PROGRAM-1

AIM: Find the largest among three numbers

ALGORITHM :

1.Read numbers n1,n2,n3

2.If n1>n2 and n1>n3, Largest is n1 Else if n2>n1 and n2>n3, Largest is n2

Else, Largest is n3

End of if

3.Print largest

SOURCE CODE :

```
n1=int(input('Enter first Number:'))
n2=int(input('Enter second Number:'))
n3=int(input('Enter third Number:'))
 if(n1>=n2) and (n1>=n2):
    largest=n1
 elif(n2>=n1) and (n2>=n3):
    largest=n2
 else:
    largest=n3
print('largest number is:',largest)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
Enter first Number:23
Enter second Number:43
Enter third Number:53
largest number is: 53
```

PROGRAM-2

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

ALGORITHM:

1. Read a file name with extension
2. Using split('.')function extract the extension from the file
3. Print extension

SOURCE CODE:

```
fname=input('enter the file name:')
exten=fname.split('.')
 print('extension is:',exten[-1])
```

RESULT:

Program is executed an output is obtained

OUTPUT:

```
enter the file name:program.html
extension is: html
```

PROGRAM-3

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

ALGORITHM:

1. Read string str.
2. Print rearranged string sliced using index.

   str= l + str[1:-1] + f
3. Print str

SOURCE CODE:

```
str=input('Enter a string:')
f=str[0] l=str[-1]
 str=l+str[1:-1]+f
print('New string is:',str)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
Enter a string:program
New string is: mrograp
```

PROGRAM-4

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

ALGORITHM:

1. read a list of colors in list.
2. Print first color, list[0]
3. Print last color list[-1]

SOURCE CODE:

```
list=[] n=int(input('enter no element:"))
for i in range(0,n):
    element=input('enter the color:')
    list.append(element)
print('full list:',list) print('first
element:',list[0]) print('last
element:',list[1])
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
enter the no of element:5
enter the color:red
enter the color:green
enter the color:blue
enter the color:yellow
enter the color:black
full list: ['red', 'green', 'blue', 'yellow', 'black']
first element: red
last element: black
```

PROGRAM-5

AIM: Generate positive list of numbers from a given list of integers

ALGORITHM:

1.Read a list of numbers from user
2.For each in list Check list[i]>0
3.If yes display positive number
4.End of for

SOURCE CODE:

```
list=[]
n=int(input('Enter the no of elements:'))
print('Elements are:')
for i in range (0,n):
    element=int(input())
    list.append(element)
print('Positive Numbers are:')
 for i in range(0,n):
    if(list[i]>=0):
    print(list[i])
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
Enter the no of elements:6
Elements are:
-12
43
22
0
-4
-7
Positive Numbers are:
43
22
0
```

PROGRAM-6

AIM: Python program to find the square of N numbers.

ALGORITHM:

1.Read numbers and append to the list

2.Square=list[i]**2 for each number in the list

3. Display the squared number

SOURCE CODE:

```
list=[]
n=int(input('Enter the no of elements:'))
print('Elements are:')
for i in range (0,n):
    element=int(input())
    list.append(element)
print('sqaure of elements:')
for i in range(0,n):
        print(list[i]**2)
```

RESULT:

The program is executed and output is obtained

OUTPUT:

```
Enter the no of elements:4
Elements are:
12
45
34
23
sqaure of elements:
144
2025
1156
529
```

PROGRAM-7

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

ALGORITHM:

1. Read a string str
2. Enter the element to be searched
3. Initialize count=0
4. Check each elements of string with search if it presen increment count
5. Display count

SOURCE CODE:

```
str=input('Enter the String:')
search=input('Enter the element to be searched:') x=str.split(" ")
count=0
for i in str:
    if(i==search):
        count=count+1
print(count)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
Enter the String:programming
Enter the element to be searched:g
2
```

PROGRAM-8

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

ALGORITHM:

1. Read list of numbers in values.
2. For each i in values do
3. If a<=100 then Append i to x Else:

   Append "over" to x
4. Print x

SOURCE CODE:

```
list=[]
n=int(input('Enter the no of elements:'))
print('Numbers are : \t')
for i in range(0,n):
     number=int(input())
    list.append(number) print('\n',
list)
for i in range(0, n):
    if(list[i]>100):
      list[i]='over'
print(list)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
Enter the no of elements:6
Numbers are :
23
103
44
2
106
23

 [23, 103, 44, 2, 106, 23]
[23, 'over', 44, 2, 'over', 23]
```

PROGRAM-9

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

ALGORITHM:

1. .Read a list of names in a list
2. Initialize list = []
3. Initialize count=0
4. For name in list

   Count=count+name.count(‘a’)
5. print count

SOURCE CODE:

```
list=[]
n=int(input('Enter the Limit:'))
for i in range(0,n):
    fname=input('Enter the first name:')
    list.append(fname)
print('Full List',list) count=0
for name in list:
   count=count+name.count('a')
print('occurance of a:',count)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
Enter the Limit:3
Enter the first name:amal
Enter the first name:samson
Enter the first name:arnold
Full List ['amal', 'samson', 'arnold']
occurance of a: 4
```

PROGRAM-10

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

ALGORITHM:

1. read a string
2. assign $ to first element

   '$'+str[1:]
3. print string

SOURCE CODE:

```
str=input('Enter a string:')
str='$'+str[1:] print('String is :',str)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
Enter a string:program
String is : $rogram
```

PROGRAM – 11

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

ALGORITHM:

1.Read a list of integers in list.

2.Initialize a list, oddlist=[] 3. For each i in list do If list[i] mod 2 != 0 then

3.Append list[i] to oddlist

4. Print new list without even numbers

5.Print oddlist

SOURCE CODE:

```
list=[]
n=int(input("Enter the no of elements:"))
print("numbers are:")
for i in range(0,n):
  number=int(input())
  list.append(number)
  oddlist=[]
for i in range(o,n):
   if((list[i]%2)!=0):
        oddlist.append(list[i])
print(oddlist)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
Enter the no of elements:5
numbers are:
11
12
13
14
15
[11, 13, 15]
```

PROGRAM-12

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

ALGORITHM:

1. Read two list of numbers list1 and list2.
2. //check if the lengths of the two lists are equal.
   If length(list1)=length(list2) then
   Print the lists are same length Else:
   Print the lists are not same length
3. Print sum of elements in both list1 and list2 using sum() method.
4. //check sum of the two lists are same. If sum(list1)=sum(list2) then Print the sum of the lists are same Else:
   Print the sum of lists are not same
5. Initialize list common=[]
6. For each value in list1 do If value in list2 then
   Append value to common
7. //check if the common list is empty. If common then Print common value in both lists Else:
   Print no common values in both lists

SOURCE CODE:

```
list_1=input("Enter the first list of integers(seperated by comma):")
list1=[int(x) for x in list_1.split(',')]
list_2=input("Enter the second list of integers(seperated by comma):")
 list2=[int(x) for x in list_2.split(',')]
if len(list1)==len(list2):
print("The lists are same length",len(list1))
else:
```

```
        print("The list are not same length")
print("The sum of elements of list1:",sum(list1))
print("The sum of elements of list2:",sum(list2))
 if sum(list1)==sum(list2):
    print("The sums of list are same")
else:
   print("The sums of lists are not same")
common=[]
for value in list1:
     if value in list2:
           common.append(value)
     if common:
        print("The common value in both  list:",common)
 else:
       print("No common value found in both list")
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
Enter the first list of integers(seperated by comma):2,3,4
Enter the second list of integers(seperated by comma):3,4,5
The lists are same length 3
The sum of elements of list1: 9
The sum of elements of list2: 12
The sums of lists are not same
The common value in both list: [3, 4]
```

PROGRAM-13

AIM: Write a python program to read two lists color-list1 and color-list2. Print out all colors from color-list1 not contained in color-list2.

ALGORITHM:

1.Read two list of colors in color_list1 and color_list2

2.Initialize list, list1=[]

3.//Append values from color_list1 to list1

For each item in color_list1 do

Append item to list1

Initialize list,list2=[]

1. //Append values from color_list2 to list2
2. For each item in color_list2 do

   Append item to list2
3. Initialize list,result=[]
4. For each item in list1 do If item not in list2 then

   Append item to result
5. Print result

SOURCE CODE:

```
list1=[]
list2=[]
n=int(input("enter the no.of colours in list 1:"))
print("colours are:")
for i in range(0,n):
    color=input()
    list1.append(color)
n=int(input("enter the no.of colours in list 2:"))
print("colours are:")
for i in range(0,n):
     color=input()
list2.append(color)
```

```
set1=set(list1)
set2=set(list2)
print("Orginal set are:\n",list1,"\n",list2)
print("Colour difference:",set1-set2)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
enter the no of colours in list 1:4
colours are:
red
green
yellow
blue
enter the no of colours in list 2:3
colours are:
black
green
purple
Orginal set elements are:
 ['red', 'green', 'yellow', 'blue']
 ['black', 'green', 'purple']
Colour difference: {'blue', 'yellow', 'red'}
```

PROGRAM-14

AIM: Find GCD of 2 numbers.

ALGORITHM:

1. Read two numbers
2. Initialize gcd=1 and n=minimum(num1,num2)+1
3. For I in range 1,n

   If num1%2==0 and num2%2==0

   Then gcd=i
4. Print gcd

SOURCE CODE:

```
num1=int(input('enter number1:'))
num2=int(input('enter number2:'))
gcd=1
n=min(num1,num2)+1
for i in range(1,n):
   if((num1%i==0) and (num2%i==0))
      gcd=i
print("GCD",num1,num2,'is',gcd)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
enter number1:45
enter number2:1675
GCD 45 1675 is 5
```

PROGRAM-15

AIM: Generate all factors of a number.

ALGORITHM:

1. Read a Number n
2. If n%2==0 then print element

   Iterate upto i<=n

SOURCE CODE:

```
n=int(input("enter a number:"))
print("factors of",n,"are:")
for i in range(1,n):
    if(n%i==0):
        print(i)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
enter a number:34
factors of 34 are:
1
2
17
```

PROGRAM – 16

AIM: Find the sum of all items in a list

ALGORITHM:

1.Read the number of elements in the list, n.
2.Initialize a list, list=[] 3. For i=0 to n-1:

3.Get an element from the user

4.Append element to list

5.//Find the sum of the list

sum= sum(list)

6.Print sum

SOURCE CODE:

```
list=[]
n=int(input("enter the limit:"))
for i in range(0,n):
  num=int(input("enter the number:"))
  list.append(num)
sum=0
for i in range(0,n):
   sum=sum+list[i]
print("sum is",sum)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
enter the limit:5
enter the number:23
enter the number:45
enter the number:6
enter the number:98
enter the number:55
sum is 227
```

PROGRAM-17

AIM: Generate Fibonacci series of N terms

ALGORITHM:

1.Read the limit of the series, n.

2.Initialize a:= 0 and b:= 1

3.For i=0 to n-1:

Print a

Store thevalue of ‘a’ in a temporary variable temp = a

Update ‘a’ with the value of ‘b’ a =b

Update ‘b’ with the sum of temp and ‘b’ b= temp+b

SOURCE CODE:

```
n=int(input("enter the limit:"))
a=0
b=1
print("fibonacci series: \n",a,"\n",b)
for i in range(1,n-1):
    c=a+b
    a=b
    b=c
print(c)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
enter the limit:10
fibonacci series:
 0
 1
1
2
3
5
8
13
21
34
```

PROGRAM-18

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

ALGORITHM:

1. Read a string str
2. Check the last 3 letter are ing if yes

   Str=str+”ly”

   Else str=str+”ing”
3. Print str

SOURCE CODE:

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

RESULT

Program is executed and output is obtained

OUTPUT

```
enter a string:work
working
```

```
enter a string:working
workingly
```

PROGRAM – 19

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

ALGORITHM:

1.Read a list of words in list.

2.Initialize longest_length:=0

3.For each word in list do

4.If length(word) > longest_length then

Longest_length= length(word) //store the word as the longest word

long_word= word

5.Print longest_word

6.Print longest_length

SOURCE CODE:

```
list=[]
n=int(input("eenter the limit:"))
for i in range(0,n):
    el=input("enter the value:")
    list.append(el)
print(list)
long=len(list[0])
str=list[0]
for i in range(1,n):
     if(len(list[i])>long):
          str=list[i]
          long=len(str)
print("longest word:",str)
print("length:",long)
```

RESULT:

Program is executed and output is obtained.

OUTPUT:

```
eenter the limit:3
enter the value:python
enter the value:data structure
enter the value:web development
['python', 'data structure', 'web development']
longest word: web development
length: 15
```

PROGRAM-20

AIM: Sort dictionary in ascending and descending order

ALGORITHM:

1.Read the number of key-value pairs in the dictionary, n.
2.Initialize a dictionary, dict={}
3.For i=0 to n-1 do

Get the key from the user

Get the value from the user

Assign dict[key]= value

4.Print the dictionary
5.//Sort the dictionary by keys in ascending orde dict_asc= sort dict.items() by

item[0]

6.Print dict_asc
7.//Sort the dictionary by keys in descending order

Dict_ desc= srt dict.items() by item[0] in reverse

8.Print dict_desc

SOURCE CODE:

```
n=int(input("Enter the number of key value pairs in the dictionary:"))
dict={}
for i in range(n):
   key=input("Enter the key:")
   value=input("Enter the value:")
   dict[key]=value
print("The dictonary is:",dict) dict_asc=sorted(dict.items(),key=lambda
item:item[0])
print("sorted dictinary in ascending order:",dict_asc)
dict_desc=sorted(dict.items(),key=lambda item:item[0],reverse=True)
print("sorted dictinary in descending order:",dict_desc)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
Enter the number of key value pairs in the dictionary:3
Enter the key:orange
Enter the value:3
Enter the key:apple
Enter the value:1
Enter the key:banana
Enter the value:2
The dictonary is: {'orange': '3', 'apple': '1', 'banana': '2'}
sorted dictinary in ascending order: [('apple', '1'), ('banana', '2'), ('orange', '3')]
sorted dictinary in descending order: [('orange', '3'), ('banana', '2'), ('apple', '1')]
```

PROGRAM-21

AIM: Merge two dictionaries.

ALGORITHM:

1. Read the number of key-value pairs in first dictionary, n1.
2. Initialize a dictionary, dict1={}
3. For i=0 to n-1 do

   Get the key from the user

   Get the value from the user

   Assign dict1[key]= value
4. Read the number of key-value pairs in second dictionary, n2. 5. Initialize a dictionary, dict2={}
5. For i=0 to n-1 do

   Get the key from the user

   Get the value from the user

   Assign dict2[key]= value
6. Print dict1
7. Print dict2
8. //Create a copy of the first dictionary

   merge_dict= dict1.copy()
9. //Update the copy with the key value pairs from the second dictionary

   merge_dict.update(dict2)
10. Print “merged dictionary”, merge_dic

SOURCE CODE:

```
n1=int(input("ENter the number of key value pairs in first dictionary:"))
dict1={}
for i in range(n1):
   key=input("Enter key:")
   value=input("Enter value:")
   dict1[key]=value
n2=int(input("Enter the number of key value paies in second dictionary:"))
dict2={}
```

```
for i in range(n2):
    key=input("Enter key:")
    value=input("Enter value:")
    dict2[key]=value print(dict1) print(dict2)
merge_dict=dict1.copy()
merge_dict.update(dict2) print("Merged
dictionary:",merge_dict)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
ENter the nuber of key value pairs in first dictionary:3
Enter key:a
Enter value:1
Enter key:b
Enter value:2
Enter key:c
Enter value:3
Enter the number of key value paies in second dictionary:3
Enter key:d
Enter value:4
Enter key:e
Enter value:5
Enter key:f
Enter value:6
{'a': '1', 'b': '2', 'c': '3'}
{'d': '4', 'e': '5', 'f': '6'}
Merged dictionary: {'a': '1', 'b': '2', 'c': '3', 'd': '4', 'e': '5', 'f': '6'}
```

PROGRAM-22

AIM: Program to find the factorial of a number using function

ALGORITHM:

1.Read a number

2.Call Factorial function factorial(n)

3.Define factorial

If n==0 return 1

If n==1 return 1

Initialize fact=1

For i=1 to n+1 do fact=fact*i

4.Print fact

SOURCE CODE:

```
def factorial(n):
    if(n==0):
        return 1
   if(n==1):
        return 1
fact=1
for i in range(1,n+1):
   fact=fact*i
   return fact
num=(int(input("enter a number :")))
fact=factorial(num)
print("factorial is ",fact)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
enter a number :5
factorial is  120
```

PROGRAM – 23

AIM: Generate all factors of a number using function

ALGORITHM:

1. Read a number to num
2. Call fuction factors(n)
   Def factors(n)
   For I =1 to n+1 check n%i==0 if true print i

SOURCE CODE:

```
def factors(n):
for i in range(1,n+1):
if(n%i==0):
    print(i)
num=int(input("enter a number:"))
print("factors are:")
factors(num)
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
enter a number:104
factors are:
1
2
4
8
13
26
52
104
```

PROGRAM – 24

AIM: Write lambda functions to find the area of square, rectangle and triangle

ALGORITHM:

1.Read length of square, a

2.Call lambda function area1(a) Print area1(a)

3.Read length (l) and width(w) of rectangle .

4.call lambda area2(l,w) Print area2(l,w)

5.Read height (h) and base (b) of triangle

6.lambda area3(b,h) Print area3(b,h) lambda functions

7.Define lambda function area1(x)

Return x*x

8.Define lambda function area2(x,y)

Return x*y

9.Define lambda function area3(x,y)

Return 0.5*x*

SOURCE CODE:

```
area1=lambda x:x*x
area2=lambda x,y:x*y
area3=lambda x,y:0.5*x*y

a=int(input("Enter the length of the square:"))
print("Area of the square is",area1(a))
l=int(input("Enter the length of the rectangle:"))
w=int(input("Enter the width of the rectangle:"))
print("Area of the rectangle is",area2(l,w))
h=int(input("Enter the height of the triangle:"))
b=int(input("Enter the base of the triangle:"))
print("Area of the triangle is",area3(h,b))
```

RESULT:

Program is executed an output is obtained

OUTPUT:

```
Enter the length of the square:5
Area of the square is 25
Enter the length of the rectangle:6
Enter the width of the rectangle:4
Area of the rectangle is 24
Enter the height of the triangle:5
Enter the base of the triangle:7
Area of the triangle is 17.5
```

PROGRAM – 25

AIM: Create class person with attribute name and age display the name and age of a particular person

ALGORITHM:

1. Define a class person
2. Define a constructor that takes name and age as parameters
3. Define display function to print name and age of a person
4. Create an object of class person
5. call display function using object.display()

SOURCE CODE:

```
class person:
def __init__(self,name,age):
    self.name=name
    self.age=age     def
display(self):
    print("name is:",self.name)
    print("age is:",self.age)
p1=person("sanju",23)
p1.display()
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
name is: sanju
age is: 23
```

PROGRAM-26

AIM: Create a class Publisher with attributes publisher id and 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 overridin

ALGORITHM:

1.Define a class publlisher

2.Define a constructor that takes publisher_id and publisher_name as parameters.

3.Define a class ,book that inherits from publisher.

4.Define a constructor that takes publisher_id,publisher_name,title and author as Parameters.

5.Define a class python, that inherits from book.

6.Define a constructor that takes publisher_id,publisher_name,title and author, Price and no_of_pages a parameters.

7.Define a function display_info with parameter self.

8.Read publisher id,publisher name,book title,author name,book price and number of pages.

9.Create a object of python class.

10.Call the function display_info and display book information

SOURCE CODE:

```
class Book(Publisher):
   def
__init__(self,name,title,author):
       super().__init__(name)
       self.title=title
       self.author=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  display(self):
print("publishername:",self.name,"\n
","book title:",self.title,"\n","book
author:",self.author,"\n","book
price",self.price,"\n","no_of_pages:"
,self.no_of_pages)
a=Python("Thomas","python
fundamentals","mark",100,200)
a.display()
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
publishername: Thomas
 book title: python fundamentals
 book author: mark
 book price 100
 no_of_pages: 200
```

PROGRAM-27

AIM: Define a class to represent a bank account. Include the following details like name of the depositor, account number, type of account, balance amount in the account. Write methods to assign initial values, to deposit an amount , withdraw an amount after checking the balance, to display details such as name, account number, account type and balance

ALGORITM

1.Define a class bank.

2.Define a function create with parameter self.

3.Define a function deposit with parameter self.

4.Define a function withdraw with parameter self.

5.Create a empty list, bank=[]

6. Print menu options.

1. Ask the user to enter a choice :

If choice is 1 then

Create an object of bank class. Call the create function. Append account to bank list. Else if choice 2 then

Set a flag to 1

Ask the user to enter an account number to search. For account in bank do

If account no = = user input then

Set flag to 1

Break the loop. If flag = 1 then

Call the deposit function

Else:

Print “no such account” Else if choice is 3 then

Set a flag to 0

Ask the user to enter an account number

For account in bank do

If account no = = user input then

Set flag to 1

Break the loop

If flag =1 then

Call the withdraw function

Else:

Print “no such account” Else if choice is 4 then

Set a flag to 0

Ask the user to enter an account number

For account in bank do

If account no = = user input then

Set flag to 1

Break the loop

If flag = 1 then

Print your

balance Else:

Print “no such account” Else:

Print “invalid choice”

Create(self)

1. Read account no as input.
2. Read account holder name.
3. While True do

Read the FD amount

If FD<500 then

Print “deposit more then 500” Else:

Print “Account created”

deposit(self)

1. While True do

Read the deposit amount

If deposit amount is not a multiple of 100 then

Print “enter nultiples of 100” Else:

Print “Amount deposited”

withdraw(self)

1. While True do

Read withdrawal amount

If withdrawal amount > FD amount then

Print “entered amount exceeds balance” Else If withdrwal amount is not a multiple of
100 then

Print “can only withdraw amount which is multiple of 100” Else:

FD - withdrawal amount

Print “Amount withdraw

SOURCE CODE:

```
ClassBank:
    def create(self):
        self.no = int(input("Enter account number: "))
        self.name=input("Enter account holder name: ")
while True:
     self.fd=int(input("Enter FD amount: "))
     if self.fd<500:
         print("Please deposit more than 500")
    else:
          self.fd=self.fd
          print("Account created!")
          break
def deposit(self):
   while True:
      namount = int(input("Enter amount to be deposited: ")
      if namount%100==0:
          self.fd = namount
          print("Amount deposited!")
          break
     else:
          print("Enter multiples of 100!")
def withdraw(self):
    while True:
     w_amount=int(input("Enter money to be withdrawn:")
     if w_amount>self.fd:
        print("Entered amount exceeds balance")
elif w_amount%100!=0:
   print("Can only withdraw amount which is multiple of 100!")
elif self.fd-w_amount<500:
    print("can only withdraw money with minimum balance 500")
else:
```

```
    self.fd=self.fd
    w_amountprint("Amount withdrawn!")
    break
bank = []
while True:
    print("1. A/C creation", "2. Money deposit", "3. Withdraw money", "4.Checkbalance")
    choice =int(input("Enter choice: "))
    if choice == 1:
        account = Bank()
        account.create()
        bank.append(account)
   elif choice == 2:
      flag = 0
      sacno = int(input("Enter account number to search:"))
      for account in bank:
      if account.no == sacno:
          flag=1
          break
         if flag==1
print("Welcomeaccoun
t.name)
account.deposit()
else:
        print("No such account!")     elif
choice == 3:
       flag = 0
       sacno = int(input("Enter account number to search: "))         for
account in bank:
    if account.no == sacno:
          flag = 1
          break
          if flag == 1:
            print("Welcome", account.name)
account.withdraw()
else:
    print("No such account!")
```

```
elif choice == 4:
        flag = 0
       sacno = int(input("Enter account number to search: "))
for account in bank:
     if account.no == sacno:
          print("Welcome", account.name)
          print("Balance: ", account.fd)
elif choice == 5:
        break
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
1. A/C creation 2. Money deposit 3. Withdraw money 4. Checkbalance
Enter choice: 1
Enter account number: 12345
Enter account holder name: abcd
Enter FD amount: 345000
Account created!
1. A/C creation 2. Money deposit 3. Withdraw money 4. Checkbalance
Enter choice: 4
Enter account number to search: 12345
Welcome abcd
Balance:  345000
1. A/C creation 2. Money deposit 3. Withdraw money 4. Checkbalance
Enter choice: 2
Enter account number to search: 12345
Welcome abcd
Enter amount to be deposited: 25000
Amount deposited!
```

PROGRAM-28

AIM: Write a python program to read each row from a given CSV file and print list of string

ALGORITM

1. import csv file
2. open csv file in write mode
3. write contents into csv file using writerow() function
4. close csv file
5. open csv file in read mode
6. print each row

SOURCE CODE:

```
import csv csvfile=open('example.csv','w',newline='')
writer=csv.writer(csvfile)
writer.writerow(['name','age','occupation'])
writer.writerow(['sanju','30','cricketer'])
writer.writerow(['ronaldo','37','footballer']) csvfile.close()
csvfile=open('example.csv','r)
reader=csv.reader(csvfile) for row
in reader:
print(row)
csvfile.close()
```

RESULT:

Program is executed and output is obtained

OUTPUT:

```
['name', 'age', 'occupation']
['sanju', '30', 'cricketer']
['ronaldo', '37', 'footballer']
```