CYCLE-1

1. WRITE A PYTHON PROGRAM TO FIND THE AREA OF A CIRCLE

AIM : Area of a circle.

ALGORITHM :

STEP 1: START

STEP 2: import pi from math

STEP 3: Read r, radius

STEP 4: Apply pi*r*r

STEP 5: STOP.

PROGRAM

```
from math import pi
r = float(input ("Input the radius of the circle : "))
print ("The area of the circle with radius " + str(r) + " is: " + str(pi * r**2))
```

RESULT

The output is obtained and the result is verified

2. WRITE A PYTHON PROGRAM TO FIND OUT SIMPLE INTEREST

AIM : Program to find simple interest.

ALGORITHM :

STEP 1: START

STEP 2: Define a function simple_interest(p, t, r)

STEP 2: Read pr, ti, ra (principal, time period, rate of interest)

STEP 3: When the function is called, perform (p*t*r)/100

STEP 4: The result is returned.

STEP 5: STOP.

SOURCE CODE:

```
def simple_interest(p,t,r):
    si = (p * t * r)/100
    print('The Simple Interest is', si)
    return si
pr=float(input("Enter the principle: "))
ti=float(input("Enter the time: "))
ra=float(input("Enter the rate of interest: "))
simple_interest(pr,ti,ra)
```

RESULT :

Output is obtained and the result is verified.

3. WRITE A PROGRAM TO SWAP TWO NUMBERS WITHOUT USING TEMPORARY VARIABLES.

AIM : Program to swap two numbers without using temporary variables.

ALGORITHM :

STEP 1: START

STEP 2: Read x and y

STEP 3: Print the values before swapping

STEP 4: Swap the values – x,y=y,x

STEP 5: Print the values after swapping

STEP 6: STOP.

SOURCE CODE :

```
x = int(input("Enter the first value: "))
y = int(input("Enter the second value: "))
print ("Before swapping: ")
print("Value of x : ", x, " and y : ", y)
x, y = y, x
print ("After swapping: ")
print("Value of x : ", x, " and y : ", y)
```

RESULT :Output is obtained and the result is verified

4. WRITE A PYTHON PROGRAM TO CONVERT TEMPERATURE FROM CELSIUS TO FAHRENHEIT.

AIM : Convert temperature from Degree Celsius to Fahrenheit.

ALGORITHM :

STEP 1: START

STEP 2: Read the values of temperature in Degree Celsius

STEP 3: Apply (d*1.8)+32

STEP 4: Print the converted temperature.

STEP 5: STOP.

SOUCE CODE :

```
d=float(input("Temperature in Degree Celsius: "))
f=(d*1.8)+32
print("Converted temperature is: ",f,"F")
```

RESULT :

Output is obtained and the result is verified.

5. WRITE A PYTHON PROGRAM TO CHECK WHETHER A GIVEN NUMBER IS ODD OR EVEN.

AIM : Program to check whether a given number is odd or even.

ALGORITHM :

STEP 1: START

STEP 2: Read a number

STEP 3: Using if check whether the number is divisible by 2- if(num%2)==0

STEP 4: Print the result

STEP 5: STOP.

SOURCE CODE :

```
num = int(input("Enter a number: "))
if (num % 2) == 0:
   print(str(num) + " is an even number")
else:
   print(str(num) + " is an odd number")
```

RESULT :

Output is obtained and the result is verified.

6. WRITE A PYTHON PROGRAM TO FIND THE LARGEST OF THREE NUMBERS.

AIM : Program to find the largest of three numbers.

ALGORITHM :

STEP 1: START

STEP 2: Read a, b, c

STEP 3: Compare the numbers using if..elif..else

STEP 4: Print the result.

STEP 5: STOP.

SOURCE CODE :

```
a=int(input("Enter the first number: "))
b=int(input("Enter the second number: "))
c=int(input("Enter the third number: "))
if a>b and a>c:
  print(a,"is the largest number")
elif b>c:
  print(b,"is the largest number")
else:
  print(c,"is the largest number")
```

RESULT :

Output is obtained and the result is verified.

7. WRITE A PYTHON PROGRAM TO CHECK WHETHER A GIVEN YEAR IS LEAP YEAR OR NOT.

AIM : Check whether the given year is leap year or not.

ALGORITHM :

STEP 1: Read the year.

STEP 2: Using nested if check if the year is leap year or not.

STEP 3: Print the result.

STEP 4: STOP.

PROGRAM

```
year = int(input("Enter the year: "))
if (year % 4) == 0:
  if (year % 100) == 0:
     if (year % 400) == 0:
        print(str(year) + " is a leap year")
     else:
        print(str(year) + " is not a leap year")
  else:
     print(str(year) + " is a leap year")
else:
  print(str(year) + " is not a leap year")
```

RESULT :

Output is obtained and the result is verified.

CYCLE-2

8. WRITE A PYTHON PROGRAM TO GENERATE A LIST OF POSITIVE NUMBERS FROM A GIVEN LIST OF INTEGERS.

AIM : Program to generate a list of positive numbers from a given list of integers.

ALGORITHM :

STEP 1: Read N, the total number of elements in the list.

STEP 2: Enter the elements and append it in lists

STEP 3: Print the list of elements that are greater than 0

STEP 4: STOP.

SOURCE CODE :

```
N=int(input("Enter Total number of elements in list : "))
lists=[]
for i in range(N):
    value=int(input("Enter a number :"))
    lists.append(value)
test = [each for each in lists if each>0]
print(test)
```

RESULT :

Output is obtained and the result is verified.

9. WRITE A PYTHON PROGRAM TO FORM A LIST OF VOWELS SELECTED FROM A GIVEN WORD USING LIST COMPREHENSION.

AIM : PROGRAM TO FORM A LIST OF VOWELS SELECTED FROM A WORD USING LIST COMPREHENSION.

ALGORITHM :

STEP 1: START

STEP 2: Read a word

STEP 3: Find the vowels from the word and append in list[]

STEP 4: Print the result

STEP 5: STOP

SOURCE CODE :

```
a= input("ENTER THE WORD: ")
list=[]
vowels="AaEeIiOoUu"
ans=set(each for each in a if each in vowels)
list.append(ans)
print(list)
```

RESULT :

Output is obtained and the result is verified.

10. WRITE A PYTHON PROGRAM TO GET A STRING FROM THE GIVEN STRING WHERE ALL OCCURRENCES OF ITS FIRST CHAR HAVE BEEN CHANGED TO '$' EXCEPT THE FIRST CHAR ITSELF.

AIM : PROGRAM TO GET A STRING FROM THE GIVEN STRING WHERE ALL OCCURRENCES OF ITS FIRST CHAR HAVE BEEN CHANGED TO '$' EXCEPT THE FIRST CHAR ITSELF.

ALGORITHM :

STEP 1: START

STEP 2: Define a function change_char(str1)

STEP 3: Perform str1 = str1.replace(char, '$'), str1 = char + str1[1:] and return str1

STEP 4: Print the result

STEP 5: STOP

SOURCE CODE :

```
def change_char(str1):
  char = str1[0]
  str1 = str1.replace(char, '$')
  str1 = char + str1[1:]
  return str1
print(change_char('onion'))
```

RESULT :

Output is obtained and the result is verified.

11. WRITE A PYTHON PROGRAM TO COUNT THE OCCURRENCES OF EACH WORD IN A LINE OF TEXT.

AIM : PROGRAM TO COUNT THE OCCURRENCES OF EACH WORD IN A LINE OF TEXT.

ALGORITHM :

STEP 1: START

STEP 2: Store a sentence in s

STEP 3: Split the sentence and store it in words

STEP 4: Find the result by performing- {i:words.count(i) for i in set(words)}

STEP 5: Print

STEP 6: STOP

SOURCE CODE :

```
s = 'This course introduces a basic step towards program writing and develops the logical ability and problem-solving skill using Python Programming Language'
words = s.split(' ')
result = {i:words.count(i) for i in set(words)}
print(result)
```

RESULT :

Output is obtained and the result is verified.

12. WRITE A PYTHON PROGRAM THAT SORTS DICTIONARIES IN ASCENDING AND DESCENDING ORDER.

AIM : PROGRAM THAT SORTS DICTIONARIES IN ASCENDING AND DESCENDING ORDER.

ALGORITHM :

STEP 1: START

STEP 2: Print d, the original dictionary

STEP 3: Sort the dictionary in ascending order by applying sorted(d) and print the result.

STEP 4: Sort the dictionary in descending order by applying sorted(d,reverse=True) and print the result.

STEP 5: STOP.

SOURCE CODE :

```
d = {1: 2, 3: 4, 4: 3, 2: 1, 0: 0}
print('Original dictionary : ',d)
x = sorted(d)
print("Ascending order:",x)
x=sorted(d,reverse=True)
print("Descending order:",x)
```

RESULT :

Output is obtained and the result is verified.

13. WRITE A PYTHON PROGRAM TO MERGE TWO DICTIONARIES.

AIM : PROGRAM TO MERGE TWO DICTIONARIES.

ALGORITHM :

STEP 1: START

STEP 2: Create two dictionaries dict1 and dict2

STEP 3: Merge two dictionaries by applying {**dict1 , **dict2} and store it in dict3

STEP 4: Print dict3

STEP 5: STOP.

SOURCE CODE :

```
dict1 = {  'Ritika': 5, 'Sam': 7, 'John' : 10 }
dict2 = {'Aadi': 8,'Mark' : 11 }
dict3 = {**dict1 , **dict2}
print('Merged dictionary :')
print(dict3)
```

RESULT :

Output is obtained and the result is verified.

14. WRITE A PYTHON PROGRAM TO FIND GCD OF 2 NUMBERS.

AIM : PROGRAM TO FIND GCD OF 2 NUMBERS.

ALGORITHM :

STEP 1: START
STEP 2: Create two dictionaries dict1 and dict2
STEP 3: Merge two dictionaries by applying {**dict1 , **dict2} and store it in dict3
STEP 4: Print dict3
STEP 5: STOP.

SOURCE CODE :

```
def compute_hcf(x, y):
   if x > y:
      smaller = y
   else:
      smaller = x
   for i in range(1, smaller+1):
      if((x % i == 0) and (y % i == 0)):
         hcf = i
   return hcf

num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))

print("The H.C.F. is", compute_hcf(num1, num2))
```

RESULT :

Output is obtained and the result is verified.

CYCLE-3

15. WRITE A PYTHON PROGRAM TO FIND THE FACTORIAL OF A NUMBER USING FUNCTIONS

AIM : PROGRAM TO FIND THE FACTORIAL OF A NUMBER USING FUNCTIONS.

ALGORITHM :

STEP 1: START

STEP 2: Define a function fact(n)

STEP 3: Read num

STEP 4: When the function is called, the factorial is computed by performing n*fact(n-1) and the value is returned, which is then displayed.

Step 5: STOP.

SOURCE CODE :

```
def fact(n):
  if n == 1:
     return n
  else:
     return n*fact(n-1)
num = int(input("Enter a number: "))
if num < 0:
  print("Sorry, factorial does not exist for negative numbers")
```

```
elif num == 0:
  print("The factorial of 0 is 1")
else:
  print("The factorial of",num,"is", fact(num))
```

RESULT :

Output is obtained and the result is verified.

16. WRITE A PROGRAM TO GENERATE FIBONACCI SERIES OF N TERMS USING FUNCTIONS.

AIM : PROGRAM TO GENERATE FIBONACCI SERIES OF N TERMS USING FUNCTIONS.

ALGORITHM :

STEP 1: START

STEP 2: Define a function fib(n)

STEP 3: Read nterms

STEP 4: When the function fib() is called, the Fibonacci sequence of numbers upto nterms is found by performing-

(fib(n-1) + fib(n-2)) , and the result is returned and displayed.

SOURCE CODE :

```
def fib(n):
  if n <= 1:
     return n
  else:
     return(fib(n-1) + fib(n-2))
nterms = int(input("Enter the limit: "))
if nterms <= 0:
  print("Plese enter a positive integer")
else:
  print("Fibonacci sequence:")
  for i in range(nterms):
     print(fib(i))
```

RESULT :

Output is obtained and the result is verified.

17. WRITE A PYTHON PROGRAM TO DISPLAY THE GIVEN PYRAMID WITH THE STEP NUMBER ACCEPTED FROM USER USING FUNCTIONS.

Eg: N=4

1

2 4

3 6 9

4 8 12 16

AIM : PROGRAM TO DISPLAY THE GIVEN PYRAMID WITH THE STEP NUMBER ACCEPTED FROM USER USING FUNCTIONS.

ALGORITHM :

STEP 1: START

STEP 2: Read N

STEP 3: Using nested for loop, increment the values and perform i * j , and finally print the pattern upto N

STEP 4: STOP.

SOURCE CODE :

```
N = int(input("N = "))
for i in range(1,N+1):
   for j in range(1,i+1):
      print(i * j," ", end="")
   print()
```

RESULT : Output is obtained and the result is verified

18. WRITE A LAMBDA FUNCTIONS TO FIND THE AREA OF SQUARE, RECTANGLE AND TRIANGLE.

AIM : PROGRAM TO WRITE LAMBDA FUNCTIONS TO FIND THE AREA OF SQUARE, RECTANGLE AND TRIANGLE.

ALGORITHM :

STEP 1: START

STEP 2: area_of_a_rectangle is a function that takes two integers and return l*b

STEP 3: area_of_a_square is a function that takes an integer and return a*a

STEP 4: area_ of_a_triangle is a function that takes two integers and return (1/2)*l*b

STEP 5: Returns and prints the result corresponding to the values

STEP 6: STOP.

SOURCE CODE :

```
area_of_a_rectangle = lambda l,b : l*b
area_of_a_square=lambda a: a*a
area_of_a_triangle=lambda l,b: (1/2)*l*b
print("Area of rectangle is:", area_of_a_rectangle(3,4))
print("Area of square is:", area_of_a_square(5))
print("Area of triangle is:", area_of_a_triangle(6,7))
```

RESULT :

Output is obtained and the result is verified.

19. 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)

AIM : Write programs that finds area and perimeter of figures by different importing statements.

ALGORITHM :

STEP 1: START
STEP 2: Create a folder graphics and in it store: rectangle.py, circle.py and init.py
STEP 3: Create another folder in graphics and in it store: cuboid.py, sphere.py and init.py
STEP 4: Import these packages into another program and print the output
STEP 5: STOP

SOURCE CODE :

1. Graphics:
   circle.py

```
from math import pi

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

   rectangle.py

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

   init.py

2. tdgraphics
   cuboid.py

```
def area_cuboid(l,b,h):
    return 2*(l*h + b*h + l*b)
def perimeter_cuboid(l,b,h):
    return 4*(l+b+h)
```

sphere.py

```
from math import pi
def area_sphere(radius):
   return 4*(pi*radius*radius)
def perimeter_sphere(radius):
   return 2*pi*radius
```

init.py

3. graphics module check.py

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

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

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

RESULT :

Output is obtained and the result is verified.

20. 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.

AIM : Write a program to 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.

ALGORITHM :

STEP 1: START
STEP 2: Read input as name, account number, type, balance
STEP 3: Call function as per the entered option
STEP 4: View details as per the predefined function.
STEP 5: STOP.

SOURCE CODE :

```
class bank:
    __acc_name=""
    __acc_no = ""
    __acc_type = ""
    __acc_balance = 0

    def __init__(self,a_name,a_no,a_type,a_balance):
        self.__acc_name = a_name
        self.__acc_no = a_no
        self.__acc_type = a_type
        self.__acc_balance = a_balance

    def deposit(self,a_deposit):
        print("Initial balance is  : ",self.__acc_balance)
        self.__acc_balance += a_deposit
        print("Current balance is  : ",self.__acc_balance)
```

```
    def withdraw(self):
        print("Current balance is  : ",self.__acc_balance)
        self.amount = int(input("Please enter the amount you want to withdraw: "))
        if self.amount > self.__acc_balance:
            print("You don't have enough balance to withdraw ")
            print("Current balance is  : ",self.__acc_balance)
        else:
            print(self.amount," has been withdrawn .")
            self.__acc_balance -= self.amount
            print("Current balance is  : ",self.__acc_balance)

    def acc_info(self):

        print("Account holder name  :  ",self.__acc_name)
        print("Account number       :  ",self.__acc_no)
        print("Account type         :  ",self.__acc_type)
        print("Account Balance is   :  ",self.__acc_balance)



def main():

    name  = input("Enter Account holder name : ")
    no    = input("Enter Account number       : ")
    atype  = input("Enter Account type         : ")
    bal    = int(input("Enter Account initial balance : "))
    holder = bank(name,no,atype,bal)

    while(True):
        print("\n")
```

```
    opt = int(input("1)Deposit \n2)Withdraw \n3)Account info \n0)Exit\nChoose
your option :: "))


    if opt == 1:
        amount = int(input("Deposit amount : "))
        holder.deposit(amount)
    elif opt == 2:
        holder.withdraw()
    elif opt == 3:
        holder.acc_info()
    elif opt == 0:
        break
    else:
        print("Invalid Option !")


if __name__ == "__main__":
  while(True):
    main()
```

RESULT :

Output is obtained and the result is verified.

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

AIM : Write a program to create a class Rectangle with private attributes length and width. Overload '<' operator to compare the area of 2 rectangles.

ALGORITHM :

STEP 1: START

STEP 2: Take input as length and breadth from two rectangles

STEP 3:Calculate area of rectangles and compare them.

STEP 4: Print the result of area and the largest rectangle among them.

SOURCE CODE :

```
class rectangle:
    __area = 0
    __perimeter = 0
    def __init__(self,length,width):
        self.__length = length
        self.__width = width
    def calc_area(self):
        self.__area = self.__length*self.__width
        print("Area is  :",self.__area)
    def __lt__(self,second):
        if self.__area < second.__area:
            return True
        else:
            return False

length1= int(input("Enter length of the rectangle 1 : "))
width1 = int(input("Enter breadth of the rectangle  1 : "))
length2 = int(input("Enter length of the rectangle  2  : "))
width2 = int(input("Enter breadth of the rectangle  2 : "))

obj1 = rectangle(length1,width1)
obj2 = rectangle(length2,width2)
obj1.calc_area()
```

```
obj2.calc_area()


if obj1 < obj2:
      print("Rectangle two is large")
elif obj1>obj2:
      print("Rectangle one is large ")
else:
       print("These are equal")
```

RESULT

The output is obtained and result is obtained