. [eg: python ->nythop]		

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

25		Count the number of characters (character frequency) in a string.		
26		Add 'ing' at the end of a given string. If it already ends with 'ing', then add 'ly'		
27		Accept a list of words and return length of longest word.		
28		Construct following pattern using nested loop *		
29		Generate all factors of a number.		
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)		
31		Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two Rectangle objects by their area.		
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.		
33		Create a class Rectangle with private attributes length and width. Overload '<' operator to compare the area of 2 rectangles.		
34		Create a class Time with private attributes hour, minute and second. Overload '+' operator to find sum of 2 time.		
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.		
36		Write a Python program to read a file line by line and store it into a list.		
37		Write a Python program to read each row from a given csv file and print a list of string.		

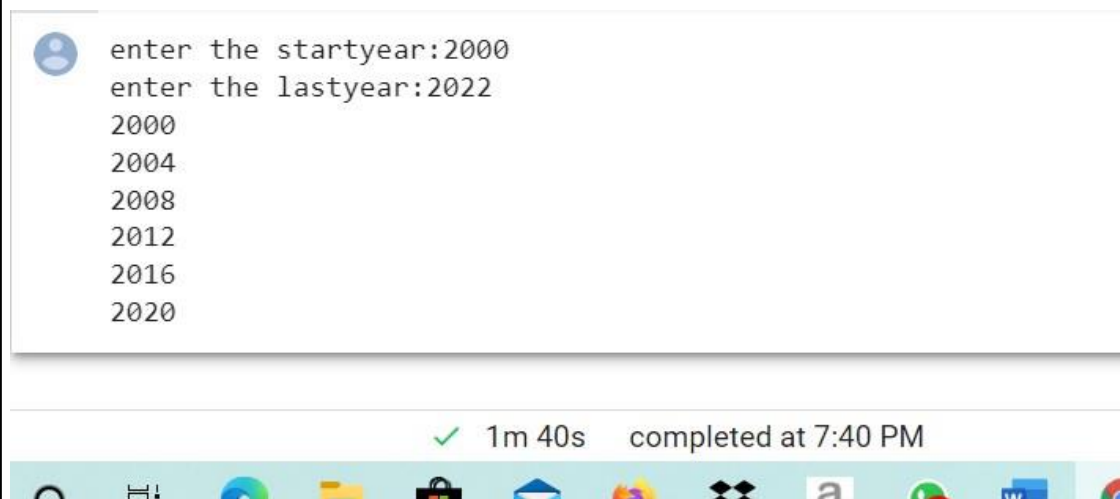
### **PROGRAM 1**

**AIM:** Display future leap years from current year to a final year entered by user.

**SOURCE CODE:**

```
l=int(input("enter the startyear:"))
m=int(input("enter the lastyear:"))
for i in range (l,m):
    if(i%400==0)or(i% 100!=0)and(i%4==0):
        print(i)
```

**OUTPUT:**

A screenshot of a Python program execution. The input shows 'enter the startyear:2000' and 'enter the lastyear:2022'. The output displays a list of leap years: 2000, 2004, 2008, 2012, 2016, and 2020. Below the output, a status bar indicates the program was completed successfully in 1m 40s at 7:40 PM. The bottom of the screenshot shows a Windows taskbar with various application icons.

```
enter the startyear:2000
enter the lastyear:2022
2000
2004
2008
2012
2016
2020
```

✓ 1m 40s completed at 7:40 PM

### **PROGRAM 2**

**AIM:**

List comprehensions:

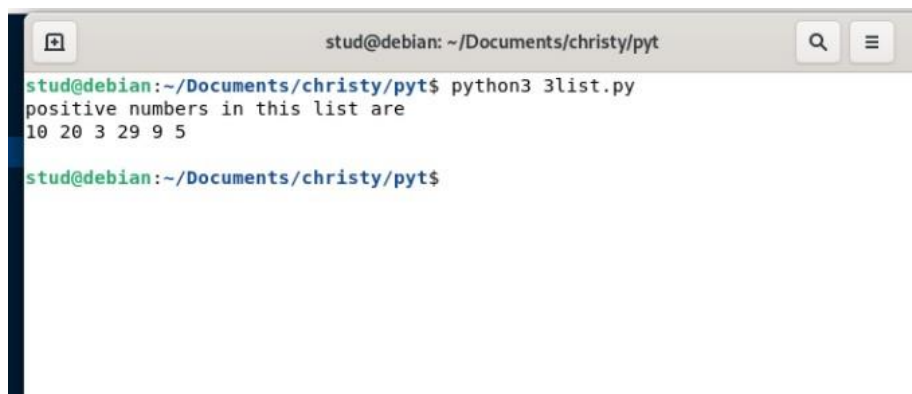
- e Generate positive list of numbers from a given list of integers
- f Square of N numbers
- g Form a list of vowels selected from a given word
- (h) List ordinal value of each element of a word (Hint: use ord() to get ordinal values)

### SOURCE CODE:

(e):

```
list0=[10,20,-49,-29,3,0,29,-8,-5,9,5]
print("positive numbers in this list are")
for num in list0:
    if(num>0):
        print(num,end = " ")
print("\n")
```

### OUTPUT:



```
stud@debian: ~/Documents/christy/pyt
stud@debian:~/Documents/christy/pyt$ python3 3list.py
positive numbers in this list are
10 20 3 29 9 5
stud@debian:~/Documents/christy/pyt$
```

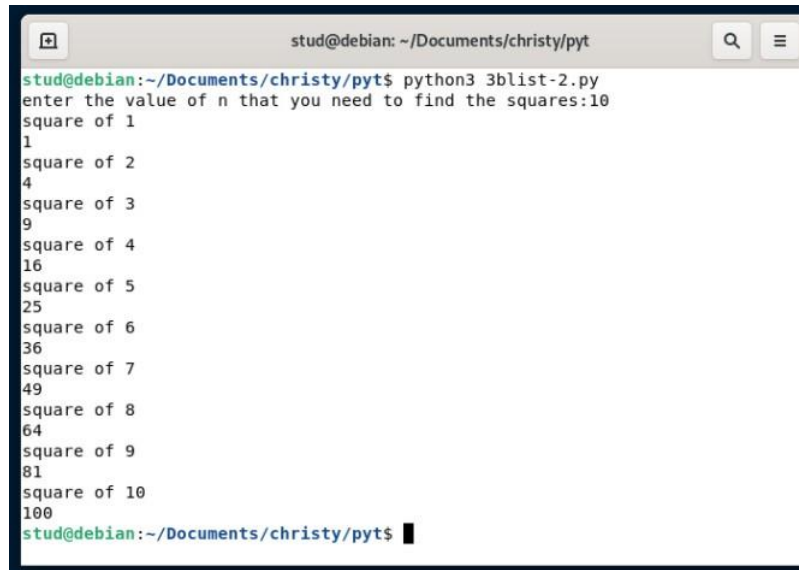
(f)

### SOURCE CODE:

```
sq=int(input("enter the value of n that you need to find the squares:"))
s=0
for i in range(1,sq+1):
    print("square of",i,)
    s=i*i
    print(s)
```



## OUTPUT:



```
stud@debian: ~/Documents/christy/pyt
stud@debian:~/Documents/christy/pyt$ python3 3blist-2.py
enter the value of n that you need to find the squares:10
square of 1
1
square of 2
4
square of 3
9
square of 4
16
square of 5
25
square of 6
36
square of 7
49
square of 8
64
square of 9
81
square of 10
100
stud@debian:~/Documents/christy/pyt$
```

(g)

## SOURCE CODE:

```
a=(input("enter the word:"))
list1=['a','e','i','o','u']
list2=[]
for v in a:
    if(v in list1 and v not in list2):
        list2.append(v)
print("vowels in this word are:",list2)
```

## OUTPUT:



```
stud@debian: ~/Documents/christy/pyt
stud@debian:~/Documents/christy/pyt$ python3 3cvow.py
enter the word:Hello world
vowels in this word are: ['e', 'o']
stud@debian:~/Documents/christy/pyt$
```

(h)

## SOURCE CODE:

```
s=input("Enter the word")
print([ord(p) for p in s])
```

## OUTPUT



```
stud@debian: ~/Documents/christy/pyt
stud@debian:~/Documents/christy/pyt$ python3 3d.py
Enter the wordWorld
[87, 111, 114, 108, 100]
stud@debian:~/Documents/christy/pyt$
```

### **PROGRAM 3**

#### **AIM:**

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:**



```
stud@debian: ~/Documents/christy/pyt
stud@debian:~/Documents/christy/pyt$ python3 33.py
Enter a string:programming world
programming      1
world            1
stud@debian:~/Documents/christy/pyt$
```

### **PROGRAM 4**

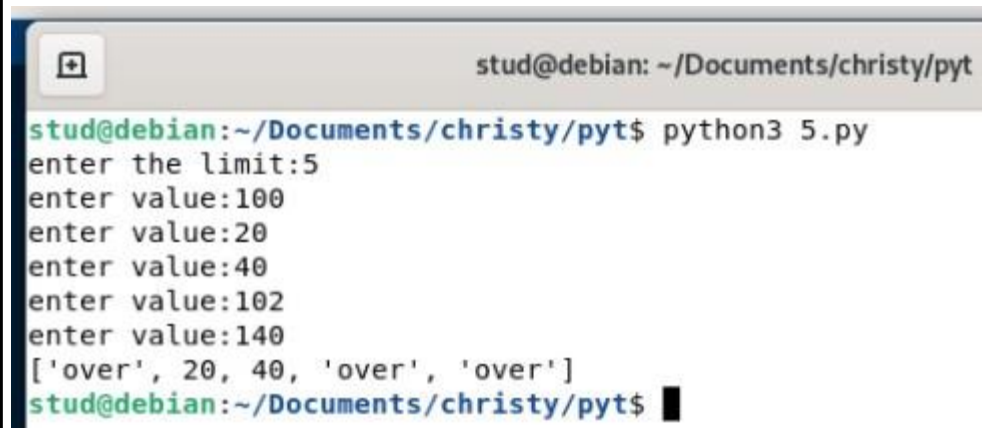
**AIM:** Prompt the user for a list of integers. For all values greater than 100, store 'over' instead.

#### **SOURCE CODE:**

```
num=int(input("enter the limit:"))
list1=[]
for r in range(0,num):
    num1=int(input("enter value:"))
    if(num1<100):
        list1.append(num1)
```

```
        else:
            list1.append("over")
print(list1)
```

**OUTPUT:**

A terminal window screenshot with a title bar showing 'stud@debian: ~/Documents/christy/pyt'. The terminal content shows the execution of 'python3 5.py'. It prompts for a limit (5) and five values (100, 20, 40, 102, 140). The output is a list: ['over', 20, 40, 'over', 'over'].

```
stud@debian:~/Documents/christy/pyt$ python3 5.py
enter the limit:5
enter value:100
enter value:20
enter value:40
enter value:102
enter value:140
['over', 20, 40, 'over', 'over']
stud@debian:~/Documents/christy/pyt$
```

**PROGRAM 5**

**AIM:**

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

**SOURCE CODE:**

```
name=['anna','aldrina','anjoe']
p=0
alp="a"
for v in name:
    for i in v:
        if i=='a':
            p=p+1

print("occurance of a in list is:",p)
```

## OUTPUT:

A terminal window with a title bar that says "stud@debian: ~/Documents/christy/pyt". The terminal shows the command "python3 6a.py" being executed. The output of the script is "occurrence of a in list is: 5". The prompt "stud@debian:~/Documents/christy/pyt\$" is visible on the line following the output.

```
stud@debian:~/Documents/christy/pyt$ python3 6a.py
occurrence of a in list is: 5
stud@debian:~/Documents/christy/pyt$
```

## PROGRAM 6

**AIM:** 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

## **SOURCE CODE:**

(a)

```
l1=[1,2,3,4,7]
l2=[2,3,4,5]
p=len(l1)
s=len(l2)
if(p==s):
    print("both list length are same")
else:
    print("both list length are not same")
```

**(b)**

```
l1=[2,3,4]
l2=[3,4,2]
s=0
p=0
for i in l1:
    s=s+i
for i in l2:
    p=p+i
print("sum of first list=",s)
print("sum of second list=",p)
if(p==s):
    print("the sum of values of both list are same")
else:
    print("the sum of values of both list are not same")
```

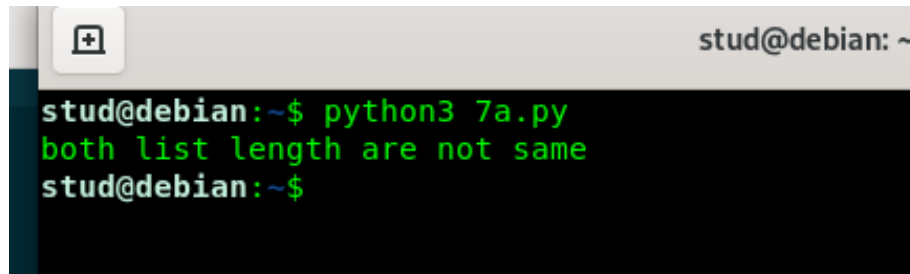
**(c)**

```
l1=[4,6,8,9]
l2=[5,7,9]
f=0
for i in l1:
    if(i in l2):
        print("values are in both is",i)
        f=f+1

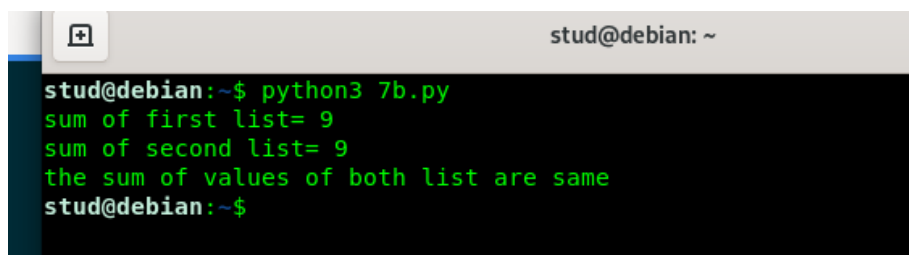
if(f==0):
```

```
print("no common values are in both")
```

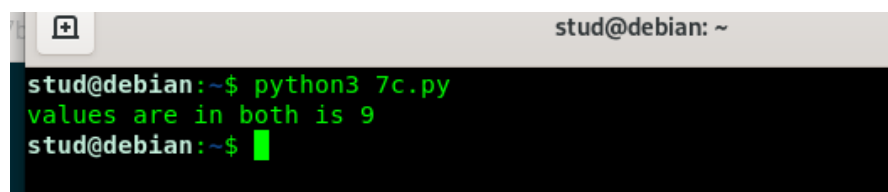
**OUTPUT:**



```
stud@debian: ~  
stud@debian:~$ python3 7a.py  
both list length are not same  
stud@debian:~$
```



```
stud@debian: ~  
stud@debian:~$ python3 7b.py  
sum of first list= 9  
sum of second list= 9  
the sum of values of both list are same  
stud@debian:~$
```



```
stud@debian: ~  
stud@debian:~$ python3 7c.py  
values are in both is 9  
stud@debian:~$
```

## **PROGRAM 7**

### **AIM:**

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 the string:")  
print("orginal string:",str1)  
char=str1[0]  
str1=str1.replace(char,'$')  
str1=char+str1[1:]  
print("replaced string=",str1)
```

### OUTPUT:

A terminal window titled 'stud@debian: ~/Documents/christy/pyt' showing the execution of a Python script. The prompt is 'stud@debian:~/Documents/christy/pyt\$'. The user enters 'python3 8.py'. The script prompts 'enter the string: onion', then outputs 'orginal string: onion' and 'replaced string= oni\$'. The prompt returns to 'stud@debian:~/Documents/christy/pyt\$'.

```
stud@debian: ~/Documents/christy/pyt
stud@debian:~/Documents/christy/pyt$ python3 8.py
enter the string: onion
orginal string: onion
replaced string= oni$
stud@debian:~/Documents/christy/pyt$
```

### PROGRAM 8

#### AIM:

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)
ns=t1+s[1:n-1]+t
print("given string:",s)
print("exchanged is:",ns)
```

### OUTPUT:

A terminal window titled 'stud@debian: ~/Documents/christy' showing the execution of a Python script. The prompt is 'stud@debian:~/Documents/christy/pyt\$'. The user enters 'python3 9.py'. The script outputs 'given string: python' and 'exchanged is: nythop'. The prompt returns to 'stud@debian:~/Documents/christy/pyt\$'.

```
stud@debian: ~/Documents/christy
stud@debian:~/Documents/christy/pyt$ python3 9.py
given string: python
exchanged is: nythop
stud@debian:~/Documents/christy/pyt$
```



### **PROGRAM 9**

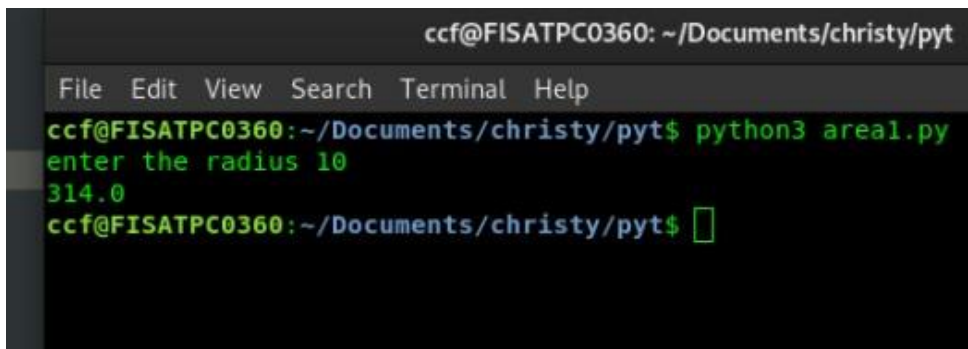
#### **AIM:**

Accept the radius from user and find area of circle.

#### **SOURCE CODE:**

```
area of circle
a=int(input("enter the radius "))
c=3.14*a*a
print(c)
```

#### **OUTPUT:**

A screenshot of a terminal window with a dark background. The title bar at the top reads 'ccf@FISATPC0360: ~/Documents/christy/pyt'. Below the title bar is a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal shows the command 'python3 areal.py' being executed. The prompt 'ccf@FISATPC0360:~/Documents/christy/pyt\$' is followed by the input 'enter the radius 10'. The output '314.0' is displayed on the next line. The prompt is then followed by a green cursor block.

```
ccf@FISATPC0360: ~/Documents/christy/pyt
File Edit View Search Terminal Help
ccf@FISATPC0360:~/Documents/christy/pyt$ python3 areal.py
enter the radius 10
314.0
ccf@FISATPC0360:~/Documents/christy/pyt$ █
```

### **PROGRAM 10**

#### **AIM:**

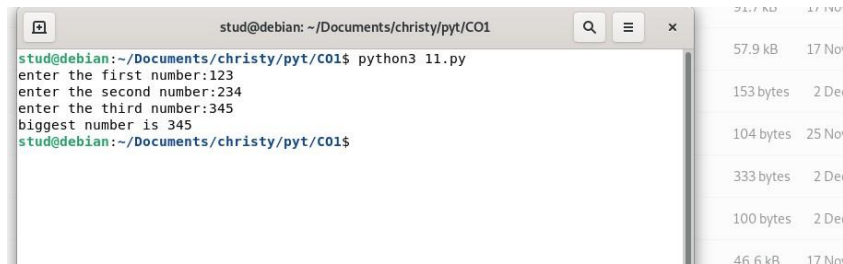
Find biggest of 3 numbers entered.

#### **SOURCE CODE:**

```
a=int(input("enter the first number:"))
b=int(input("enter the second number:"))
c=int(input("enter the third number:"))
#check a
if(a>b):
```

```
if(a>c):  
    print("biggest number is",a)  
else:  
    print("biggest number is",c)  
#check b  
else:  
    if(b>c):  
        print("biggest number is",b)  
    else:  
        print("biggest number is",c)
```

**OUTPUT:**



```
stud@debian: ~/Documents/christy/pyt/CO1  
stud@debian:~/Documents/christy/pyt/CO1$ python3 11.py  
enter the first number:123  
enter the second number:234  
enter the third number:345  
biggest number is 345  
stud@debian:~/Documents/christy/pyt/CO1$
```

**PROGRAM 11**

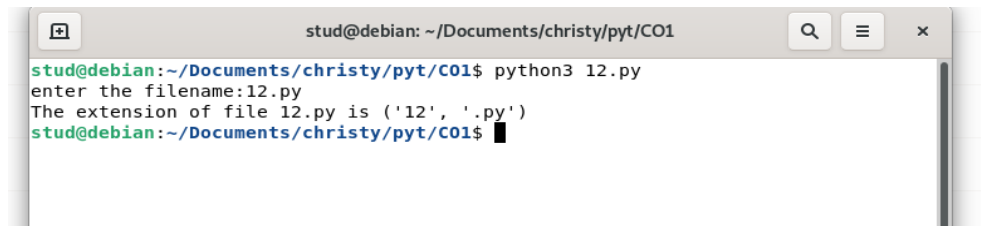
**AIM:**

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

**SOURCE CODE:**

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

## OUTPUT:



```
stud@debian: ~/Documents/christy/pyt/CO1$ python3 12.py
enter the filename:12.py
The extension of file 12.py is ('12', '.py')
stud@debian: ~/Documents/christy/pyt/CO1$
```

## PROGRAM 12

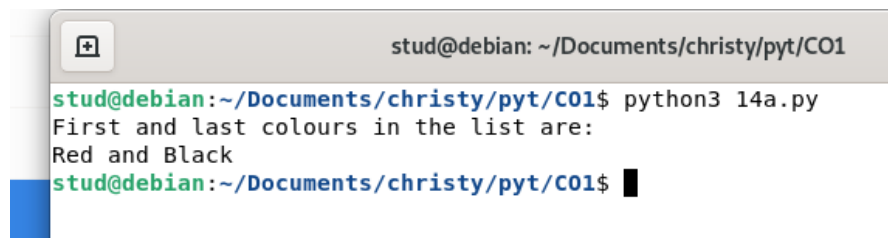
### AIM:

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

### SOURCE CODE:

```
l1=['Red','blue','white','yellow','Black']
print("First and last colours in the list are:")
print(l1[0],'and',l1[-1])
```

## OUTPUT:



```
stud@debian: ~/Documents/christy/pyt/CO1$ python3 14a.py
First and last colours in the list are:
Red and Black
stud@debian: ~/Documents/christy/pyt/CO1$
```

## PROGRAM 13

### AIM:

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

### SOURCE CODE:

```
a=int(input("enter the integer:"))
x=str(a)
```

```
print(" ",x)
y=x+x
print("",y)
Z=X+X+X
print(z)
s=a+int(y)+int(z)
print(s)
```

**OUTPUT:**



```
stud@debian: ~/Documents/christy/pyt/CO1
stud@debian:~/Documents/christy/pyt/CO1$ python3 14.py
enter the integer:2
2
22
222
246
stud@debian:~/Documents/christy/pyt/CO1$
```

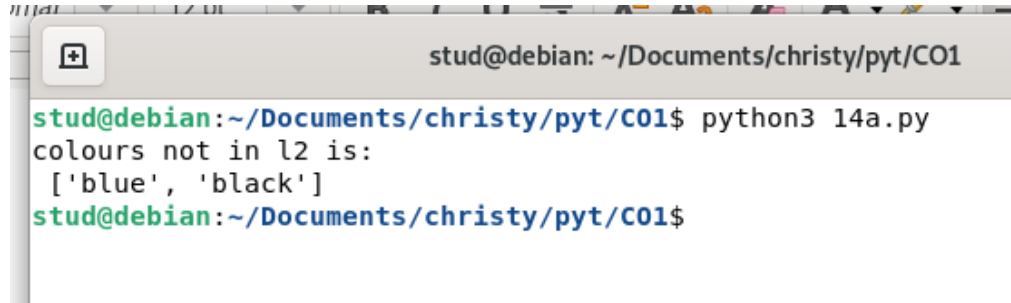
**PROGRAM 14**

**AIM:**Print out all colors from color-list1 not contained in color-list2.

**SOURCE CODE:**

```
l1=['red','blue','black']
l2=['red','white','pink'] l3=[]
    for i in l1:
        if i not in l2:
            l3.append(i) print('colours not in
l2 is:\n',l3)
```

## OUTPUT



```
stud@debian: ~/Documents/christy/pyt/CO1$ python3 14a.py
colours not in l2 is:
['blue', 'black']
stud@debian: ~/Documents/christy/pyt/CO1$
```

## PROGRAM 15

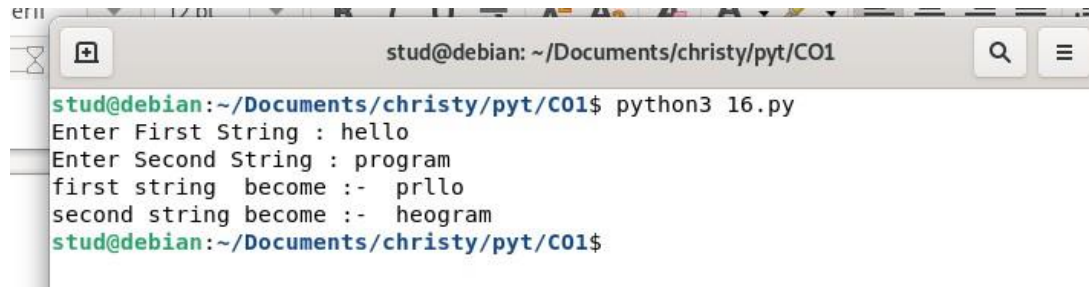
### AIM :

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

### SOURCE CODE:

```
#take strings from user
str1 = input("Enter First String : ")
str2 =input("Enter Second String : ")
x=str1[0:2]
str1=str1.replace(str1[0:2],str2[0:2])
str2=str2.replace(str2[0:2],x)
print("first string become :- ",str1)
print("second string become :- ",str2)
```

## OUTPUT:

A screenshot of a terminal window titled 'stud@debian: ~/Documents/christy/pyt/CO1'. The terminal shows the execution of a Python script '16.py'. The user enters 'hello' for the first string and 'program' for the second string. The program outputs 'first string become :- prllo' and 'second string become :- heogram'.

```
stud@debian:~/Documents/christy/pyt/CO1$ python3 16.py
Enter First String : hello
Enter Second String : program
first string become :- prllo
second string become :- heogram
stud@debian:~/Documents/christy/pyt/CO1$
```

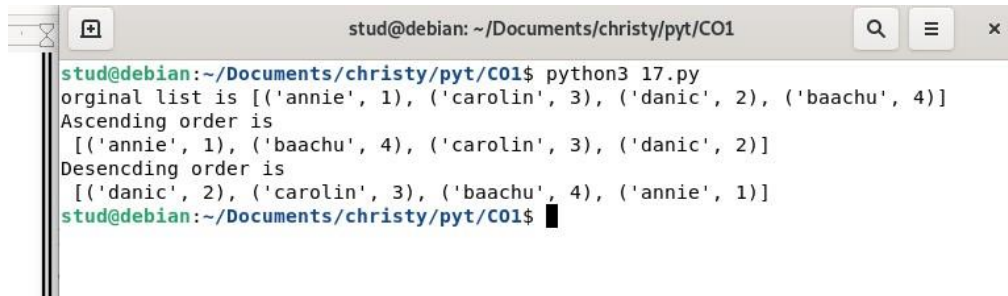
## PROGRAM 16

**AIM:**Sort dictionary in ascending and descending order.

### **SOURCE CODE;**

```
d1={"annie":1,"carolin":3,"danic":2,"baachu":4}
l=list(d1.items())
print("orginal list is",l)
l.sort()
print("Ascending order is\n",l)
l=list(d1.items())
l.sort(reverse=True)
print("Desencding order is\n",l)
```

## OUTPUT:



```
stud@debian: ~/Documents/christy/pyt/CO1
stud@debian:~/Documents/christy/pyt/CO1$ python3 17.py
original list is [('annie', 1), ('carolin', 3), ('danic', 2), ('baachu', 4)]
Ascending order is
[('annie', 1), ('baachu', 4), ('carolin', 3), ('danic', 2)]
Desencding order is
[('danic', 2), ('carolin', 3), ('baachu', 4), ('annie', 1)]
stud@debian:~/Documents/christy/pyt/CO1$
```

## PROGRAM 17

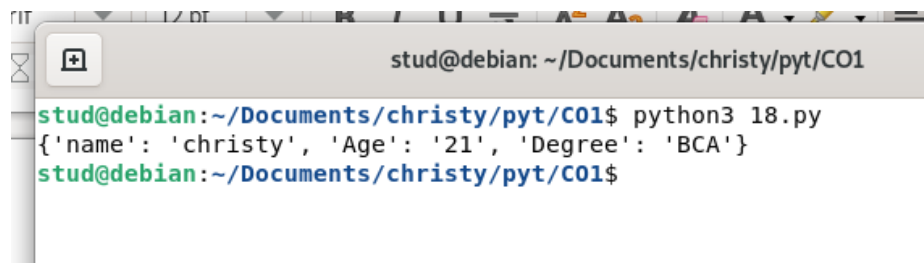
### AIM:

Merge two dictionaries

### SOURCE CODE:

```
dic={'name':"christy",'Age':'21'}
dic2={'Degree':"BCA"}
dic.update(dic2)
print(dic)
```

## OUTPUT;



```
stud@debian: ~/Documents/christy/pyt/CO1
stud@debian:~/Documents/christy/pyt/CO1$ python3 18.py
{'name': 'christy', 'Age': '21', 'Degree': 'BCA'}
stud@debian:~/Documents/christy/pyt/CO1$
```

### **PROGRAM 18**

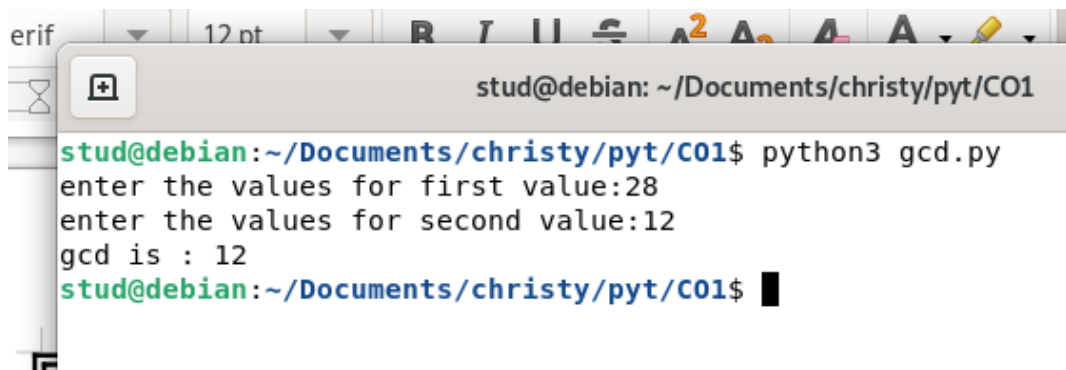
#### **AIM:**

Find gcd of 2 numbers.

#### **SOURCE Code:**

```
a=int(input("enter the values for first value:"))
b=int(input("enter the values for second value:"))
if(a<b):
    x=a
else:
    x=b
for i in range(1,x+1):
    if(x%i==0 and b%i==0):
        gcd=i
print("gcd is :",gcd)
```

#### **OUTPUT;**

A screenshot of a terminal window on a Linux system. The window title is "stud@debian: ~/Documents/christy/pyt/CO1". The terminal shows the command "python3 gcd.py" being executed. The program prompts for two values: "enter the values for first value:28" and "enter the values for second value:12". The output of the program is "gcd is : 12". The prompt "stud@debian:~/Documents/christy/pyt/CO1\$" is visible at the bottom of the terminal window.

```
stud@debian:~/Documents/christy/pyt/CO1$ python3 gcd.py
enter the values for first value:28
enter the values for second value:12
gcd is : 12
stud@debian:~/Documents/christy/pyt/CO1$
```



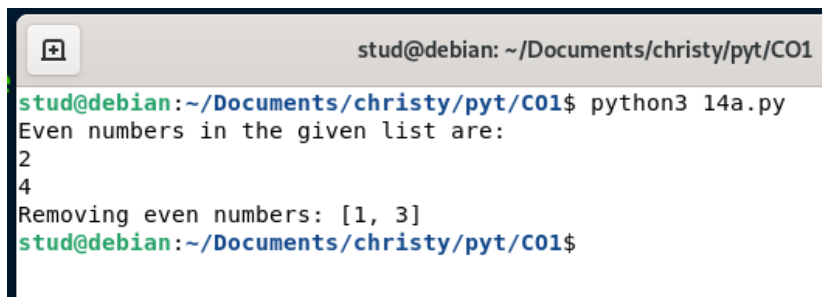
### **PROGRAM 19**

**AIM:**From a list of integers, create a list removing even numbers

**SOURCE CODE;**

```
l1=[1,2,3,4]
l3=[]
print("Even numbers in the given list are:") for i in l1:
    if(i%2==0):
        print(i)
else:
    l3.append(i)
print("Removing even numbers:",l3)
```

**OUTPUT:**



```
stud@debian: ~/Documents/christy/pyt/CO1
stud@debian:~/Documents/christy/pyt/CO1$ python3 14a.py
Even numbers in the given list are:
2
4
Removing even numbers: [1, 3]
stud@debian:~/Documents/christy/pyt/CO1$
```

### **PROGRAM 20**

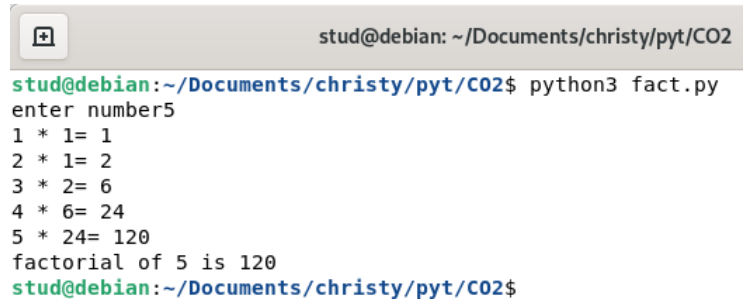
**AIM:**Program to find the factorial of a number.

**SOURCE CODE;**

```
a=int(input("enter number"))
f=1
for i in range(1,a+1):
    print(i,"*",f,end="")
```

```
f=f*i  
print("=",f)  
print("factorial of",a,"is",f)
```

**OUTPUT:**



```
stud@debian: ~/Documents/christy/pyt/CO2  
stud@debian:~/Documents/christy/pyt/CO2$ python3 fact.py  
enter number5  
1 * 1= 1  
2 * 1= 2  
3 * 2= 6  
4 * 6= 24  
5 * 24= 120  
factorial of 5 is 120  
stud@debian:~/Documents/christy/pyt/CO2$
```

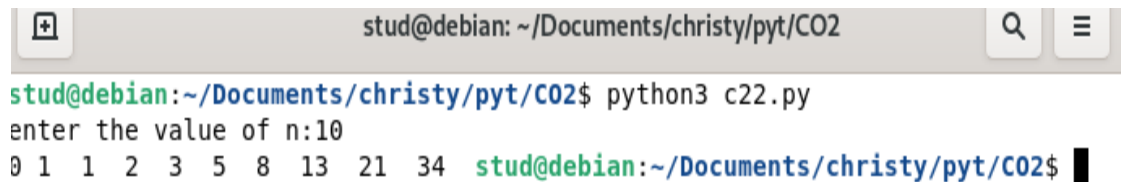
**PROGRAM 21**

**AIM:**Generate Fibonacci series of N terms.

**SOURCE CODE:**

```
#4fibnocci series  
f=int(input("enter the value of n:"))  
a=0  
b=1  
print(a,b," ",end="")  
for i in range (0,f-2):  
    fib=a+b  
    print(fib," ",end="")  
    a=b  
    b=fib
```

## OUTPUT:



```
stud@debian: ~/Documents/christy/pyt/CO2
stud@debian:~/Documents/christy/pyt/CO2$ python3 c22.py
enter the value of n:10
0 1 1 2 3 5 8 13 21 34 stud@debian:~/Documents/christy/pyt/CO2$
```

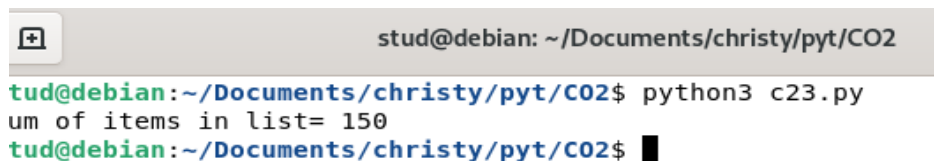
## PROGRAM 22

**AIM:**Find the sum of all items in a list.

### SOURCE CODE:

```
list1=[10,20,30,40,50]
sum=0
for i in list1:
    sum=sum+i
print("sum of items in list=",sum)
```

## OUTPUT:



```
stud@debian: ~/Documents/christy/pyt/CO2
tud@debian:~/Documents/christy/pyt/CO2$ python3 c23.py
um of items in list= 150
tud@debian:~/Documents/christy/pyt/CO2$
```

## PROGRAM 23

### AIM:

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=2000
limit2=8000
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:



```
stud@debian: ~/Documents/christy/pyt/CO2
stud@debian:~/Documents/christy/pyt/CO2$ python3 c24.py
: 68
78
80
[4624, 6084, 6400]
stud@debian:~/Documents/christy/pyt/CO2$
```

### **PROGRAM 24**

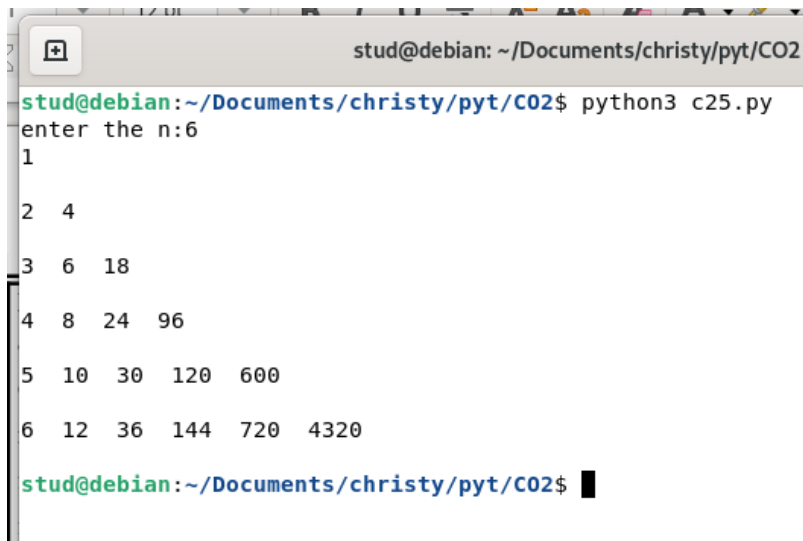
#### **AIM:**

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

#### **SOURCE CODE:**

```
n=int(input("enter the n:"))
for i in range(1,n+1):
    for j in range(1,i+1):
        i=i*j
    print(i," ",end="")
    print("\n")
```

#### **OUTPUT:**



```
stud@debian: ~/Documents/christy/pyt/CO2
stud@debian:~/Documents/christy/pyt/CO2$ python3 c25.py
enter the n:6
1
2 4
3 6 18
4 8 24 96
5 10 30 120 600
6 12 36 144 720 4320
stud@debian:~/Documents/christy/pyt/CO2$
```

### **PROGRAM 25**

**AIM:**Count the number of characters (character frequency) in a string.

**SOURCE CODE:**

```
string="The world's best place is family"
count=0
for i in range(0,len(string)):
    if(string!=""):
        count=count+1
print("Total no:of charcters is",count)
```

**OUTPUT:**



```
stud@debian: ~/Documents/christy/pyt/CO2
stud@debian:~/Documents/christy/pyt/CO2$ python3 c26.py
Total no:of charcters is 32
stud@debian:~/Documents/christy/pyt/CO2$
```

### **PROGRAM 26**

**AIM:**

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

**SOURCE CODE:**

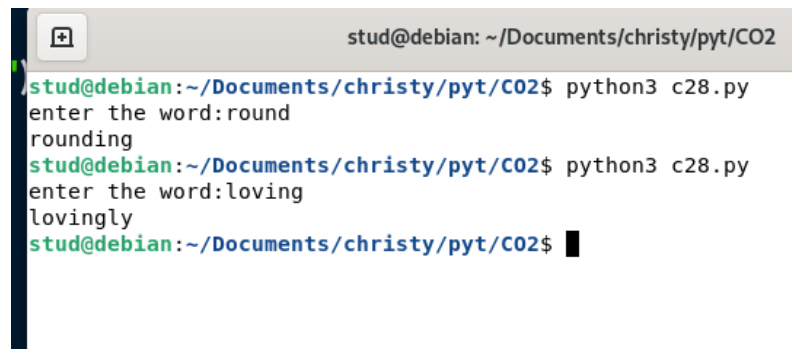
```
w=input("enter the word:")
l=len(w)
l1=w[l-3:l]
if(l1=='ing'):
    s=w+"ly"
```

else:

s=w+"ing"

print(s)

**OUTPUT:**



```
stud@debian: ~/Documents/christy/pyt/CO2
stud@debian:~/Documents/christy/pyt/CO2$ python3 c28.py
enter the word:round
rounding
stud@debian:~/Documents/christy/pyt/CO2$ python3 c28.py
enter the word:loving
lovingly
stud@debian:~/Documents/christy/pyt/CO2$
```

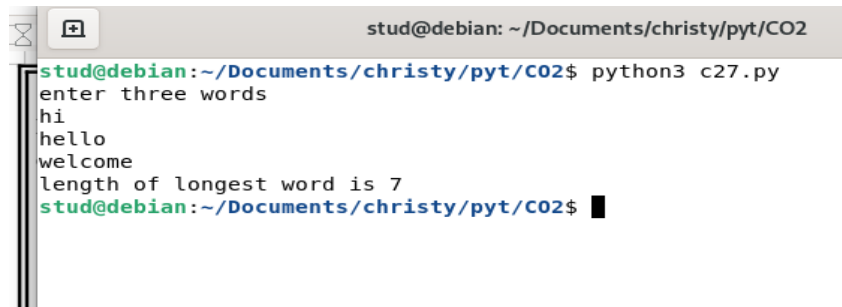
### **PROGRAM 27**

**AIM:**Accept a list of words and return length of longest word.

**SOURCE CODE:**

```
list1=[]
length=[]
print("enter three words")
for n in range(3):
    str=input()
    list1.append(str)
for m in list1:
    length.append(len(m))
print("length of longest word is",max(length))
```

## OUTPUT:

A terminal window titled 'stud@debian: ~/Documents/christy/pyt/CO2'. The prompt is 'stud@debian:~/Documents/christy/pyt/CO2\$'. The user enters 'python3 c27.py'. The program outputs 'enter three words', then 'hi', 'hello', and 'welcome' on separate lines. It then outputs 'length of longest word is 7'. The prompt returns to 'stud@debian:~/Documents/christy/pyt/CO2\$' with a cursor.

```
stud@debian: ~/Documents/christy/pyt/CO2$ python3 c27.py
enter three words
hi
hello
welcome
length of longest word is 7
stud@debian:~/Documents/christy/pyt/CO2$
```

## PROGRAM 28

**AIM :**Construct following pattern using nested loop.

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

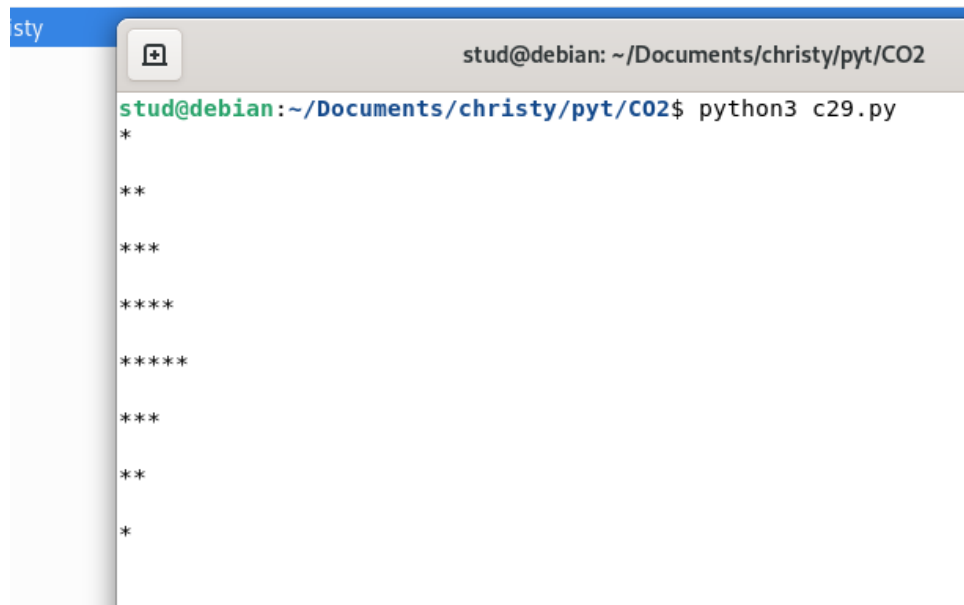
## SOURCE CODE:

```
for i in range(1,6):
    for j in range(1,i+1):
        print("*",end="")
    print("\n")
for i in range(5,0,-1):
    for j in range(1,i-1):
        print("*",end="")
```



```
print("\n")
```

### OUTPUT:



```
stud@debian: ~/Documents/christy/pyt/CO2
stud@debian:~/Documents/christy/pyt/CO2$ python3 c29.py
*
**
***
****
*****
****
***
**
*
```

### PROGRAM 29

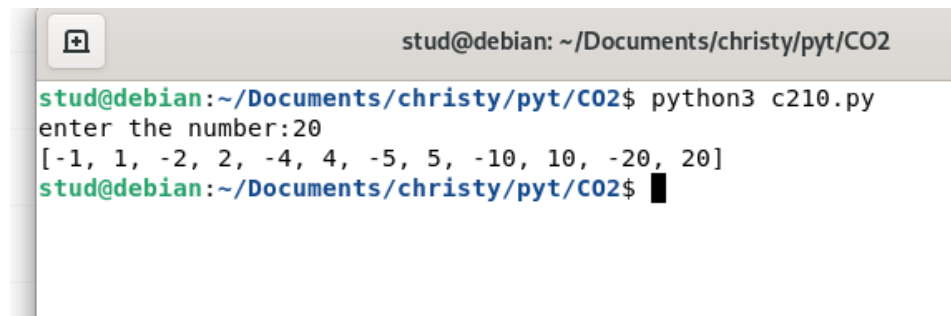
#### AIM:

Generate all factors of a number.

#### SOURCE CODE:

```
number=int(input("enter the number:"))
factors=[]
for n in range(1,number+1):
    if number % n == 0:
        factors.append(-n)
        factors.append(n)
print(factors)
```

## OUTPUT:



```
stud@debian: ~/Documents/christy/pyt/CO2
stud@debian:~/Documents/christy/pyt/CO2$ python3 c210.py
enter the number:20
[-1, 1, -2, 2, -4, 4, -5, 5, -10, 10, -20, 20]
stud@debian:~/Documents/christy/pyt/CO2$
```

## PROGRAM 30

### AIM:

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.

### SOURCE 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("Permeter 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 spere with radius 10 is : ",sphere.area_sphere(10))
print("Permeter of a spere with radius 10 is ",sphere.perimeter_sphere(10))
```

### **OUTPUT;**

```
C:\Users\hp\Desktop\python1>md graphics
C:\Users\hp\Desktop\python1>cd graphics
C:\Users\hp\Desktop\python1\graphics>notepad circle.py
C:\Users\hp\Desktop\python1\graphics>notepad rectangle.py
C:\Users\hp\Desktop\python1\graphics>md tdgraphics
C:\Users\hp\Desktop\python1\graphics>cd tdgraphics
C:\Users\hp\Desktop\python1\graphics\tdgraphics>notepad cuboid.py
C:\Users\hp\Desktop\python1\graphics\tdgraphics>notepad sphere.py
```

```
C:\Users\hp\Desktop\python1>python driver.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
```

### **PROGRAM 31**

#### **AIM:**

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

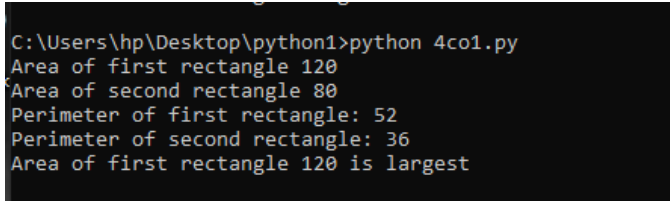
#### **SOURCE CODE:**

```
class Rectangle:
    def __init__(self,l,b):
        self.l=l
        self.b=b
    def area(self):
        return self.l*self.b
    def peri(self):
        return 2*(self.l+self.b)

r1=Rectangle(20,6)
r2=Rectangle(8,10)
```

```
x=r1.area()
print(&quot;Area of first rectangle&quot;,x)
y=r2.area()
print(&quot;Area of second rectangle&quot;,y)
x1=r1.peri()
print(&quot;Perimeter of first rectangle:&quot;,x1)
y1=r2.peri()
print(&quot;Perimeter of second rectangle:&quot;,y1)
if(x>y):
print(&quot;Area of first rectangle&quot;,x,&quot;is largest&quot;,)
else:
print(&quot;Area of second rectangle&quot;,y,&quot;is largest&quot; )
```

## OUTPUT



```
C:\Users\hp\Desktop\python1>python 4co1.py
Area of first rectangle 120
Area of second rectangle 80
Perimeter of first rectangle: 52
Perimeter of second rectangle: 36
Area of first rectangle 120 is largest
```

## PROGRAM 32

### AIM:

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,acno,name,actype,bal):
        self.acno=acno
```

```
        self.name=name
        self.actype=actype
        self.bal=bal
    def deposit(self,amt):
        self.bal=self.bal+amt
        print("Balance after deposite:",self.bal)
    def withdraw(self,amt):
        self.bal=self.bal-amt
        print("After withdraw:",self.bal)
    def display(self):
        print(".....")
        print("Account no:",self.acno)
        print("Name:",self.name)
        print("Account type:",self.actype)
        print("Balance amount:",self.bal)
b1=Bank(123,"Athira","Savings",10000)
b2=Bank(456,"Ryan","Savings",20000)
b1.display()
b1.deposit(5000)
b1.withdraw(500)
b2.display()
b2.deposit(2000)
b2.withdraw(1000)
```

**OUTPUT:**

```
C:\Users\hp\Desktop\python1>python 4co2.py
.....
Account no: 123
Name: Athira
Account type: Savings
Balance amount: 10000
Balance after deposit: 15000
After withdraw: 14500
.....
Account no: 456
Name: Ryan
Account type: Savings
Balance amount: 20000
Balance after deposit: 22000
After withdraw: 21000
```

### **PROGRAM 33**

**AIM:** 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 area(self):
        a=self.length*self.breadth
        print("\nArea=",a)
        return a

    """def perimeter(self):
        p=2*(self.length+self.breadth)
        print("Perimeter=",p,"\n")
        return p"""

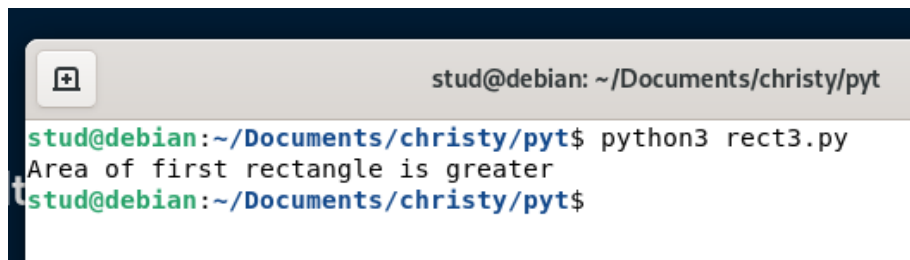
    def __lt__(self,rr):
        if(self.length*self.breadth>rr.length*rr.breadth):
```



```
        return True
    else:
        return False

r1=Rectangle(10,5)
r2=Rectangle(20,2)
if(r1<r2):
    print("Area of first rectangle is greater")
else:
    print("Area of second rectangle is greater")
```

#### OUTPUT:



```
stud@debian: ~/Documents/christy/pyt
stud@debian:~/Documents/christy/pyt$ python3 rect3.py
Area of first rectangle is greater
stud@debian:~/Documents/christy/pyt$
```

#### PROGRAM 34

##### AIM:

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, hour, minute, second):
        self.__hr=hour
        self.__min=minute
        self.__sec=second
```

```
def __add__(self,ad):  
    x=self.__hr+ad.__hr  
    y=self.__min+ad.__min  
    z=self.__sec+ad.__sec  
    print("t1+t2 is ",x,":",y,":",z)
```

```
t1=Time(2,10,20)
```

```
t2=Time(1,20,5)
```

```
t1+t2
```

**OUTPUT:**

```
C:\Users\hp\Desktop\python1>python 4co4.py  
t1+t2 is  3 : 30 : 25
```

**PROGRAM 35**

**AIM:**

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.name)
```

```
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","Python","Balaguru Swami",200,800)
p.display3()
```

**OUTPUT:**

```
C:\Users\hp\Desktop\python1>python 4co5.py
ABC Publications
Python
Balaguru Swami
200
800
```

### **PROGRAM 36**

**AIM:** Write a Python program to read a file line by line and store it into a list.

#### **SOURCE CODE:**

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

#### **OUTPUT:**

```
PS C:\Users\HP\OneDrive\Desktop\python\co5> python qn1.py
["Cats, also called domestic cats are small, carnivorous mammals, of the family Felidae.", "Domestic cat
s are often called 'house cats' when kept as indoor pets.", 'Cats have been domesticated for nearly 10,00
0 years.', 'They are one of the most popular pets in the world.']
PS C:\Users\HP\OneDrive\Desktop\python\co5> █
```

### **PROGRAM 37**

#### **AIM:**

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

#### **SOURCE CODE:**

```
import csv

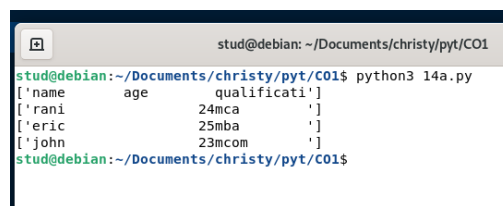
with open("profession.csv","r")as file:

    reader=csv.reader(file)

    for row in reader:

        print(row)
```

#### **OUTPUT:**

A terminal window screenshot showing the execution of a Python script. The terminal title is 'stud@debian: ~/Documents/christy/pyt/CO1'. The command entered is 'python3 14a.py'. The output is a list of lists representing rows from a CSV file: [['name', 'age', 'qualificati'], ['rani', '24mca', ''], ['eric', '25mba', ''], ['john', '23mcom', '']]. The prompt returns to 'stud@debian: ~/Documents/christy/pyt/CO1\$'.

```
stud@debian: ~/Documents/christy/pyt/CO1
stud@debian:~/Documents/christy/pyt/CO1$ python3 14a.py
[['name', 'age', 'qualificati'],
 ['rani', '24mca', ''],
 ['eric', '25mba', ''],
 ['john', '23mcom', '']]
stud@debian:~/Documents/christy/pyt/CO1$
```

