**FEDERAL INSTITUTE OF**

**SCIENCE AND TECHNOLOGY**

**(FISAT)TM**

**HORMIS NAGAR, MOOKKANNOOR**

**ANGAMALY-683577**



‘**FOCUS ON EXCELLENCE’**

**PYTHON PROGRAMMING LAB**

…………………………………………………………………..

**LABORATORY RECORD**

**Name: ABHINAV H**

**Branch: MASTER OF COMPUTER APPLICATIONS**

**Semester: 1 Batch: SEMESTER -1 A Roll No: 03**

**FEDERAL INSTITUTE OF**

**SCIENCE AND TECHNOLOGY**

**(FISAT)TM**

**HORMIS NAGAR, MOOKKANNOOR**

**ANGAMALY-683577**



‘**FOCUS ON EXCELLENCE’**

**Name :ABHINAV H**

**Branch : MASTER OF COMPUTER APPLICATION**

**Semester : 1 Roll No: 03**

**University Exam.Reg. No: FIT21MCA-2003**

**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**and submitted to* Kerala Technological University *in partial fulfillment for the award of the Master Of Computer Applications is a record of the original research work done by* ***ABHINAV H*** *in the* ***PYTHON*** *Laboratory of the Federal Institute of Science and Technology during the academic year 2021-2022.*

Signature of Staff in Charge Signature of H.O.D

Name: JOICE T Name: DEEPA MARY MATHEWS

Date:

**Date of University practical examination ………………………**

Signature of Signature of

Internal Examiner 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. |  |  |
| **2** |  | Generate positive list of numbers from a given list of integers. |  |  |
| **3** |  | Print square of N numbers. |  |  |
| **4** |  | Form a list of vowels selected from a given word. |  |  |
| **5** |  | List ordinal value of each element of a word (Hint: use ord() to get ordinal values) |  |  |
| **6** |  | Count the occurrences of each word in a line of text. |  |  |
| **7** |  | Prompt the user for a list of integers. For all values greater than 100, store ‘over’ instead. |  |  |
| **8** |  | Store a list of first names. Count the occurrences of ‘a’ within the list |  |  |
| **9** |  | Get a string from an input string where all occurrences of first character replaced with “$’, except first character. |  |  |
| **10** |  | Create a string from given string where first and last characters exchanged. |  |  |
| **11** |  | Accept the radius from user and find area of circle. |  |  |
| **12** |  | Find biggest of 3 numbers entered. |  |  |
| **13** |  | Accept a file name from user and print extension of that. |  |  |
| **14** |  | Create a list of colors from comma-separated color names entered by user. Display first and last colors. |  |  |
| **15** |  | Accept an integer n and compute n+nn+nnn. |  |  |
| **16** |  | Print out all colors from color-list1 not contained in color-list2. |  |  |
| **17** |  | Create a single string separated with space from two strings by swapping the character at position 1. |  |  |
| **18** |  | Merge two dictionaries. |  |  |
| **19** |  | Find gcd of 2 numbers. |  |  |
| **20** |  | From a list of integers, create a list removing even numbers. |  |  |
| **21** |  | Program to find the factorial of a number. |  |  |
| **22** |  | Generate Fibonacci series of N terms. |  |  |
| **23** |  | Find the sum of all items in a list. |  |  |
| **24** |  | Generate a list of four-digit numbers in a given range with all their digits even and the number is a perfect square. |  |  |
| **25** |  | Display the given pyramid with the step number accepted from the user. |  |  |
| **26** |  | Count the number of characters (character frequency) in a string. |  |  |
| **27** |  | Add ‘ing’ at the end of a given string. If it already ends with ‘ing’, then add ‘ly’. |  |  |
| **28** |  | Accept a list of words and return length of longest word. |  |  |
| **29** |  | Construct following pattern using nested loop. |  |  |
| **30** |  | Generate all factors of a number. |  |  |
| **31** |  | Create a package graphics with modules rectangle, circle and sub-package 3D graphics with module 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. |  |  |
| **32** |  | Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare to rectangle objects by their area. |  |  |
| **33** |  | Create 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. |  |  |
| **34** |  | Create a class Rectangle with private attributes length and width. Overload ‘<’ operator to compare the area of two rectangles. |  |  |
| **35** |  | Create a class Time with private attributes hour, minute and second. Overload ‘+’ operator to find sum of two time. |  |  |
|  |  |  |  |  |
| **36** |  | Create a class Publisher(name). Derive class Book from Publisher with attributes title and author. Derive class python from Book with attributes price and number\_of\_pages. Write a program that displays information about a Python book. Use base class constructor invocation and method overreading. |  |  |
| **37** |  | Write a program to read a file line by line and store it into a list |  |  |
| **38** |  | Write a Python program to read each row from a given csv file and print a list of strings. |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**COURSE OUTCOME-1**

EXPERIMENT 1

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

**CODE:**

current\_year=int(input("Enter the current year:")) f

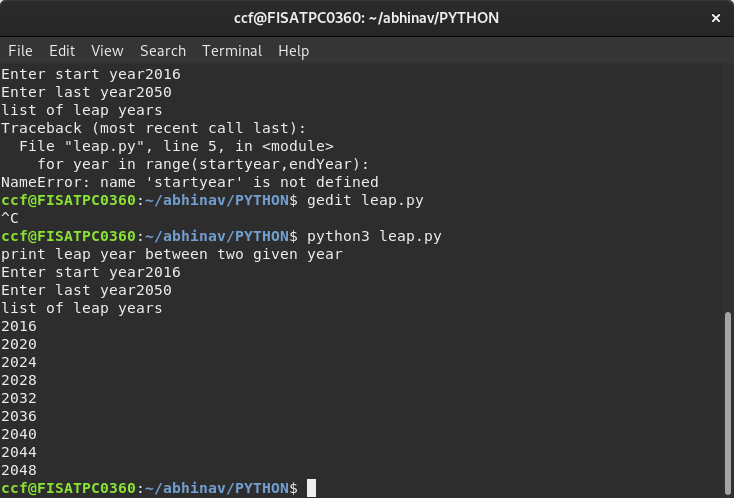
inal\_year=int(input("Enter the final year:"))

for year in range(current\_year,final\_year):

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

print(year)

**OUTPUT:**



EXPERIMENT 2

**AIM: List comprehensions:**

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

**CODE:**

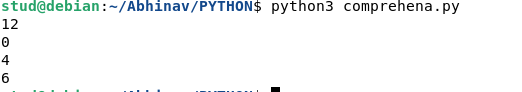
   list1=[12,-3,0,4,6]

for num in list1:

if(num>=0):

print(num)

**OUTPUT**



**AIM:**

**b.**Print square of N numbers.

**CODE:**

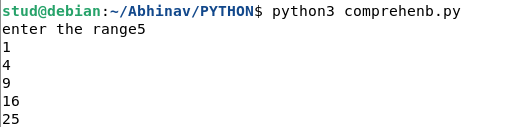
n=int(input("enter the range"))

   for num in range(1,n+1):

    num=num\*num

print(num)

**OUTPUT:**



**AIM:**

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

**CODE:**

s=input("Enter the word")

list=[]

for i in s:

if i in "aeiouAEIOU":

list.append(i)

print(list)

**OUTPUT:**

https://lh5.googleusercontent.com/xSo1QwIwTYJ_NkYh450dzN1r0PIhqcifgLNlBritA4zsoPJDcnyjSPK8uARt0utAfZQmffl3uXqD1nmwrrBmYjuSV7l-yO_exd2tBVNisP-keWDxo-Avr2AC5-GWLUWq7ZXDH-_fuyIzkoL9ow

**AIM:**

**d.**List ordinal value of each element of a word (Hint: use ord() to get ordinal values)

**CODE:**

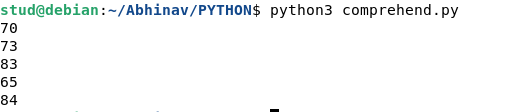
list=['F','I','S','A','T']

for i in range(0,5):

value=ord(list[i])

print(value)

**OUTPUT:**



EXPERIMENT 3

**AIM:** Count the occurrences of each word in a line of text.

**CODE:**

list1=[]

list2=[]

x=input("Enter a line of text:")

for i in x.split(" "):

list1.append(i)

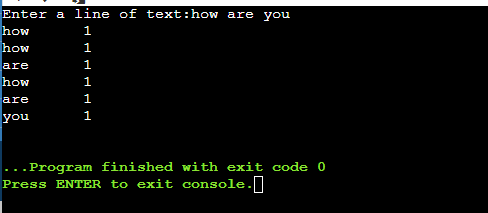
if i not in list2:

list2.append(i)

for i in list2:

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

**OUTPUT:**

****

EXPERIMENT 4

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

**CODE:**

   list=[]

n=int(input("Enter the limit:"))

print("Enter integer Numbers")

for i in range(0,n):

        j=int(input())

        if j>100:

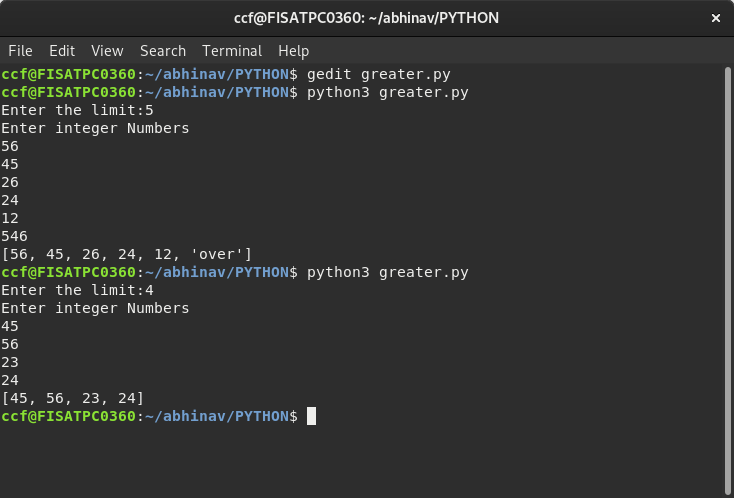
                        list.append("over")

        else:

                        list.append(j)

print(list)

**OUTPUT:**



EXPERIMENT 5

**AIM:** Store a list of first names. Count the occurrences of ‘a’ within the list

**CODE:**

list=['abhi','vyshnav','adrash'] print("Elements in the list are:")

print(list)

count=0

for word in list:

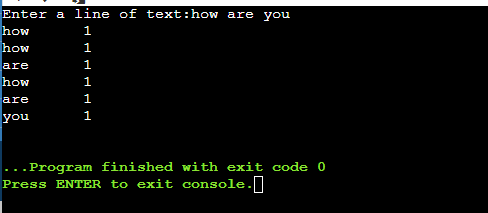
for i in word:

if i=='a':

count+=1

print("count of 'a' is:", count)

**OUTPUT:**

****

EXPERIMENT 6

1. **AIM:Enter 2 lists of integers.Check**
2. **whether list are of same length**
3. **whether list sums of same value**
4. **whether any value occur in both.**

**CODE:**

l1=[1,2,3,4]

l2=[1,3,2]

print("List 1",l1)

print("List 2",l2)

x=len(l1)

y=len(l2)

if x==y:

print("List are of same length")

else:

print("Length of lists are different")

s1=0

s2=0

for i in range(x):

s1=s1+l1[i]

print("Sum of elements of List1:",s1)

for j in range(y):

s2=s2+l2[j]

print("Sum of elememts of List2:",s2)

if s1==s2:

print("Sum of list elements is same")

else:

print("Sum of list elements is not same")

print("Common elements are:")

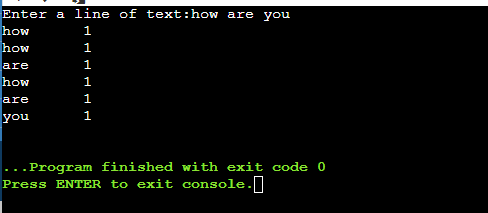
for i in range(x):

for j in range(y):

if l1[i]==l2[j]:

print(l1[i])

**OUTPUT:**

****

EXPERIMENT 7

1. **AIM: Get a string from an input string where all occurrences of first character replaced with ‘$’,except first character.[eg:onion->oni$n]**
2. **CODE:**

str=input("Enter a string: ")

print("Original string is: ",str)

char=str[0]

str=str.replace(char,'$')

str=char+str[1:]

print("String: ",str)

var=input("Enter a string: ")

beg=var[0]

end=var[len(var)-1]

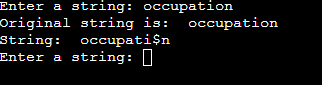
dum=beg

beg=end

end=dum

print(beg+var[1:len(var)-1]+end)

**OUTPUT:**

****

EXPERIMENT 8

**AIM:Create a string from given string where first and last characters exchanged. [eg:python->nythop]**

**Source code**

s=input("Enter a string: ")

t=s[0]

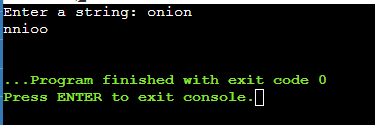
t1=s[-1]

n=len(s)

ns=t1+s[1:n-1]+t

print(ns)

**OUTPUT:**

****

EXPERIMENT 9

**AIM:** Accept the radius from user and find area of circle.

**CODE:**

x=input("enter the radious")

x=int(x)

a=3.14\*x\*x

print(a)

**OUTPUT:**



EXPERIMENT 10

**AIM:** Find biggest of 3 numbers entered**.**

**CODE:**

x=input("Enter the number")

y=input("Enter the number")

z=input("Enter the number")

x=int(x)

y=int(y)

z=int(z)

if(x>y):

if(x>z):

print("x is larger")

else:

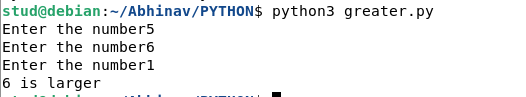
if(y>z):

print("y is larger")

else:

print("z is larger")

**OUTPUT:**



EXPERIMENT 11

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

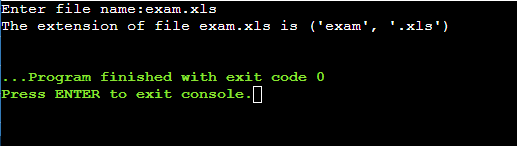
**CODE:**

import os

a=input("enter the filename : ")

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

**OUTPUT:**

****

EXPERIMENT 12

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

**CODE:**

colors=[]

str=(input("Enter color names:"))

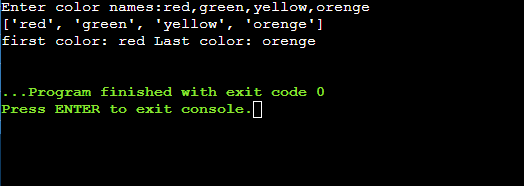
for i in str.split(','):

colors.append(i)

print(colors)

print("first color:",colors[0],"Last color:",colors[-1])

**OUTPUT:**

****

EXPERIMENT 13

**AIM:** Accept an integer n and compute n+nn+nnn.

**CODE:**

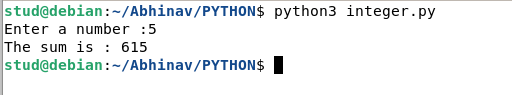
n=input("Enter a number :")

nn=n+n

nnn=nn+n

print("The sum is :",int(n)+int(nn)+int(nnn))

**OUTPUT:**



EXPERIMENT 14

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

**CODE:**

l1=['red','blue','green','black','yellow']

l2=['orenge','pink','red','brown']

l3=[]

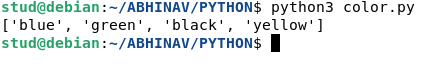
for i in l1:

if i not in l2:

l3.append(i)

print(l3)

**OUTPUT:**



EXPERIMENT 15

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

**CODE:**

string1="Fisat"

string2="Ankamaly"

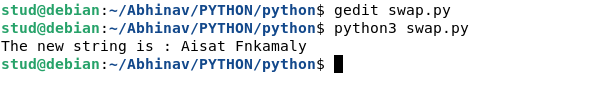
f1=string1[0]

f2=string2[0]

string=f2+string1[1:]+" "+f1+string2[1:]

print("The new string is :",string)

**OUTPUT:**



EXPERIMENT 16

**AIM:** Merge two dictionaries.

**CODE:**

D1={"Name":"Abhinav","Age":"22"}

print("Directory 1",D1)

D2={"Gender":"male","Qualification":"BCA"}

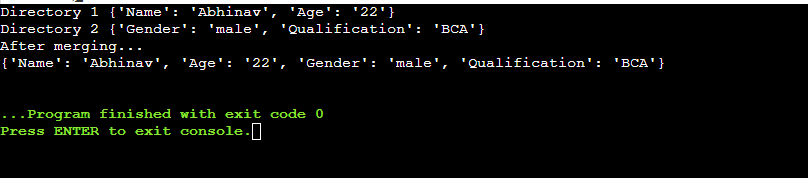
print("Directory 2",D2)

D1.update(D2)

print("After merging...")

print(D1)

**OUTPUT:**

****

EXPERIMENT 17

**AIM:** Find gcd of 2 numbers.

**CODE:**

x=int(input("Enter 1 st number"))

y=int(input("Enter 2nd number"))

if(x>y):

x1=y

else:

x1=x

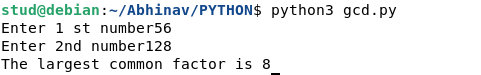
for i in range(1,x1+1):

if(x%i==0 and y%i==0):

gcd=i

print("The largest common factor is",gcd)

**OUTPUT:**



EXPERIMENT 18

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

**CODE:**

l1=[1,2,3,4,5,6,7,8,9,10]

print(l1)

l2=[]

for i in range(len(l1)):

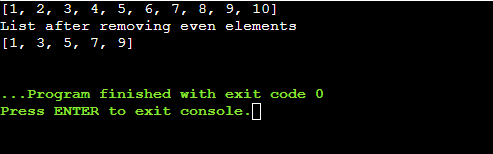
if l1[i]%2!=0:

l2.append(l1[i])

print("List after removing even elements")

print(l2)

**OUTPUT:**



**COURSE OUTCOME-2**

EXPERIMENT 19

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

**CODE**

n=int(input('Enter a number:'))

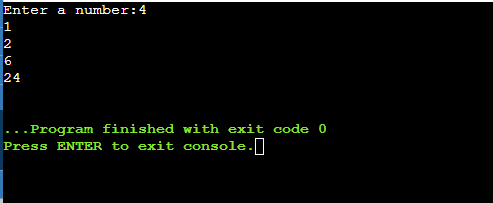
fact=1

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

fact=fact\*i

print(fact)

**OUTPUT**

****

EXPERIMENT 20

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

**CODE**

x=input("enter the number")

x=int(x)

f1=0

f2=1

count=0

for i in range(count,x):

print(f1)

f3=f1+f2

f1=f2

f2=f3

**OUTPUT**



EXPERIMENT 21

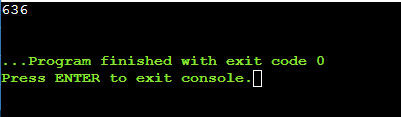
**AIM:** Find the sum of all items in a list

**CODE**

a=[15,58,66,-99,456,-66,95]

print(sum(a))

**OUTPUT**

****

EXPERIMENT 24

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

**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):

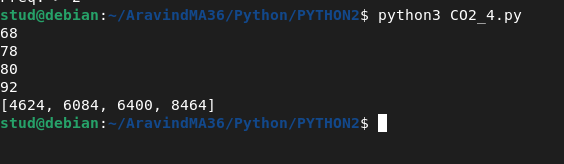
if((k\*\*2)==j):

list1.append(j)

print(k)

print(list1)

**OUTPUT**

****

EXPERIMENT 25

**AIM:** Display the given pyramid with the step number accepted from the user.

**Eg: N=4**

**1**

**2 4**

**3 6 9**

**4 8 12 16**

**CODE:**

ht=int(input("Enter height:"))

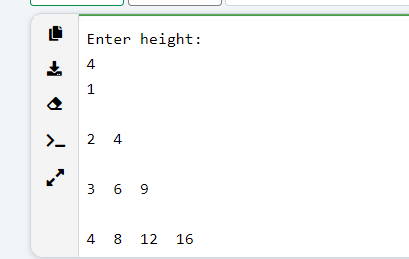
for i in range(1,ht+1):

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

print((j\*i)," ",end="")

print("\n")

**OUTPUT**

****

EXPERIMENT 26

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

**CODE**

**s**tr=input("Enter a string:")

fnd=input("Enter character:")

cnt=0

str=str.lower()

fnd=fnd.lower()

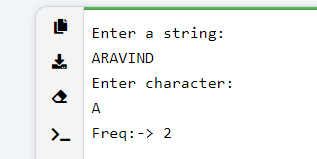
for i in str:

if i==fnd:

cnt=cnt+1

print("Freq:->",cnt)

**OUTPUT**

****

EXPERIMENT 27

**AIM:** Add ‘ing’ at the end of a given string. If it already ends with ‘ing’, then add ‘ly’

**CODE**

a=input("Enter a string:")

l=len(a)

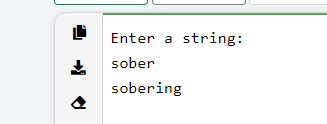
if a[l-3]=='i' and a[l-2]=='n' and a[l-1]=='g':

print(a+'ly')

else:

print(a+'ing')

**OUTPUT**

****

EXPERIMENT 28

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

**CODE**

cnt=int(input("Enter number of words:"))

a=[]

com=0

for i in range(0,cnt):

nw=input("Enter the word:")

a.append(nw)

for i in range(0,cnt):

cmp=len(a[i])

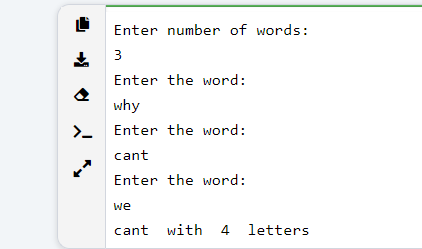
if cmp>com:

com=cmp

j=i

print(a[j]," with ",com," letters")

**OUTPUT**

****

EXPERIMENT 29

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

**\***

**\*\***

**\*\*\***

**\*\*\*\***

**\*\*\***

**\*\***

**\***

**CODE**

ht=int(input("Enter height:"))

for i in range(0,ht):

for j in range(0,i+1):

print("\* ",end="")

print("\n")

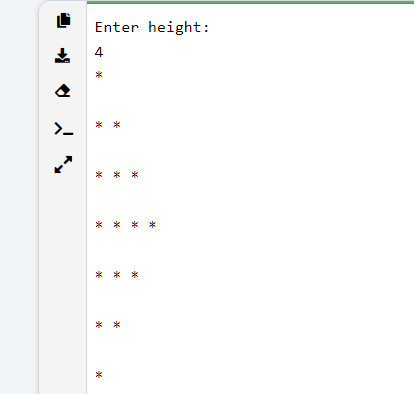
for k in range(0,ht):

for l in range(0,j-k):

print("\* ",end="")

print("\n")

**OUTPUT**

****

EXPERIMENT 30

**AIM:** Generate all factors of a number**.**

**CODE**

num=int(input("Enter a number:"))

fact=[]

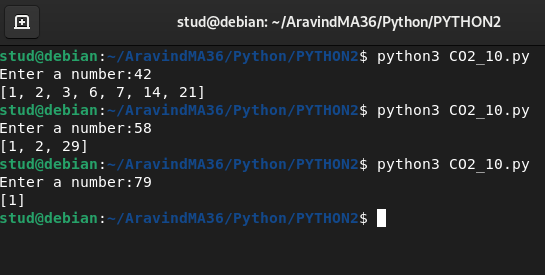
for i in range(1,num):

if num%i==0:

fact.append(i)

print(fact)

**OUTPUT**

****

**COURSE OUTCOME-3**

EXPERIMENT 31

**Aim:** Create a package graphics with modules rectangle, circle and sub-package 3D graphics with module 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.

**Terminal(Windows):**

mkdir graphics

cd graphics

notepad rectangle

notepad circle

notepad \_\_init\_\_.py

mkdir dgraphics

cd dgraphics

notepad \_\_init\_\_.py

notepad cuboid.py

notepad sphere.py

**CODE**

1. **Rectangle**

class Rectangle:

def \_\_init\_\_(self,length,width):

self.length=length

self.width=width

def area(self):

return (self.length\*self.width)

def perimeter(self):

return (2\*(self.length+self.width))

1. **Circle**

global pi pi=3.1416

class Circle:

global pi

pi=3.1416

def \_\_init\_\_(self,radius):

self.radius=radius

def area(self):

return (pi\*(self.radius\* self.radius))

def perimeter(self):

return (2\*pi\*self.radius)

1. **Sphere**

global pi

pi=3.1416

class Sphere:

def \_\_init\_\_(self,radius):

self.radius=radius

def volume(self):

r=self.radius return ((4/3)\*pi\*(r\*\*3))

def area(self):

r=self.radius

return (4\*pi\*(r\*\*2))

1. **Cuboid**

class Cuboid:

def \_\_init\_\_(self,length,width,height):

self.l=length

self.w=width

self.h=height

def volume(self):

return (self.l\*self.w\*self.h)

def area(self):

l=self.l

w=self.w

h=self.h

return (2\*((l\*w)+(w\*h)+(l\*h)))

**CODE**

from graphics import rectangle as rt

from graphics import circle

from graphics dgraphics import \*

r=rt.Rectangle(10,12) print("RECTANGLE\n")

print("length =",r.length)

print("width =",r.width)

print("area=",r.area())

print("perimeter=",r.perimeter())

c=circle.Circle(12)

print("CIRCLE\n")

print("radius =",c.radius)

print("area=",c.area())

print("perimeter=",c.perimeter())

s=sphere.Sphere(12)

print("SPHERE")

print("radius =",s.radius)

print("area=",s.area())

print("volume=",s.volume())

cu=cuboid.Cuboid(13,11,14) print("CUBOID\n")

print("length =",cu.l)

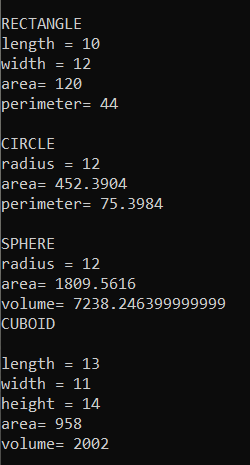
print("width =",cu.w)

print("height =",cu.h)

print("area=",cu.area())

print("volume=",cu.volume())

**OUTPUT**

****

**COURSE OUTCOME-4**

EXPERIMENT 32

**AIM:** Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare to rectangle objects by their area.

**CODE**

class Rectangle:

def \_\_init\_\_(self,ln,br):

self.ln=ln

self.br=br

def area(self):

p=self.ln\*self.br

return p

def perim(self):

q=2\*(self.ln+self.br)

return q

a=int(input("Enter length of the first rectangle:"))

b=int(input("Enter breadth of the first rectangle:"))

r1=Rectangle(a,b)

a=int(input("Enter length of the second rectangle:"))

b=int(input("Enter breadth of the second rectangle:"))

r2=Rectangle(a,b)

print("Perimeter of frist rectangle= ",r1.perim())

print("Perimeter of second rectangle= ",r2.perim())

m=r1.area()

n=r2.area()

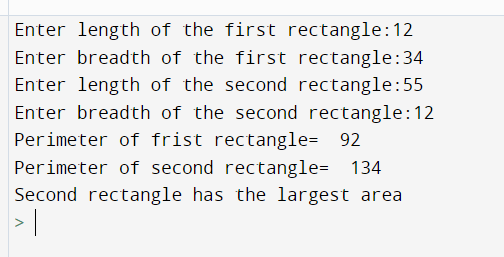
if m>n:

print("First rectangle has the largest area")

else:

print("Second rectangle has the largest area")

**OUTPUT**

****

EXPERIMENT 33

**AIM:** Create Bank account with members account number, name, type of account and balance. Write constructor and methods to deposite at the bank and withdraw an amount from the bank.

**CODE**

bnkno=1000

holders=[10]

ptr=0

class Bank:

def \_\_init\_\_(self,accno,accnme,acctype,accbal):

self.accno=accno

self.accnme=accnme

self.acctype=acctype

self.accbal=accbal

def deposit(self,dep):

self.accbal=self.accbal+dep

print("Amount deposited")

def withdraw(self,wit):

if self.accbal>=wit:

self.accbal=self.accbal-wit

print("Balance: ",self.accbal)

else:

print("Insufficient balance.")

l=0

while(l==0):

ch=int(input("Enter choice\n1.New acc\n2.Withdraw\n3.Deposit\n4.Exit\n"))

if ch==1:

name=input("Enter name:")

tipe=input("Enter Acc type:")

start=int(input("Enter the amount:"))

holders[ptr]=(Bank(bnkno,name,tipe,start))

bnkno+=1

print(“Acc no:”,bnkno)

ptr+=1

if ch==2:

srch1=int(input("Enter Acc: no:"))

srch1=srch1-1001

amt1=int(input("Enter Amount:"))

holders[srch1].withdraw(amt1)

if ch==3:

srch1=int(input("Enter Acc: no:"))

srch1=srch1-1001

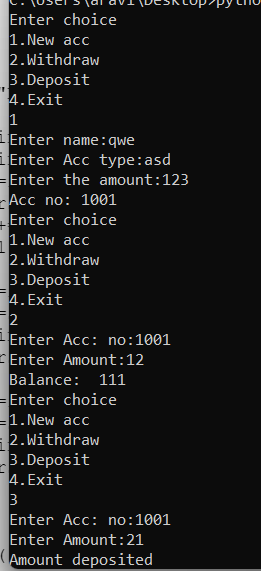
amt1=int(input("Enter Amount:"))

holders[srch1].deposit(amt1)

if ch==4:

l+=1

**OUTPUT**

****

EXPERIMENT 34

**AIM:** Create a class Rectangle with private attributes length and width. Overload ‘<’ operator to compare the area of two rectangles.

**CODE**

class Rectangle:

def \_\_init\_\_(self,l,b):

self.\_\_length=l

self.\_\_width=b

def \_\_lt\_\_(self,ob):

if((self.\_\_length\*self.\_\_width)<(ob.\_\_length\*ob.\_\_width)):

return True

else:

return False

r1=Rectangle(15,12)

r2=Rectangle(34,44)

if(r1<r2):

print("Area of r1<area of r2")

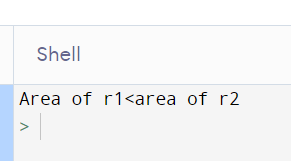
elif(r2<r1):

print("Area of r2<area of r1")

else:

print("Area of r1=area of r2")

**OUTPUT**



EXPERIMENT 35

**Aim :** Create a class Time with private attributes hour,minute and second. Overload ‘+’ operator to find sum of two time.

**CODE**

class Time:

def \_\_init\_\_(self,h,m,s):

self.\_\_hour=h

self.\_\_minute=m

self.\_\_second=s

def \_\_add\_\_(self,ob):

hour=self.\_\_hour+ob.\_\_hour

minute=self.\_\_minute+ob.\_\_minute

second=self.\_\_second+ob.\_\_second

t=Time(hour,minute,second)

return t

def print\_it(self):

print("Hour :",self.\_\_hour)

print("Minute :",self.\_\_minute)

print("Second :",self.\_\_second)

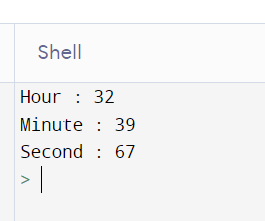
t1=Time(10,10,10)

t2=Time(20,20,20)

t3=t1+t2

t3.print\_it()

**OUTPUT**



EXPERIMENT 36

**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 number\_of\_pages. Write a program that displays information about a Python book. Use base class constructor invocation and method overreading.

**CODE**

class Publisher:

def \_\_init\_\_(self,name):

self.name=name

class Book(Publisher):

def \_\_init\_\_(self,name,title,auther):

super().\_\_init\_\_(name)

self.title=title

self.auther=auther

def print\_function(self):

print("This Fuction is a member fuction of class Publisher")

class Python(Book):

def \_\_init\_\_(self,name,title,auther,price,nop):

super().\_\_init\_\_(name,title,auther)

self.price=price

self.nop=nop

def print\_function(self):

print("Name :",self.name)

print("Title :",self.title)

print("Auther :",self.auther)

print("Price :",self.price)

print("Number of Pages :",self.nop)

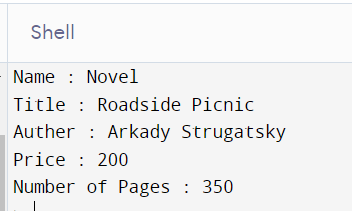
p1=Python("Text book","Python Programming","Mr.abc",100,500)

p1.print\_function()

p2=Book("a","b","c")

p2.print\_function()

**OUTPUT**



**COURSE OUTCOME-5**

EXPERIMENT 37

**Aim :** Write a program to read a file line by line and store it into a list

**CODE**

file=open("text.txt","r")

lines=[]

for line in file:

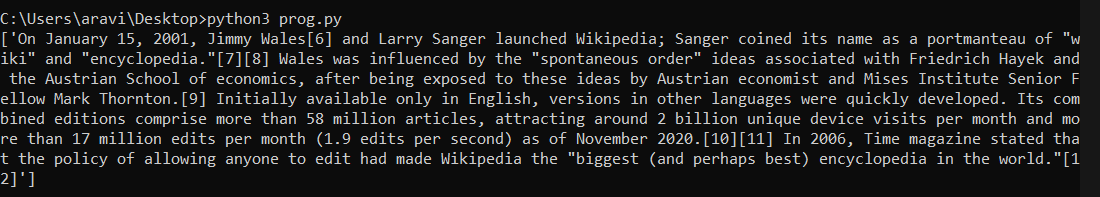
lines.append(line.strip())

print(lines)

**a.txt**

On January 15, 2001, [Jimmy Wales](https://en.wikipedia.org/wiki/Jimmy_Wales)[[6]](https://en.wikipedia.org/wiki/Wikipedia#cite_note-8) and [Larry Sanger](https://en.wikipedia.org/wiki/Larry_Sanger) launched Wikipedia; Sanger coined its name as a [portmanteau](https://en.wikipedia.org/wiki/Portmanteau) of "wiki" and "encyclopedia."[[7]](https://en.wikipedia.org/wiki/Wikipedia#cite_note-MiliardWho-9)[[8]](https://en.wikipedia.org/wiki/Wikipedia#cite_note-J_Sidener-10) Wales was influenced by the "[spontaneous order](https://en.wikipedia.org/wiki/Spontaneous_order)" ideas associated with [Friedrich Hayek](https://en.wikipedia.org/wiki/Friedrich_Hayek) and the [Austrian School](https://en.wikipedia.org/wiki/Austrian_School) of economics, after being exposed to these ideas by Austrian economist and [Mises Institute](https://en.wikipedia.org/wiki/Mises_Institute) Senior Fellow [Mark Thornton](https://en.wikipedia.org/wiki/Mark_Thornton).[[9]](https://en.wikipedia.org/wiki/Wikipedia#cite_note-11) Initially available only in English, versions in other languages were quickly developed. Its combined editions comprise more than 58 million articles, attracting around 2 billion unique device visits per month and more than 17 million edits per month (1.9 edits per second) as of November 2020.[[10]](https://en.wikipedia.org/wiki/Wikipedia#cite_note-small_screen-12)[[11]](https://en.wikipedia.org/wiki/Wikipedia#cite_note-Wikimedia_Stats-13) In 2006, [*Time*](https://en.wikipedia.org/wiki/Time_(magazine)) magazine stated that the policy of allowing anyone to edit had made Wikipedia the "biggest (and perhaps best) encyclopedia in the world."[[12]](https://en.wikipedia.org/wiki/Wikipedia#cite_note-14)

**OUTPUT**



EXPERIMENT 38

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

**CODE**

import csv

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

reader=csv.reader(file)

for row in reader:

print(row)

**a.csv**

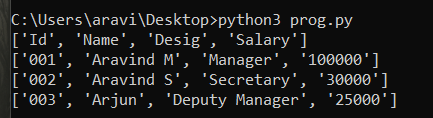
Id,Name,Desig,Salary

001,Aravind M,Manager,100000

002,Aravind S,Secretary,30000

003,Arjun,Deputy Manager,25000

**OUTPUT**

****