FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)™

HORMIS NAGAR, MOOKKANNOOR

ANGAMALY-683577

'FOCUS ON EXCELLENCE'

20MCA131 PROGRAMMING LAB LABORATORY RECORD

Name: **AKSHAYA K R**

Branch: MASTER OF COMPUTER APPLICATION

Semester:1 **Batch: 2021 A**

Roll No:11 **Reg No: FIT21MCA-2010**

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 ***AKSHAYA K R(FIT21MCA-2010)*** *in the* ***20MCA131 PROGRAMMING LAB*** *Laboratorytowardsthe partial fulfilment for the award of the Master Of Computer Applications during the academic year 2021-2022.*

.

Signature of Staff in Charge

Name:

Date:

Signature of H.O.D

Name:

Date of University practical examination ……

Signature of

Internal Examiner

Signature of

External Examiner

CONTENT

9	**10/11/2021**	Accept the radius from user and find area of circle.		
10	**11/11/2021**	Find biggest of 3 numbers entered.		
11	**11/11/2021**	Accept a file name from user and print extension of that.		
12	**11/11/2021**	Create a list of colors from comma separated color names entered by user. Display first and last colors.		
13	**11/11/2021**	Accept an integer n and compute n+nn+nnn.		
14	**11/11/2021**	Print out all colors from colorlist1 not contained in color-list2.		
15	**17/11/2021**	Create a single string separated with space from two strings by swapping the character at position 1.		
16	**17/11/2021**	Sort dictionary in ascending and descending order		
17	**17/11/2021**	Merge two dictionaries.		
18	**17/11/2021**	Find gcd of 2 numbers.		
19	**17/11/2021**	From a list of integers, create a list removing even numbers.		
		CO2		
20	**25/11/2021**	Program to find the factorial of a number.		
21	**25/11/2021**	Generate Fibonacci series of N terms.		
22	**25/11/2021**	Find the sum of all items in a list		
23	**25/11/2021**	Generate a list of four digit numbers in a given range with all their digits even and the number is a perfect square		

24	**02/12/2021**	Display the given pyramid with step number accepted from user. Eg: N=4 1 2 4 3 6 9 4 8 12 16		
25	**02/12/2021**	Count the number of characters (character frequency) in a string.		
26	**02/12/2021**	Add 'ing' at the end of a given string. If it already ends with 'ing', then add 'ly'		
27	**09/12/2021**	Accept a list of words and return length of longest word.		
28	**09/12/2021**	Construct following pattern using nested loop *		
29	**09/12/2021**	Generate all factors of a number.		
		CO3		
30	**29/01/2022**	Create a package graphics with modules rectangle, circle and subpackage 3D-graphics with modules cuboid and sphere. Include methods to find area andperimeter 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)		

		CO4		
31	**13/01/2022**	Create Rectangle class with attributes length and breadth and methods to find area and perimeter. Compare two Rectangle objects by their area.		
32	**13/01/2022**	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	**13/01/2022**	Create a class Rectangle with private attributes length and width. Overload '<' operator to compare the area of 2 rectangles.		
34	**20/01/2022**	Create a class Time with private attributes hour, minute and second. Overload '+' operator to find sum of 2 time.		
35	**20/01/2022**	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.		
		CO5		
36	**03/02/2022**	Write a program to read a file line by line and store it into a list		
37	**03/02/2022**	Write a Python program to read each row from a given csv file and print a list of strings		

COURSE OUTCOME 1

PROGRAM:1

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

CODE:

```
cy=int(input("current year:"))
year=int(input("any year:"))
for i in range(cy,year):
if(i%4==0):
print(i)
```

OUTPUT

```
stud@debian:~/akshaya11/pythonco1$ python3 2.py
current year:2021
any year:2040
2024
2028
2032
2036
stud@debian:~/akshaya11/pythonco1$
```

PROGRAM:2

AIM: List comprehensions:

q) Generate positive list of numbers from agiven list of integers.

b) Square of N numbers.

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

d) list ordinal value of each of a word to get ordinal values.

CODE:

a)

```
 print('a. Generate positive list of numbers from a given list of
intergers') a=[14,6,-8,-7,6,3]
 print('positive intergers')
for i in a:
 if i>=0:
 print(i)
```

b)

```
print('b. Square of N numbers')
b=[4,3,8]
print('Square of numbers')
for i in b:
 i=i*i
 print(i)
```

c)

```
print('c. Form a list of vowels selected from a given word')
c = input("Enter a word : ")
vowel =['a','e','i','o','u']
l=[]
for i in c:
 if (i in vowel and i not in l):
 l.append(i)
print("Vowels present in given word : ",l)
```

d)

```
wd=input("enter word:")
ls=[]
for i in wd:
        od=ord(i)
        ls.append(od)
print(ls)
```

OUTPUT:

```
stud@debian:~/akshaya11/pythoncol$ python3 3.py
a. Generate positive list of numbers from a given list of intergers
positive intergers
14
6
6
3
b. Square of N numbers
Square of numbers
16
9
64
c. Form a list of vowels selected from a given word
Enter a word : apple
Vowels present in given word :  ['a', 'e']
d)list ordinal value of each element
enter word:akshaya
[97, 107, 115, 104, 97, 121, 97]
stud@debian:~/akshaya11/pythoncol$
```

PROGRAM:3

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

CODE:

```
str=input("enter the string: ")
 counts = dict()
words = str.split()
for i in words:
 if i in counts:
 counts[i] += 1
 else:
 counts[i] = 1
print("count of each word is:")
print(counts)
```

OUTPUT

```
stud@debian:~/akshaya11/pythoncol$ python3 4.py
enter the string: eveything is possible nothing is impossible
count of each word is:
{'eveything': 1, 'is': 2, 'possible': 1, 'nothing': 1, 'impossible': 1}
stud@debian:~/akshaya11/pythoncol$
```

PROGRAM:4

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

CODE:

```
integers=int(input("enter the limit:"))
ls=[]
for i in range(0,integers):
        num=int(input("enter a number:"))
        if(num <= 100):
                ls.append(num)
        else:
                ls.append("over")
print("list elements are",ls)
```

OUTPUT:

```
stud@debian:~/akshaya11/pythoncol$ python3 5a.py
enter the limit:4
enter a number:200
enter a number:100
enter a number:50
enter a number:78
list elements are ['over', 100, 50, 78]
stud@debian:~/akshaya11/pythoncol$
```

PROGRAM:5

AIM: Store alist of names.count the occurences of ‘a’ within the list?

CODE:

```
l=['apple','aasam','lala']
count=0
for i in l:

 num= i.count('a')
 count=count+num
print(l);
print(count)
```

OUTPUT

```
stud@debian:~/akshayall/pythoncol$ python3 6.py
['apple', 'aasan', 'lala']
6
stud@debian:~/akshayall/pythoncol$
```

PROGRAM:6

AIM: Enter 2 list of integers.Check

(a) whether list are of same length

(b)whether list sums to same value

(c)whether any value occur in both

CODE:

```
a)l1=[6,4,7,8,5]
l2=[5,9,3,8,4]
print(l1)
print(l2)
x=len(l1)
y=len(l2)
if(x==y):
print("same length")
else:
print("not same length")

b)s1=0
s2=0
for i in l1:
s1=s1+i
print("sum of first list is:",s1)
for j in l2:
s2=s2+j
print("sum of second list is:",s2)
if(s1==s2):
print("same sum")
else:
print("not same sum")
```

c)
for i in l1:
if i in l2:
print(i,"occur in both list")

OUTPUT:

```
stud@debian:~/akshaya11/pythonco1$ python3 7.py
[6, 4, 7, 8, 5]
[5, 9, 3, 8, 4]
same length
sum of first list is: 30
sum of second list is: 29
not same sum
4 occur in both list
8 occur in both list
5 occur in both list
stud@debian:~/akshaya11/pythonco1$
```

PROGRAM:7

AIM: Get a string from an input string where all occurences of first character replaced with '$' except first character.

CODE:

```
s1=input("enter a string:")
print("original string",s1)
char=s1[0]
s1=s1.replace(char,'$')
s1=char+s1[1:]
print("replaced string:",s1)
```

OUTPUT:

```
stud@debian:~/akshaya11/pythonco1$ python3 8.py
enter a string:onion
original string onion
replaced string: oni$n
stud@debian:~/akshaya11/pythonstud@debian:~/akshaya11/pythonc
stud@debian:~/akshaya11/pythonco1$
```

PROGRAM:8

AIM: Create a string from given string where first and last characters exchanged

CODE:

```
s=input("enter astring:")
print("original string:",s)
sf=s[0]
sl=s[-1]
n=len(s)
ns=sl+s[1:n-1]+sf
print(ns)
```

OUTPUT:

```
stud@debian:~/akshaya11/pythoncol$ python3 9.py
enter astring:python
original string: python
nythop
stud@debian:~/akshaya11/pythoncol$
```

PROGRAM:9

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

CODE:

```
r=int(input('enter the radius: '))
a=3.14*r*r
print("Area of circle: ",a)
```

OUTPUT:

```
stud@debian:~/akshaya11/pythoncol$ python3 10.py
enter the radius: 4
Area of circle:  50.24
stud@debian:~/akshaya11/pythoncol$
```

PROGRAM:10

AIM: Find the biggest of 3 numbers

CODE:

```
print("Enter the three numbers: ")
a=int(input())
b=int(input())
c=int(input())
if a>b and a>c:
        print("The biggest of three numbers: ",a)
if b>a and b>c:
        print("The biggest of three numbers: ",b)
if c>a and c>b:
        print("The biggest of three numbers: ",c)
```

OUTPUT:

```
stud@debian:~/akshaya11/pythonco1$ python3 11.py
Enter the three numbers:
5
8
2
The biggest of three numbers:  8
stud@debian:~/akshaya11/pythonco1$
```

PROGRAM:11

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

CODE:

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

OUTPUT:

```
stud@debian:~/akshaya11/pythonco1$ python3 12.py
enter the file name:12.py
the extension of file 12.py is ('12', '.py')
stud@debian:~/akshaya11/pythonco1$
```

PROGRAM:12

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

CODE:

```
ls=["red","green","blue","yellow"]
print(ls)
print(ls[0])
print(ls[-1])
```

OUTPUT:

```
stud@debian:~/akshaya11/pythonco1$ python3 13.py
['red', 'green', 'blue', 'yellow']
red
yellow
stud@debian:~/akshaya11/pythonco1$
```

PROGRAM:13

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

CODE:

```
a=int(input("enter the number"))
 n=(a+((a*10)+a)+((a*100)+(a*10)+a))
print(n) output
```

OUTPUT:

```
stud@debian:~/akshaya11/pythonco1$ python3 14.py
enter the number2
246
stud@debian:~/akshaya11/pythonco1$
```

PROGRAM:14

AIM: Print out all colors from clor-list not contained in color-list2

CODE:

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

OUTPUT:

```
stud@debian:~/akshaya11/pythonco1$ python3 15.py
['red', 'blue', 'green', 'black']
['yellow', 'orange', 'blue']
['red', 'green', 'black']
stud@debian:~/akshaya11/pythonco1$
```

PROGRAM:15

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

CODE:

```
a=input("enter string 1:")
b=input("enter string 2:")
new_a = b[:1] + a[1:]
new_b = a[:1] + b[1:]
c=new_a+ ' ' + new_b
print(c)
```

OUTPUT:

```
stud@debian:~/akshaya11$ python3 16.py
enter string 1:akshaya
enter string 2:kr
kkshaya ar
stud@debian:~/akshaya11$
```

PROGRAM:16

AIM: Sort dictionary in ascending and descending order

CODE:

```
d1={"a":1,"c":3,"d":2,"b":4}
l=list(d1.items())
print(l)
l.sort()
print("Ascending order is\n",l)
l=list(d1.items())
l.sort(reverse=True)
print("Descending order is\n",l)
```

OUTPUT

```
stud@debian:~/akshaya11/pythonco1$ python3 merge.py
[('a', 1), ('c', 3), ('d', 2), ('b', 4)]
Ascending order is
 [('a', 1), ('b', 4), ('c', 3), ('d', 2)]
Descending order is
 [('d', 2), ('c', 3), ('b', 4), ('a', 1)]
stud@debian:~/akshaya11/pythonco1$
```

PROGRAM:17

AIM: Merge two dictionaries

CODE:

```
thisdict ={ "place":'perumbavoor' ,"age":'21'}
Di={"name":'akshaya' , "rollno":'11'}
Di.update(thisdict)
print(Di)
```

OUTPUT:

```
stud@debian:~/akshaya11/pythonco1$ python3 18.py
{'name': 'akshaya', 'rollno': '11', 'place': 'perumbavoor', 'age': '21'}
stud@debian:~/akshaya11/pythonco1$
```

PROGRAM:18

AIM: Find the gcd of 2 numbers

CODE:

```
a=int(input("enter first number:"))
b=int(input("enter second number:"))
if(a>b):
        x1=a
        x2=b
        for i in range(1,x2+1):
                if((x1%i==0) and (x2%i==0)):
                        gcd=i
        print(gcd)
```

OUTPUT:

```
stud@debian:~/akshaya11/pythonco1$ python3 gcd.py
enter first number:50
enter second number:10
10
stud@debian:~/akshaya11/pythonco1$
```

PROGRAM:19

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

CODE:

```
l1=[4,7,8,9,18,17]
print(l1)
for i in l1:
        if(i%2==0):
                l1.remove(i)
print(l1)
```

OUTPUT:

```
stud@debian:~/akshaya11/pythonco1$ python3 20.py
[4, 7, 8, 9, 18, 17]
[7, 9, 17]
stud@debian:~/akshaya11/pythonstud@debian:~/akshaya11/pythonc
stud@debian:~/akshaya11/pythonco1$
```

COURSE OUTCOME 2

PROGRAM: 20

AIM: Program to find the factorial of a number

CODE:

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

OUTPUT

```
stud@debian:~/akshaya11/CO2$ python3 1.py
enter the number 5
factorial 120
stud@debian:~/akshaya11/CO2$
```

PROGRAM: 21

AIM: Genetate fibonacci series of N terms

CODE:

```
n=int(input('Enter the number: '))
a=0
b=1
c=0
print("Fibonacci Series:")
print(a)
print(b)
for i in range(3,n+1):
        c=a+b
        print(c)
        a=b
        b=c
```

OUTPUT

```
stud@debian:~/akshaya11/CO2$ python3 2.py
Enter the number: 5
Fibonacci Series:
0
1
1
2
3
stud@debian:~/akshaya11/CO2$
```

PROGRAM: 22

AIM: Find the sum of all items in a list

CODE:

```
def sum_of_list(l):
  total = 0
  for i in l:
    total = total + i
  return total

li = [2,4,6,8,10]
print("The sum of my list is", sum_of_list(li))
```

OUTPUT

```
stud@debian:~/akshaya11/CO2$ python3 3.py
The sum of my list is 30
stud@debian:~/akshaya11/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

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

```
stud@debian:~/akshaya11/CO2$ python3 4.py
68
78
80
92
[4624, 6084, 6400, 8464]
stud@debian:~/akshaya11/CO2$
```

PROGRAM: 24

AIM: Display the given pyramid with step number acceted from user
eg:N=4
1

2 4

3 6 9

4 8 12 16

CODE:

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

OUTPUT

```
stud@debian:~/akshaya11/CO2$ python3 5.py
Enter the limit:4
1

2   4

3   6   9

4   8   12   16

stud@debian:~/akshaya11/CO2$
```

PROGRAM: 25

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

CODE:

```
string=input("Enter a string:")
list1=[]
for i in string:
  if i not in list1:
    list1.append(i)
for i in list1:
  count=0
  for j in string:
    if(i==j):
      count=count+1
  print(i,"\t:",count)
```

OUTPUT

```
stud@debian:~/akshaya11/CO2$ python3 6.py
Enter a string:INDIA IS MY COUNTRY
I       : 3
N       : 2
D       : 1
A       : 1
        : 3
S       : 1
M       : 1
Y       : 2
C       : 1
O       : 1
U       : 1
T       : 1
R       : 1
stud@debian:~/akshaya11/CO2$
```

PROGRAM : 26

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

-

CODE:

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

OUTPUT

```
stud@debian:~/akshaya11/CO2$ python3 7.py
Enter a string:JUMB
JUMBing
stud@debian:~/akshaya11/CO2$
```

PROGRAM :27

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

CODE:

```
list1=[]
n=int(input("Enter the range:"))
print("Enter the words:")
for i in range(0,n):
  list1.append(input(""))
longest=list1[0]
for i in range(1,n):
  if(len(list1[i])>len(longest)):
    longest=list1[i]
print("Length of longest word is",len(longest))
```

OUTPUT

```
stud@debian:~/akshaya11/CO2$ python3 8.py
Enter the range:3
Enter the words:
papergrid
successfull
yourself
Length of longest word is 11
stud@debian:~/akshaya11/CO2$
```

PROGRAM : 28

AIM: Construct following pattern using nest loop

CODE:

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

OUTPUT

```
stud@debian:~/akshaya11/CO2$ python3 9.py

*
* *
* * *
* * * *
* * * * *
* * * * * *
* * * * *
* * * *
* * *
* *
*
stud@debian:~/akshaya11/CO2$
```

PROGRAM: 29

AIM: Generate all factors of a number

CODE:

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

OUTPUT

```
stud@debian:~/akshaya11/CO2$ python3 10.py
Enter a number:10
Factors are
1
2
5
10
stud@debian:~/akshaya11/CO2$
```

COURSE OUTCOME 3

PROGRAM:30

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

Terminal Commands

```
stud@debian:~/akshaya11/graphics$ gedit circle.py
stud@debian:~/akshaya11/graphics$ gedit rectangle.py
stud@debian:~/akshaya11/graphics$ mkdir tdgraphics
stud@debian:~/akshaya11/graphics$ cd tdgraphics
stud@debian:~/akshaya11/graphics/tdgraphics$ gedit cuboid
stud@debian:~/akshaya11/graphics/tdgraphics$ gedit sphere
stud@debian:~/akshaya11/graphics/tdgraphics$ cd ..
stud@debian:~/akshaya11/graphics$ cd ..
```

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
```

Program1.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("Permeter 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

```
Area of a circle with radius 12 is :  452.3893421169302
Permeter of a circle with radius 12 is  75.39822368615503


Area of a Rectangle with length and width 12 is :  144
Permeter of a Rectangle with length and width 12 is :  48


Area of a  cuboid with length,width,height 12 is :  864
Volume of a cuboid with length,width,height 12 is :  1728


Area of a spere with radius 12 is :  1809.5573684677208
Permeter of a spere with radius 12 is  75.39822368615503
```

COURSE OUTCOME 4

PROGRAM:31

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

CODE:

```
class Rectangle():
  def __init__(self,l,b):
    self.length=l
    self.breadth=b
  def area(self):
    return self.length*self.breadth
  def peri(self):
    return 2*(self.length+self.breadth)
r1=Rectangle(10,2)
r2=Rectangle(5,9)
x=r1.area()
y=r2.area()
m=r1.peri()
n=r2.peri()
print("rectangle1 area=",x)
print("rectangle2 area=",y)
print("rectangle1 perimeter=",m)
print("rectangle2 perimeter=",n)
if(x<y):
  print("r1 is smaller")
else:
  print("r2 is smaller")
```

OUTPUT:

```
stud@debian:~/akshaya11/Py.co4$ python3 pgm1.py
Enter the length2
Enter the breadth5
Area of rectangle1 10
Area of rectangle2 24
Peremeter of rectangle1 14
Peremeter of rectangle2 20
Area of Rectangle1 is less than Rectangle2
stud@debian:~/akshaya11/Py.co4$
```

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

CODE:

```
class bank:
  def __init__(self,account_no,name,type_of_account,balance):
    self.acno=account_no
    self.name=name
    self.toa=type_of_account
    self.b=balance
  def withdraw(self,x):
    self.b=self.b-x
  def deposit(self,y):
    self.b=self.b+y
  def print(self):
    print("Account number=",self.acno,"Name=",self.name,"Type of account=",self.toa
,"Balance=",self.b)
acc1=bank(123,"akshaya","fixed",4000)
acc2=bank(456,"archana","sb",10000)
acc1.withdraw(2000)
acc2.deposit(30000)
acc1.print()
acc2.print()
```

OUTPUT

```
stud@debian:~/akshaya11/Py.co4$ python3 pgm2.py
Account number= 123 Name= akshaya Type of account= fixed Balance= 2000
Account number= 456 Name= archana Type of account= sb Balance= 40000
stud@debian:~/akshaya11/Py.co4$
```

PROGRAM:33

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

CODE:

```
class Rectangle:
  def __init__(self,l,b):
    self.__length=l
    self.__breadth=b
  def area(self):
    return self.__length*self.__breadth
  def __lt__(self,m):
    if((self.__length*self.__breadth)<(m.__length*m.__breadth)):
      return True
    else:
      return False
r1=Rectangle(8,2)
r2=Rectangle(4,3)

x=r1.area()
y=r2.area()

print("rectangle1 area=",x)
print("rectangle2 area=",y)
if(r1<r2):
  print("area of rectangle1 smaller")
else:
  print("area of rectangle2 is smaller")
```

OUTPUT

```
stud@debian:~/akshaya11/Py.co4$ python3 pgm3.py
rectangle1 area= 16
rectangle2 area= 12
area of rectangle2 is smaller
stud@debian:~/akshaya11/Py.co4$
```

PROGRAM:34

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

CODE:

```
class Time:
  def __init__(self,hr,min,sec):
    self.__hr=hr
    self.__min=min
    self.__sec=sec

  def __add__(t1,t2):
    hr=t1.__hr+t2.__hr
    min=t1.__min+t2.__min
    sec=t1.__sec+t2.__sec
    print(hr,":",min,":",sec)

t1=Time(2,36,10)
t2=Time(6,20,37)
t1+t2
```

OUTPUT

```
stud@debian:~/akshaya11/Py.co4$ python3 pgm4.py
8 : 56 : 47
stud@debian:~/akshaya11/Py.co4$
```

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

CODE:

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

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

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


p=Python("ABC Publications","Taming Python","jeeva jose",9000,600)
p.display3()
```

OUTPUT

```
stud@debian:~/akshaya11/Py.co4$ python3 pgm5.py
Taming Python
jeeva jose
Taming Python
jeeva jose
9000
600
stud@debian:~/akshaya11/Py.co4$
```

COURSE OUTCOME 5

PROGRAM:36

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)
```

text.txt

Python is a high-level general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small- and large-scale projects.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library

OUTPUT

```
stud@debian:~/akshaya11/c05$ python3 p1.py
['Python is a high-level general-purpose programming language. Its design philos
ophy emphasizes code readability with the use of significant indentation. Its la
nguage constructs and object-oriented approach aim to help programmers write cle
ar, logical code for small- and large-scale projects. Python is dynamically-type
d and garbage-collected. It supports multiple programming paradigms, including s
tructured (particularly, procedural), object-oriented and functional programming
. It is often described as a "batteries included" language due to its comprehens
ive standard library']
stud@debian:~/akshaya11/c05$
```

PROGRAM:37

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)
```

text.csv

	A	B
1	Region	Country
2	Australia and Oceania	Tuvalu
3	Central America and the Caribbean	Grenada
4	Europe	Russia
5	Sub-Saharan Africa	Sao Tome and Principe
6	Sub-Saharan Africa	Rwanda
7	Australia and Oceania	Solomon Islands
8	Sub-Saharan Africa	Angola
9	Sub-Saharan Africa	Burkina Faso
10	Sub-Saharan Africa	Republic of the Congo
11		
12		

OUTPUT

```
stud@debian:~/akshaya11/c05$ python3 p2.py
['Region', 'Country']
['Australia and Oceania', 'Tuvalu']
['Central America and the Caribbean', 'Grenada']
['Europe', 'Russia']
['Sub-Saharan Africa', 'Sao Tome and Principe']
['Sub-Saharan Africa', 'Rwanda']
['Australia and Oceania', 'Solomon Islands']
['Sub-Saharan Africa', 'Angola']
['Sub-Saharan Africa', 'Burkina Faso']
['Sub-Saharan Africa', 'Republic of the Congo']
stud@debian:~/akshaya11/c05$
```