**LIST OF EXPERIMENTS**

1. Write a program to display the largest number among three numbers.

2. Write a program to check the prime number and to display the twin prime numbers.

3. Write a program to display the Fibonacci series and multiplication table by using looping constructs.

4. Write a program for converting decimal to octal, hexadecimals and vice versa by using functions.

5. Write a function to compute the GCD of two numbers.

6. Write a function to perform sorting list of numbers.

7. With the help of string array or list, display a simple calendar in python program without using the calendar module.

8. Demonstrate class and inheritance in python.

9. Create a text file using python file I/O. Read the content of the file and change them from lower to upper case characters. Write the updated content in another file and display it.

10. Write a program to demonstrate the user-defined exception handling mechanism in Python.

11. Design and implement a graphical user interface to perform any arithmetic operation.

12. Write a python program to insert and retrieve data using MySQL.

**AIM:**

Experiment No: 1 Page No: 1

**Program to display the largest number among three numbers**

To write a program to display the largest among three numbers.

**ALGORITHM:**

Step 1: Start the process.

Step 2: Declare the necessary variable (ex. a,b,c).

Step 3: Get the values from the user using input() function.

Step 4: Compare the values using if and display the largest value .

Step 5: Stop the process.

**SOURCE CODE:**

a=int(input(“Enter value 1 : ”)

b=int(input(“Enter value 2 : ”)

c=int(input(“Enter value 3 : ”)

if (a>=b)and(a>=c):

 print(a,“ is largest among three values”)

elif(b>=a)and(b>=c):

 print(b,” is largest among three values”)

else:

 print(c,” is largest among three values”)

**RESULT:**

The above code got executed successfully.

Page no: 2

**OUTPUT:**

**\*Note:** Twin primes are pairs of primes which differ by two i.e, (p+2).

Experiment No: 2 Page No: 3

**Program to check the prime number and to display the twin prime numbers.**

**AIM:**

To write a program to check the prime number and to display its twin prime.

**ALGORITHM:**

Step 1: Start the process.

Step 2: Declare necessary variables.

Step 3: Define a function name “check\_prime()” with arguments of n .

Step 4: Set a flag value and increment for consonants and return the value to main function.

Step 5: Check the flag value and if prime increment entered value by 2 and call the function “check\_prime()” and display the result.

Step 6: If the number at step 4 is not prime then print the output as “is not a prime” and exit the code.

Step 7: Stop the process.

**SOURCE CODE:**

def check\_prime(n):

 val=0

 for i in range(2,n-1):

 if n%i==0:

 val+=1

 return val

n=int(input("Enter a positive integer :"))

val=check\_prime(n)

Page No:4

if val==0:

 val=check\_prime(n+2)

 if val==0:

 print(n," is a prime number and it's twin prime is ",n+2)

 else:

 print(n," is a prime number and it's twin prime does not exist")

else:

 print(n," is not a prime number ")

**RESULT:**

The above code has been executed successfully.

Page No:5

**OUTPUT:**

**AIM:**

Experiment No: 3 Page No: 6

**Program to** **display the Fibonacci series and multiplication table by using looping constructs.**

**.**

To write a program to display Fibonacci series and multiplication table by using looping constructs.

**ALGORITHM:**

Step 1: Start the process.

Step 2: Declare necessary variables.

Step 3: Print 0 and 1 as a start for our sequence.

Step 3: Using while loop set condition n not equals to ‘0’.

Step 4: In while loop add the previous values and store to third variable swap the values (ex. c=a+b ,a=b ,b=c) and print the value in same line and decrement ‘n’ after every alliteration.

Step 5: Stop the process.

**SOURCE CODE:**

n=int(input("Enter sequence value :"))

f\_val=n

a=0

b=1

print("Fibonacci series of ",n,":")

print(a,b,end=",")

while not(f\_val==0):

 c=a+b

 print(c,end=",")

 a=b

 b=c

 f\_val-=1

Page No :7

print("\nMultiplication table of ",n," :")

for i in range(1,11):

 print(i,"X",n,"=",i\*n)

**RESULT:**

The above code has been executed successfully.

Page No:8

**OUTPUT:**

**AIM:**

Experiment No:4 Page No: 9

**Program for converting decimal to octal, hexadecimals and vice versa by using functions.**

To write a program to convert decimal to octal, hexadecimal and vice versa.

**ALGORITHM:**

Step 1: Start the process.

Step 2: Declare necessary variables.

Step 3: Define the functions to convert decimal to octal and hexadecimal “conv\_oct()” and “conv\_hexa()”

Step 4: Converting decimal to octal.

1. Divide given decimal by 8 and obtain it’s remainder and quotient. Store the remainder.
2. Repeat the above step until quotient becomes zero.
3. Store the octal value and print.

Step 5: Converting decimal to hexadecimal.

1. Divide given decimal by 16 and obtain it’s remainder and quotient. Store the remainder.
2. Repeat the above step until quotient becomes zero.
3. Store the value in an array and print.

Step 6: Stop the process.

Page no: 10

**SOURCE CODE:**

def conv\_hexa(n):

 val=n

 hexa=[]

 i=0

 while not(val==0):

 temp=val%16

 val=val//16

 if temp<10:

 hexa.append(temp)

 i+=1

 else:

 hexa.append(chr(temp+55))

 i+=1

 print("The hexadecimal number of the decimal value is: ",end="")

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

 print(hexa[j-1],end="")

def conv\_oct(n):

 val=n

 oct\_num=0

 i=1

 while not(val==0):

 oct\_num+=(val%8)\*i

 val=val//8

 i\*=10

 print("The octal number of the decimal value is: ",oct\_num)

dec\_num=int(input("Enter a decimal value: "))

oct\_num=int(input("Enter an octal Value: "),8)

hex\_num=int(input("Enter a hexadecimal value: "),16)

conv\_oct(dec\_num)

conv\_hexa(dec\_num)

print("\nDecimal value for entered octalnumber is: ",int(oct\_num))

print("Decimal value for entered hexadecimal number is: ",int(hex\_num))

**RESULT:**

The above code has been executed successfully.

Page no:11

**OUTPUT:**

**AIM:**

Experiment No : 5 Page No : 12

**Function to compute the GCD of two numbers.**

To write a function to compute GCD of two numbers.

**ALGORITHM:**

Step 1: Start the process.

Step 2: Declare necessary variables and get those values from the user.

Step 3:Find the largest value among two values and say as ‘n’.

Step 4:Using for with i in range of 1 to n.

Step 5:Check if both the values are divided by same values of i then that is the GCD of those values display the value of GCD.

Step 6: Stop the process.

**SOURCE CODE:**

n1=int(input("Enter the number 1: "))

n2=int(input("Enter the number 2: "))

if n1>n2:

 n=n1

else:

 n=n2

for i in range(1,n):

 if (n1%i==0)and(n2%i==0):

 gcd=i

print("The GCD for the given values ",n1,"&",n2," is: ",gcd)

**RESULT:**

The above code has been executed successfully.

Page No: 13

**OUTPUT:**

**AIM:**

EXPERIMENT NO:6 Page No:14

**Function to perform sorting list of numbers**

To write a function to perform sorting list of numbers.

**ALGORITHM:**

Step1: Start the process.

Step2: Declare a list and get it’s values from user using for loop.

Step3:Sort the using nested for loop constructs and display in ascending and descending order.

Step4:Stop the process.

**SOURCE CODE:**

lis=[]

n=int(input("Enter the maximum number of values to be sorted "))

for i in range(n):

 val=int(input("Enter the value: "))

 lis.append(val)

print("List before sorting: ")

for i in range(n):

 print(lis[i],end=",")

print("\nList in ascending order :")

for i in range(n):

 for j in range(n):

 if lis[j]>lis[i]:

 s=lis[i]

 lis[i]=lis[j]

 lis[j]=s

for i in range(n):

 print(lis[i],end=",")

print("\nList in descending order: ")

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

 print(lis[i],end=",")

**RESULT:**

The above code has been executed successfully.

Page No: 15

**OUTPUT:**

**\*Note:** We are going to use **Key-Value Method** to find the day of a date

Experiment No:7 Page No: 16

**Program to print a simple calendar without using Calendar Module**

**AIM:**

To write a program to print simple Calendar without using Calendar module.

**ALGORITHM:**

Step 1: Start the process.

Step 2: Declare a dictionary and initialize from January to December using key values.

Step 3: Assign a list for number of days in leap year and nonleap year.

Step 4: Get the desired year from the user.

Step 5: Define a function named “print\_calendar()” to print the calendar.

Step 6: Find the starting day of the month using key-value method.

Step 7: Stop the process.

**SOURCE CODE:**

def print\_calendar(yy,month,year):

 k=0

 yn=yy%100

 rd=yn//4

 mm=[1,4,4,0,2,5,0,3,6,1,4,6]

 if (year[1]==29):

 mm[0]-=1

 mm[1]=1

 if (yy>=1700)and(yy<1800):

 cc=0

 elif (yy>=1800)and(yy<1900):

 cc=2

 if (yy>=1900)and(yy<2000):

 cc=4

 if (yy>=2000)and(yy<2100):

 cc6

 else:

 print(“Enter Year in range of 1700 to 2099”)

Page No:17

 for i in range(len(month)):

 day=0

 day=rd+1

 day+=mm[i]

 day+=cc

 day+=yn

 day=day%7

 print(“\n ”,month[i+1],yy)

 print(“SU MO TU WE TH SA ”

 if not(day==0) and(not( (day==1)) and(not(day==2)):

 for k in range(-1,day):

 print(“ ”,end=’ ’)

 elif day==0:

 for k in range(18):

 print(“ ”,end=’’)

 elif day==2:

 for k in range(day):

 print(“ ”,end=’’)

 for j in range(1,year[i]+1):

 k+=1

 if j < 10:

 print(j,end=’ ’)

 else:

 print(j,end=’’)

 if(k>7)

 print()

 k=0

yy=int(input(“Enter a year in range (1700 to 2099) to print Calendar for: ”))

month={1:’January’,2:’Febrauary’,3:’March’,4:’April’,5:’May’,6:’June’,7:’July’,8:’August’, 9:’September’,10:’October’,11:’November’,12:’December’}

lyr=[31,29,31,30,31,30,31,31,30,31,30,31]

nlyr=[31,28,31,30,31,30,31,31,30,31,30,31]

if(yy%4==0):

 if(yy%100==0):

 if(yy%400==0):

 print\_calendar(yy,month,lyr)

 else:

 print\_calendar(yy,month,lyr)

else:

 print\_calendar(yy,month,nlyr)

Page No: 18

**OUTPUT:**

 **AIM:**

Experiment No: 8 Page No :19

**Demonstrate inheritance in Python**

To write a program to demonstrate inheritance in python.

**ALGORITHM:**

Step 1: Start the process.

Step 2: Create a class with required attributes.

**SYNTAX:** class class\_name1:

Step 3 : Create another class as the sub class

**SYNTAX:** class class\_name2(class\_name 1

derived from the parent class (previous class)

Step 4: Access the parent class from sub class and enter some values and display them.

Step 5: Stop the process.

**SOURCE CODE:**

class Employee:

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

 print("Hello from Employee")

 self.ID=ID

 self.name=name

 def employeedetail(self):

 print("ID: %s"%self.ID)

 print("Name: %s"%self.name)

class Manager(Employee):

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

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

 print("Hello from Manager" )

 self.project=project

 def projectdetail(self):

 print("project : %s"%self.project)

obj=Manager(1,'James','python programing')

print("\_ \_ \_ \_ \_ \_ \_Employee Detail\_ \_ \_ \_ \_ \_ \_")

obj.employeedetail()

print("\_ \_ \_ \_ \_ \_ \_Project Detail\_ \_ \_ \_ \_ \_ \_")

obj.projectdetail()

**RESULT:**

The above code has been executed successfully.

Page No: 20

**OUTPUT:**

**AIM:**To write a program to create a text file using python file I/O. Read the content of the file and change them from lower to upper case characters. Write the updated content in another file and display it.

Experiment No:9 Page No: 21

 **Create a text file using python file I/O. Read the content of the file and change them from lower to upper case characters. Write the updated content in another file and display it.**

**ALGORITHM:**

Step 1: Start the process.

Step 2: Create a text file using open() function by opening that in append and assign them to a file pointer.

Step 3: Using write() function write the required content in the file and close it.

Step 4: Open the file in overwriting mode and convert the lower-case letters to upper case using string function and close the file.

Step 5: Now open the file and another new file in writing mode using open() function and copy the contents from file 1 to file 2 and close both the file.

Step 6: Open both files read and display its contents.

Step 7: Stop the process.

**SOURCE CODE:**

f=open("file1.txt","a")

f.write("This a demonstration of python i/o")

print("Create operation successful")

print("Write operation successful")

f.close()

f=open("file1.txt","r+b")

s=f.read()

f.seek(0)

f.write(s.upper())

print("overwriting successful")

f.close()

f=open("file1.txt","r")

f.close()

f=open("file1.txt","r")

f1=open("file2.txt","w")

f1.write(f.read())

f.close()

f1.close()

print("copied the contents successfully ")

Page No: 22

f=open("file1.txt")

f1=open("file2.txt")

print("Contents in file 1 \n ",f.read(),"\nContents in file 2 \n ",f1.read())

**RESULT:**

The above Code has been executed successfully.

Page No: 23

**OUTPUT:**

**AIM:**

Experiment No: 10 Page No: 24

**Program to demonstrate User defined Exception in Python**

To write a program to demonstrate User defined Exception.

**ALGORITHM:**

Step 1: Start the process.

Step 2: Create a class named Error and Exception as its attribute.

Step 3: Create required sub-class to perform the tasks.

Step 4: Using while with try and except method raise error for the number of exceptions occurring.

Step 5: Stop the Process.

**SOURCE CODE:**

class Error(Exception):

 pass

class valueToosmallerror(Error):

 pass

class valueToolargeerror(Error):

 pass

n=10

while True:

 try:

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

 if gnum<n:

 raise valueToosmallerror

 elif gnum>n:

 raise valueToolargeerror

 break

 except valueToosmallerror:

 print("This value is too small,try again")

 print()

 except valueToolargeerror:

 print("This Value is too large, try again")

 print()

print("Congrats! You have guessed it correctly")

**RESULT:**

The above code has been successfully executed.

Page No:25

**OUTPUT:**

**AIM:**

Experiment No: 12 Page No: 26

**Program to insert and retrieve data using MySQL.**

To write a program to insert and retrieve data using MySQL.

**ALGORITHM:**

Step 1: Start the process.

Step 2: Import MySQL. Connector to connect to sql server.

Step 3: Connect to the database and localhost using username and password.

Step 4: Create a table and insert data in it using sql queries and execute them using execute() function.

Step 5: Display the contents as list.

Step 6: Stop the process.

**SOURCE CODE:**

import mysql.connector

mydb = mysql.connector.connect(

 host="localhost",

 user="yourusername",

 password="yourpassword",

 database="mydatabase"

)

mycursor = mydb.cursor()

mycursor.execute("CREATE TABLE customers (name VARCHAR(255), address VARCHAR(255))")

sql = "INSERT INTO customers (name, address) VALUES (%s, %s)"
val = ("John", "Highway 21")
mycursor.execute(sql, val)

mycursor.execute("SELECT \* FROM customers")
myresult = mycursor.fetchall()
for x in myresult:
  print(x)

**RESULT:**

The above code has been executed successfully.

Page No: 27

**OUTPUT:**