INDEX

	user defined exceptions.	
8	**Write a Pandas program to join the two given data frames along rows. Sample Data frame may contain details of student like rollno , name , Total Marks.**	**13-14**

Tittle

7

Create a table electricity_bill(TariffCode, Customer_Name, Meter Number, Previous_Reading and Current_Reading) using MySQL and perform the followings.

a. **To accept the details of employees and store it in database.**
b. **To Update the Customer details by Meter Number.**
c. **Calculate Bill of Particular Customer using below criteria.**

Tariff Code	Units Consumed	Rate/Units
LT1	0-30	2.0
	31-100	3.0
	101-200	4.5
	Above 200	5.0
LT2	0-30	3.5
	31-100	5.0
	101-200	6.0
	Above 200	7.5

36-40

8

Consider following data and draw the bar graph using matplot library.(Use CSV or Excel).Add the data Using GUI.

Batsman	2017	2018	2019	2020
Virat Kohli	2501	1855	2203	1223
Steve Smith	2340	2250	2003	1153
Babar Azam	1750	2147	1896	1008
Rohit Sharma	1463	1896	1854	1638
Kane Williamson	1256	1854	1874	1974
Jos Butler	1125	1769	1769	1436

41-43

PART-A

/* ********** */

Aim :1. Write a program create list with N elements. find all unique elements in the list. If an element is found only once in the list, then add that element to the unique list.

Date :

/* ********** */

```
old_list=[]
unique_list=[]
n=int(input("Enter number of elements:"))
print("Enter elements")
for i in range(0,n):
    ele=int(input())
    old_list.append(ele)
print(f"The elements of list are{old_list}")
for x in old_list:
    if old_list.count(x)==1:
        unique_list.append(x)
print(f"The unique elements in the list are{unique_list}")
```

OUTPUT:

```
Enter number of elements:5
Enter elements
5
5
8
7
6
The elements of list are[5, 5, 8, 7, 6]
The unique elements in the list are[8, 7, 6]
```

/* ********** */

Aim :2. Program, using user-defined functions to find the area of rectangle, square, circle and triangle by accepting suitable input parameters from user.

Date :

/* ********** */

```
def rect(L,B):
   return(L*B)

def square(s):
   return(s*s)

def circle(r):
   return(3.14*r*r)

def triangle(b,h):
   return(0.5*b*h)

def main():
   L=int(input("Enter the Length of the Rectangle:"))
   B = int(input("Enter the Breath of the Rectangle:"))
   s = int(input("Enter the Side of the Square:"))
   r = int(input("Enter the Radius of the Circle:"))
   b = int(input("Enter the Base of the Triangle:"))
   h = int(input("Enter the Height of the Triangle:"))
   print(f"Area of Rectangle:{rect(L,B)}")
   print(f"Area of Square:{square(s)}")
   print(f"Area of Circle:{circle(r)}")
   print(f"Area of Triangle:{triangle(b,h)}")
if __name__=="__main__":
      main()
```

OUTPUT:

Enter the Length of the Rectangle:5

Enter the Breath of the Rectangle:9

Enter the Side of the Square:6

Enter the Radius of the Circle:4

Enter the Base of the Triangle:8

Enter the Height of the Triangle:8

Area of Rectangle:45

Area of Square:36

Area of Circle:50.24

Area of Triangle:32.0

/* ********** */

Aim :3. Consider a tuple t1= (1,2,5,7,9,2,4,6,8,10). Write a program to perform following operations:

a. **Print half the values of tuple in one line and the other half in the next line.**
b. **Print another tuple whose values are even numbers in the given tuple.**
c. **Concatenate a tuple t2= (11,13,15) with t1.**
d. **Return maximum and minimum value from this tuple.**

Date :

/* ********** */

```
t1=(1,2,5,7,9,2,4,6,8,10,12)
print("Tuple:",t1)
t2=(11,13,15)
print(f"Tuple:{t2}")
print("After Slicing")
n=int(len(t1)/2)
print(t1[:n])
print(t1[n:])
l1=list()
for i in t1:
    if i%2==0:
        l1.append(i)
    tuple_even=tuple(l1)
print(f"Even numbers:{tuple_even}")
t3=t1+t2
print(f"Tuple3:{t3}")
print(f"Maximum value:{max(t3)}")
print(f"Minimum value:{min(t3)}")
```

OUTPUT:

Tuple: (1, 2, 5, 7, 9, 2, 4, 6, 8, 10, 12)

Tuple:(11, 13, 15)

After Slicing

(1, 2, 5, 7, 9)

(2, 4, 6, 8, 10, 12)

Even numbers:(2, 2, 4, 6, 8, 10, 12)

Tuple3:(1, 2, 5, 7, 9, 2, 4, 6, 8, 10, 12, 11, 13, 15)

Maximum value:15

Minimum value:1

/* ********** */

Aim :4. Write a function that takes a sentence as input from the user and calculates the frequency of each letter. Use a variable of dictionary type to maintain the count.

Date :

/* ********** */

```
def char_frequency(str1):
    dict={}
    for n in str1:
        keys=dict.keys()
        if n in keys:
            dict[n]+=1
        else:
            dict[n]=1
    return dict
c=str(input("Enter character:\n"))
print(char_frequency(c))
```

OUTPUT:

Enter character:

Pycharm program

{'P': 1, 'y': 1, 'c': 1, 'h': 1, 'a': 2, 'r': 3, 'm': 2, ' ': 1, 'p': 1, 'o': 1, 'g': 1}

/* ********** */

Aim :5. Write a function nearly equal to test whether two strings are nearly equal. two strings a and b are nearly equal if one character change in b results in string a.

Date :

/* ********** */

```
def nearly_equal(s1,s2):
    if len(s1)!=len(s2):
        print("String are not nearly equal")
    elif s1==s2:
        print("Both are equal")
    else:
        n=len(s1)
        i=0
        count=0
        for i in range(n):
            if s1[i]==s2[i]:
                pass
            else:
                count=count+1
        if count==1:
            print("String are nearly equal")
        else:
            print("String are not nearly equal")
s1=input("Enter first String\n")
s2=input("Enter second String\n")
nearly_equal(s1,s2)
```

OUTPUT 1:	**OUTPUT 2:**	**OUTPUT 3:**
Enter first String	Enter first String	Enter fist String
Hello	hello	hello
Enter second String	Enter second String	Enter second String
Hello	hcllo	Mangalore
Both are equal	Strings are nearly equal	Strings are not nearly equal

/* ********** */

Aim :6. Write a program to create a text file and compute the number of characters, words and lines in a file.

Date :

/* ********** */

```
number_of_words=0
number_of_lines=0
number_of_characters=0
file_path="example3.txt"

with open(file_path,'w+')as file:
    file.write("Hello,\nWorld!")

with open(file_path,'r')as file:
    for i in file:
        number_of_words+=len(i.split())
        number_of_lines+=1
        number_of_characters+=len(i.strip())

print("No of words:",number_of_words)
print("No of lines:",number_of_lines)
print("No of characters:",number_of_characters)
```

OUTPUT:

No of words: 2

No of lines: 2

No of characters: 12

/* ********** */

Aim :7. Program using user defined exception class that will ask the user to enter a number until he guesses a stored number correctly. To help them figure it out, a hint is provided whether their guess is greater than or less than the stored number using user defined exceptions.

Date :

/* ********** */

```
import random
number=random.randint(1,10)
class Error(Exception):
    pass
class ValueSmallError(Error):
    pass
class ValueLargeError(Error):
    pass
while True:
    try:
        guess=int(input("Enter a number:"))
        if guess<number:
            raise ValueSmallError
        elif guess>number:
            raise ValueLargeError
        break

    except ValueSmallError:
        print("This value is small,try again")
    except ValueLargeError:
        print("This value is larger,try again")
print("Congratulations! You guessed it correctly")
```

OUTPUT:

```
Enter a number:8

This value is larger,try again

Enter a number:5

This value is small,try again

Enter a number:2

This value is small,try again

Enter a number:7

This value is larger,try again

Enter a number:6

Congratulations! You guessed it correctly
```

/* ********** */

Aim :8. Write a Pandas program to join the two given data frames along rows. Sample Data frame may contain details of student like rollno , name , Total Marks.

Date :

/* ********** */

```
import pandas as pd
data1={
   'RollNo':[1,2,3],
   'Name':['Amogh','Babitha','Chaitanya'],
   'TotalMarks':[95,88,96]
}
data2={
   'RollNo':[4,5,6],
   'Name':['David','Eshan','Ganesh'],
   'TotalMarks':[82,91,70]
}
df1=pd.DataFrame(data1)
df2=pd.DataFrame(data2)
print("Orignal DataFrame:")
print()
print(df1)
print("_"*50)
print(df2)
print("_"*50)
print("After joining the said two dataframes along rows:")
print("_"*50)
result_df=pd.concat([df1,df2],ignore_index=True)
print(result_df)
```

OUTPUT:

```
Orignal DataFrame:

   RollNo       Name  TotalMarks
0       1      Amogh          95
1       2    Babitha          88
2       3  Chaitanya          96
__________________________________________________
   RollNo    Name  TotalMarks
0       4   David          82
1       5   Eshan          91
2       6  Ganesh          70
```

After joining the said two dataframes along rows:

	RollNo	Name	TotalMarks
0	1	Amogh	95
1	2	Babitha	88
2	3	Chaitanya	96
3	4	David	82
4	5	Eshan	91
5	6	Ganesh	70

PART-B

/* ********** */

Aim :1. Program to create a class Employee with empno, name, depname, designation, age and salary and perform the following function.

i. **Accept details of N employees**
ii. **Search given employee using empno**
iii. **Display employee details in neat format.**

Date :

/* ********** */

```
class employee:
    def get_data(self):
        self.emp_no=int(input("Enter employee number:"))
        self.namame=input("Enter name:")
        self.design=input("Enter designation:")
        self.dept=input("Enter department:")
        self.age=input("Enter age:")
        self.basic=int(input("Enter basic salary:"))

    def display(self):
        print(self.emp_no,"\t",self.namame,"\t",self.design,"\t",self.dept,"\t",self.age,"\t",self.basic)

    def search(self,id):
        if id==self.emp_no:
            return True
        else:
            return False
n=int(input("Enter the number of employee:"))
lst=[]
for i in range(n):
    e1=employee()
    e1.get_data()
    lst.append(e1)
while True:
    print("1.To display employee information\n2.Search for employee")
    ch=int(input("Enter your choice:"))
    if ch==1:
        print("empno\tname\tdesignation\tdep[argument\tage\tsalary")
        for e in lst:
            e.display()
    elif ch==2:
        id=int(input("Enter the employee number to be searched:"))
        for i in lst:
            found=i.search(id)
            if found:
                print("Employee found they are:")
                i.display()
                break
            else:
```

```
            print("Employee details not found:")
    else:

        print("Invaild choice")
        exit(0)
```

OUTPUT:

Enter the number of employee:2

Enter employee number:101

Enter name:Anisha

Enter designation:Manager

Enter department:IT

Enter age:25

Enter basic salary:85000

Enter employee number:102

Enter name:Dhvani

Enter designation:accountant

Enter department:wipro

Enter age:25

Enter basic salary:80000

1. To display employee information

2. Search for employee

Enter your choice:1

empno	name	designation	dep[argument	age	salary
101	Anisha	Manager	IT	25	85000
102	Dhvani	accountant	wipro	25	80000

1. To display employee information

2. Search for employee

Enter your choice:2

Enter the employee number to be searched:101

Employee found they are:

101	Anisha	Manager	IT	25	85000

1. To display employee information

2. Search for employee

Enter your choice:2

Enter the employee number to be searched:088

Employee deatils not found

1. To display employee information

2. Search for employee

Enter your choice:5

Invalid option

/* ********** */

Aim :2. Write a program menu driven to create a BankAccount class. class should support the following methods for i) Deposit ii) Withdraw iii) GetBalanace . Create a subclass SavingsAccount class that behaves just like a BankAccount, but also has an interest rate and a method that increases the balance by the appropriate amount of interest.

Date :

/* ********** */

```
class bankaccount:
    def __init__(self):
        self.balance=int(input("Enter the initial amount:"))
        print("Your account created")
    def deposite(self):
        amount=int(input("Enter the deposit amount:"))
        self.balance+=amount
        print("Your new balance:",self.balance)

    def withdraw(self):
        amount=int(input("Enter the amount to withdraw"))
        if(amount>=self.balance):
            print("Insufficent balance")
        else:
            self.balance-=amount
            print("Your remaining balance:",self.balance)

    def GetBalance(self):
        print("Your balance:",self.balance)

class savingsaccount(bankaccount):
    def __init__(self):
        bankaccount.__init__(self)
    def interest(self):
        self.rate=float(input("Enter rate of interest:"))
        self.balance=self.balance+(self.balance*self.rate/100)
        print("Your new balance:",self.balance)

account=savingsaccount()
while(1):
    print("1.Deposit\n2.Withdraw\n3.GetBalance\n4.Interest\n5.Exit")
    c=int(input("Enter your choice:"))
    if c==1:
        account.deposite()
    elif c==2:
        account.withdraw()
    elif c==3:
        account.GetBalance()
    elif c==4:
        account.interest()
```

```
elif c==5:
        break
    else:
        print("Invalid choice")
```

OUTPUT:

Enter the initial amount:15000

Your account created

1. Deposit
2. Withdraw
3. GetBalance
4. Interest
5. Exit

Enter your choice:1

Enter the deposit amount:2000

Your new balance: 17000

1. Deposit
2. Withdraw
3. GetBalance
4. Interest
5. Exit

Enter your choice:2

Enter the amount to withdraw2000

Your remaining balance: 15000

1. Deposit
2. Withdraw
3. GetBalance
4. Interest
5. Exit

Enter your choice:2

Enter the amount to withdraw15000

Insufficent balance

1. Deposit
2. Withdraw

3. GetBalance
4. Interest
5. Exit

Enter your choice:3

Your balance: 15000

1. Deposit
2. Withdraw
3. GetBalance
4. Interest
5. Exit

Enter your choice:2

Enter the amount to withdraw15001

Insufficent balance

1. Deposit
2. Withdraw
3. GetBalance
4. Interest
5. Exit

Enter your choice:4

Enter rate of interest:12

Your new balance: 16800.0

1. Deposit
2. Withdraw
3. GetBalance
4. Interest
5. Exit

Enter your choice:8

Invalid choice

1. Deposit
2. Withdraw
3. GetBalance
4. Interest
5. Exit

Enter your choice:5

/* ********** */

Aim :3. Create a GUI to input Principal amount, rate of interest and number of years, Calculate Compound interest. When button submit is pressed Compound interest should be displayed in a textbox. When clear button is pressed all contents should be cleared.

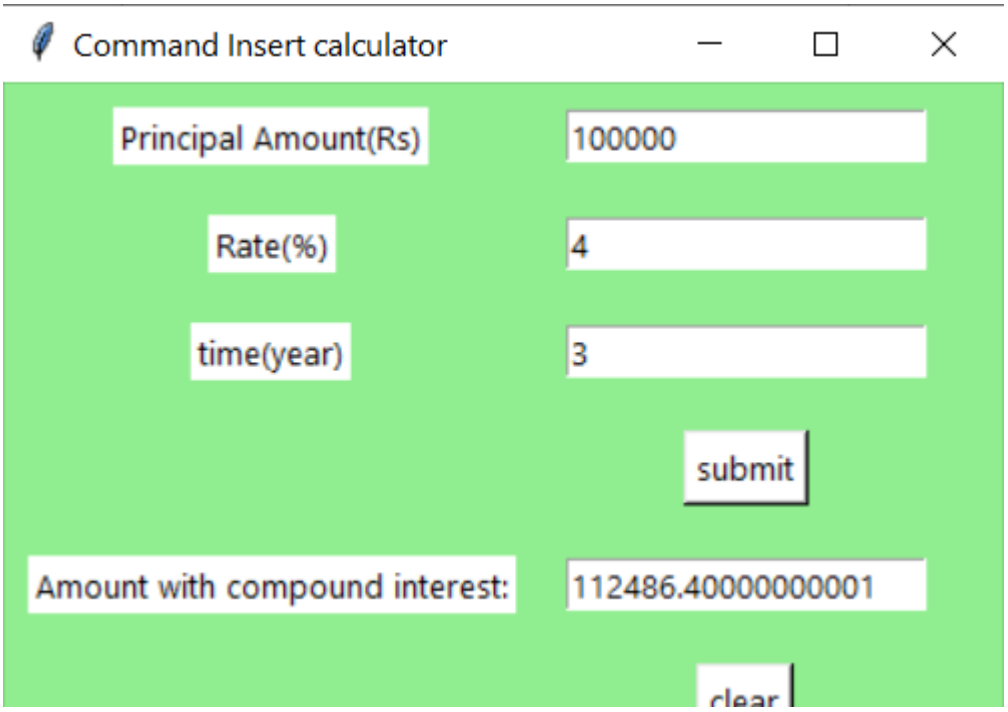
Date :

/* ********** */

```
from tkinter import*
def clear_all():
    principle_field.delete(0,END)
    rate_field.delete(0,END)
    time_field.delete(0,END)
    compound_field.delete(0,END)
    principle_field.focus_set()
def calculate_ci():
    principal=int(principle_field.get())
    rate=int(rate_field.get())
    time=int(time_field.get())
    ci=principal*(pow((1+rate/100),time))
    compound_field.insert(10,ci)
if __name__=="__main__":
    gui=Tk()
    gui.configure(background="lightgreen")
    gui.title("Command Insert calculator")
    gui.geometry("400x250")
    label1=Label(gui,text="Principal Amount(Rs)",fg='black',bg='white')
    label2=Label(gui,text="Rate(%)",fg='black',bg='white')
    label3=Label(gui,text="time(year)",fg='black',bg='white')
    label4=Label(gui,text="Amount with compound interest:",fg='black',bg='white')
    label1.grid(row=1,column=0,padx=10,pady=10)
    label2.grid(row=2,column=0,padx=10,pady=10)
    label3.grid(row=3,column=0,padx=10,pady=10)
    label4.grid(row=5,column=0,padx=10,pady=10)
    principle_field=Entry(gui)
    rate_field=Entry(gui)
```

```
time_field=Entry(gui)
compound_field=Entry(gui)
principle_field.grid(row=1,column=1,padx=10,pady=10)
rate_field.grid(row=2,column=1,padx=10,pady=10)
time_field.grid(row=3,column=1,padx=10,pady=10)
compound_field.grid(row=5,column=1,padx=10,pady=10)
button1=Button(gui,text='submit',fg='black',bg='white',command=calculate_ci)
button2=Button(gui,text='clear',fg='black',bg='white',command=clear_all)
button1.grid(row=4,column=1,pady=10)
button2.grid(row=6,column=1,pady=10)
gui.mainloop()
```

OUTPUT:



/* ********** */

Aim :4. Write a GUI program to implement Simple Calculator.

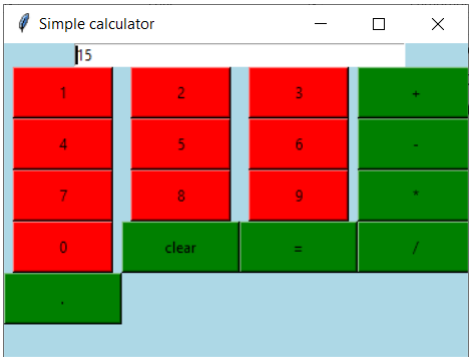
Date :

/* ********** */

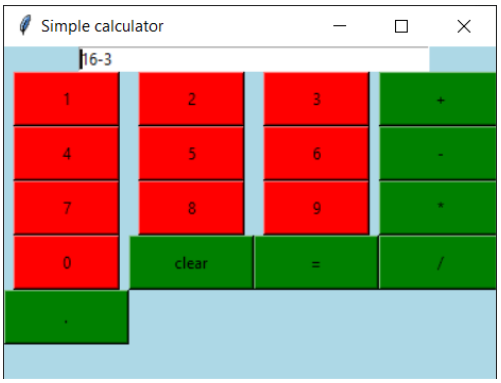
```
from tkinter import*
expression=""
def press(num):
    global expression
    expression=expression+str(num)
    equation.set(expression)
def equalpress():
    try:
        global expression
        total=str(eval(expression))
        equation.set(total)
        expression=""
    except:
        equation.set("Error")
        expression=""
def clear():
    global expression
    expression=""
    equation.set("")
if __name__=="__main__":
    gui=Tk()
    gui.configure(background="light blue")
    gui.title("Simple calculator")
    gui.geometry("370x250")
    equation=StringVar()
    expression_field=Entry(gui,textvariable=equation)
    expression_field.grid(columnspan=4,ipadx=70)
    button1=Button(gui,text='1',fg='black',bg='red',command=lambda:press(1),height=2,width=10)
    button1.grid(row=2,column=0)
    button2=Button(gui,text='2',fg='black',bg='red',command=lambda:press(2),height=2,width=10)
    button2.grid(row=2,column=1)
    button3=Button(gui,text='3',fg='black',bg='red',command=lambda:press(3),height=2,width=10)
    button3.grid(row=2,column=2)
    button4=Button(gui,text='4',fg='black',bg='red',command=lambda: press(4),height=2, width=10)
    button4.grid(row=3,column=0)
    button5=Button(gui,text='5',fg='black',bg='red',command=lambda:press(5),height=2,width=10)
    button5.grid(row=3,column=1)
    button6=Button(gui,text='6',fg='black',bg='red',command=lambda:press(6),height=2,width=10)
    button6.grid(row=3,column=2)
    button7=Button(gui,text='7',fg='black',bg='red',command=lambda:press(7),height=2,width=10
```

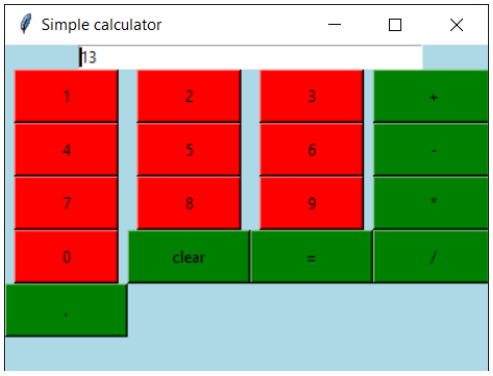
```
button7.grid(row=4,column=0)
button8=Button(gui,text='8',fg='black',bg='red',command=lambda:press(8),height=2,width=10)
button8.grid(row=4,column=1)
button9=Button(gui,text='9',fg='black',bg='red',command=lambda:press(9),height=2,width=10)
button9.grid(row=4,column=2)
button0=Button(gui,text='0',fg='black',bg='red',command=lambda:press(0),height=2,width=10)
button0.grid(row=5,column=0)
plus=Button(gui,text=("+"),fg='black',bg='Green',command=lambda:press("+"),height=2,width=12)
plus.grid(row=2,column=3)
minus=Button(gui,text=("-"),fg='black',bg='Green',command=lambda:press("-"),height=2,width=12)
minus.grid(row=3,column=3)
multiply=Button(gui,text=("*"),fg='black',bg='Green',command=lambda:press("*"),height=2,width=12)
multiply.grid(row=4,column=3)
divide=Button(gui,text=("/"),fg='black',bg='Green',command=lambda:press("/"),height=2,width=12)
divide.grid(row=5,column=3)
equal=Button(gui,text=("="),fg='black',bg='Green',command=equalpress,height=2,width=12)
equal.grid(row=5,column=2)
clear=Button(gui,text='clear',fg='black',bg='Green',command=clear,height=2,width=12)
clear.grid(row=5,column=1)
decimal=Button(gui,text=("."),fg='black',bg='Green',command=lambda:press("."),height=2,width=12)
decimal.grid(row=6,column=0)
gui.mainloop()
```

OUTPUT1:

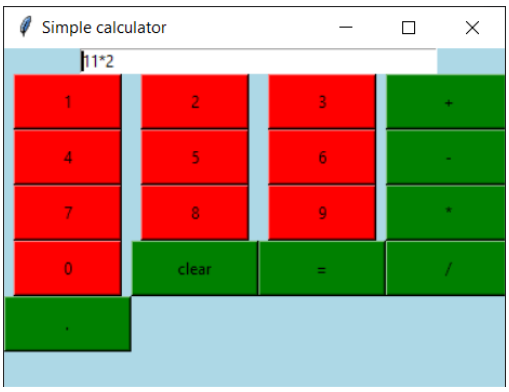


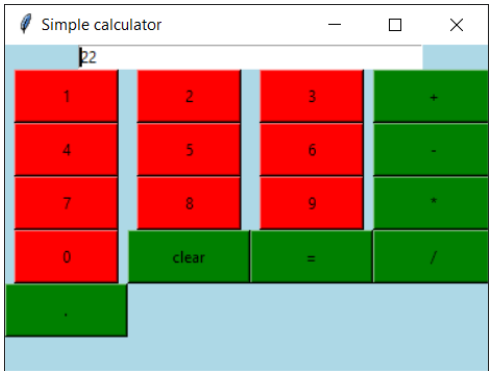
OUTPUT 2:



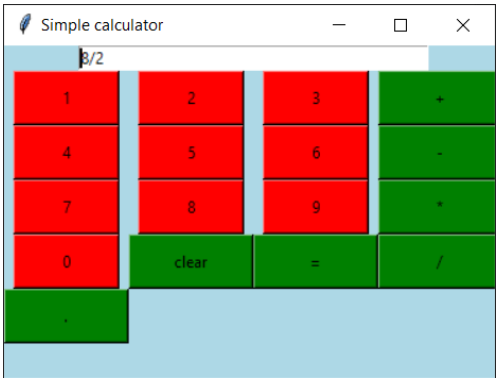


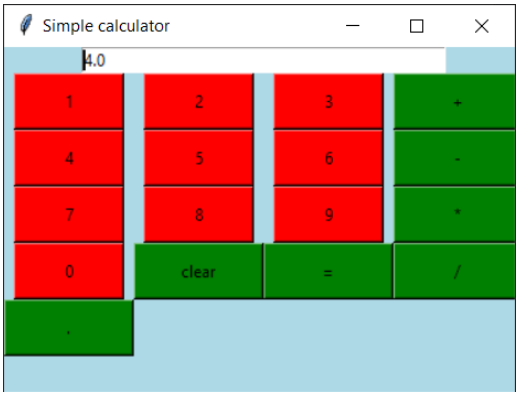
OUTPUT 3:



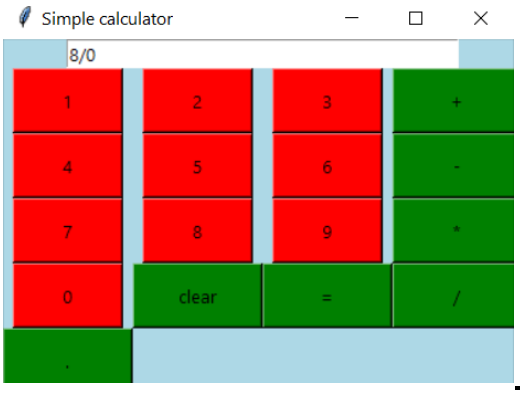


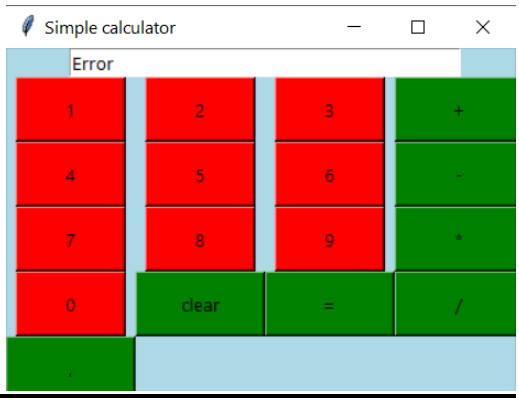
OUTPUT 4:





OUTPUT 5:





/* ********** */

Aim :5. Create a table student table (regno, name and marks in 3 subjects) using MySQL and perform the followings

a. **To accept the details of students and store it in database.**
b. **To display the details of all the students**
c. **Delete particular student record using regno.**

Date :

/* ********** */

```
import sqlite3
con=sqlite3.connect('student.db')
cursor=con.cursor()

def student_exist(regno):
    sql="'create table if not exists student(regno interger primary key,name varchar(10),mark1 int,mark2
int,mark3 int)'"
    cursor.execute(sql)
    data=(regno,)
    sql="select*from student where regno=?"
    cursor.execute(sql,data)
    result=cursor.fetchall()
    if len(result)>0:
        return True
    else:
        return False

def add_student():
    regno=int(input("Enter student Register number:"))
    if(student_exist(regno)==True):
        print("Student Already Exists\n Try again\n")
    else:
        name=input("Enter student name:")
        mark1=int(input("Enter mark in subject1:"))
        mark2=int(input("Enter mark in subject2:"))
        mark3=int(input("Enter mark in subject3:"))
        data=(regno,name,mark1,mark2,mark3)
        sql="insert into student values(?,?,?,?,?)"
        cursor.execute(sql,data)
        con.commit()
        print("Student added successfully")

def display_student():
    cursor.execute("select*from student")
    result=cursor.fetchall()
    if len(result)>0:
        for i in result:
            print("Student Regno:",i[0])
            print("Student Name:", i[1])
```

```
            print("Mark in subject1:", i[2])
            print("Mark in subject2:", i[3])
            print("Mark in subject3:", i[4])
            print('_____________________________')
    else:
        print("No record exists")

def remove_student():
    regno=input("Enter the register number of student to be removed:")
    if(student_exist(regno)==False):
        print("Student does not exist\n Try agian\n")
    else:
        sql="delete from student where regno=?"
        data=(regno,)
        cursor.execute(sql,data)
        con.commit()
        print("Student removed")

def menu():
    print("1 To add new student")
    print("2 To display all student delails")
    print("3 To remove particular student")
    print("4 To exit")
    ch=int(input("Enter your choice:"))
    if(ch==1):
        add_student()
    elif ch==2:
        display_student()
    elif ch==3:
        remove_student()
    elif ch==4:
        exit()
    else:
        print("Invalid choice")
    menu()
menu()
```

OUTPUT:

1 To add new student

2 To display all student delails

3 To remove particular student

4 To exit

Enter your choice:1

Enter student Register number:095

Enter student name:Karthik

Enter mark in subject1:85

```
Enter  mark  in  subject2:99

Enter  mark  in  subject3:96

Student added successfully

1 To add new student

2 To display all student delails

3 To remove particular student

4 To exit

Enter your choice:1

Enter student Register number:095

Student Already Exists

 Try again


1 To add new student

2 To display all student delails

3 To remove particular student

4 To exit

Enter your chioce:2

Student Regno: 95

Student Name: Karthik

Mark in subject1: 85

Mark in subject2: 99

Mark in subject3: 96

_____

1 To add new student

2 To display all student delails

3 To remove particular student

4 To exit

Enter your choice:095

Invalid choice


1 To add new student

2 To display all student delails

3 To remove particular student

4 To exit
```

```
Enter your choice:3
Enter the register number of student to be removed:098
Student does not exist
 Try agian

1 To add new student
2 To display all student delails
3 To remove particular student
4 To exit
Enter your choice:3
Enter the register number of student to be removed:095
Student removed

1 To add new student
2 To display all student delails
3 To remove particular student
4 To exit
Enter your chioce:2
No record exists

1 To add new student
2 To display all student delails
3 To remove particular student
4 To exit
Enter your choice:3
Enter the register number of student to be removed:095
Student does not exist
 Try agian
1 To add new student
2 To display all student delails
3 To remove particular student
4 To exit
Enter your choice:4
```

/* ********** */

Aim :6. Create a table employee (empno, name and salary) using MySQL and perform the followings

a. **To accept the details of employees and store it in database.**
b. **To display the details of a specific employee**
c. **To display employee details whose salary lies within a certain range**

Date :

/**********/

```
import sqlite3
con=sqlite3.connect('employee.db')
cursor=con.cursor()

def employe_exist(empid):
    sql='''create table if not exists employee(empid integer primary key,empname varchar(10),salary int)'''
    cursor.execute(sql)
    data=(empid,)
    sql="select*from employee where empid=?"
    cursor.execute(sql,data)
    result=cursor.fetchall()
    if len(result)>0:
        return True
    else:
        return False
def add_employee():
    empid=int(input("Enter Employe ID:"))
    if (employe_exist(empid)==True):
        print("Employee already exixts\n Try Again\n")
    else:
        empname=input("Enter Employee Name:")
        salary=int(input("Enter Enployee Salary:"))
        data=(empid,empname,salary)
        sql="Insert or replace into employee values(?,?,?)"
        cursor.execute(sql,data)
        con.commit()
        print("Employee added succecfully")
def display_employee():
    min=input("Enter min Salary:")
    max=input("Enter max Salary:")
    sql="select*from employee where salary between ? and?"
    data=(min,max)
    cursor.execute(sql,data)
    r=cursor.fetchall()
    if len(r)>0:
        for i in r:
            print("Employee ID:",i[0])
            print("Employee Name:",i[1])
            print("Employee Salary:",i[2])
            print("_________________________")
```

```
    else:
        print("Rcord Not Exists")
def search_employee():
    empid=input("Enter employee ID to be searched")
    sql='select*from employee where empid=?'
    data=(empid,)
    cursor.execute(sql,data)
    r=cursor.fetchone()
    if r:
        print("Employee ID:",r[0])
        print("Employee Name:",r[1])
        print("Employee Salary:",r[2])
        print("_____")
    else:
        print("Records not found")
def menu():
    print("select")
    print("1.To Add new Employee")
    print("2.To search Employee")
    print("3.To Fetch employee details whose salary lies with in a ceartain range")
    print("4.To exit")
    ch=int(input("Enter your choice:"))
    if(ch==1):
        add_employee()
    elif ch==2:
        search_employee()
    elif ch==3:
        display_employee()
    elif ch==4:
        exit()
    else:
        print("Invalid choice")
    menu()
menu()
```

OUTPUT:

```
select

1. To Add new Employee

2. To search Employee

3. To Fetch employee details whose salary lies with in a ceartain range

4. To exit

Enter your choice:1

Enter Employe ID:0011

Enter Employee Name:Avish
```

Enter Enployee Salary:85000

Employee added succecfully

select

1. To Add new Employee
2. To search Employee
3. To Fetch employee details whose salary lies with in a ceartain range
4. To exit

Enter your choice:1

Enter Employe ID:0011

Employee already exixts

Try Again

select

1. To Add new Employee
2. To search Employee
3. To Fetch employee details whose salary lies with in a ceartain range
4. To exit

Enter your choice:2

Enter employee ID to be searched101

Employee ID: 101

Employee Name: anisha

Employee Salary: 78999

---

select

1. To Add new Employee
2. To search Employee
3. To Fetch employee details whose salary lies with in a ceartain range
4. To exit

Enter your choice:2

Enter employee ID to be searched103

Records not found

select

1. To Add new Employee
2. To search Employee
3. To Fetch employee details whose salary lies with in a ceartain range
4. To exit

Enter your choice:3

Enter min Salary:12000

Enter max Salary:35000

Employee ID: 112

Employee Name: manoj

Employee Salary: 25000

---

select

1. To Add new Employee
2. To search Employee
3. To Fetch employee details whose salary lies with in a ceartain range
4. To exit

Enter your choice:3

Enter min Salary:600

Enter max Salary:850

Record Not Exists

select

1. To Add new Employee
2. To search Employee
3. To Fetch employee details whose salary lies with in a ceartain range
4. To exit

Enter your choice:5

Invalid choice

select

1. To Add new Employee
2. To search Employee
3. To Fetch employee details whose salary lies with in a ceartain range
4. To exit

Enter your choice:4

/* ********** */

Aim :7. Create a table electricity_bill(TariffCode, Customer_Name, Meter Number, Previous_Reading and Current_Reading) using MySQL and perform the followings

a. **To accept the details of employees and store it in database.**
b. **To Update the Customer details by Meter Number.**
c. **Calculate Bill of Particular Customer using below criteria.**

Date :

/* ********** */

```
import sqlite3

file="mydb2.db"

con=sqlite3.connect("mydb2.db")

mycursor=con.cursor()

mycursor.execute("CREATE TABLE IF NOT EXISTS ebill (tcode char(20),cname char(20),mno   INT,pmr INT,cmr INT)")

while(1):

   print("1. for insert data to consumer table\n2. To update customer details\n3. To calcuate customer bill\n4. for exists")

   c=int(input("Enter your choice:"))

   if(c==1):

      print("Enter tariff code:",end=" ")

      tcode1=input()

      print("Enter customer name:",end=" ")

      cname1 = input()

      print("Enter meter number:",end=" ")

      mno1 = int(input())

      print("Enter previous meter reading:",end=" ")

      pmr1 = int(input())

      print("Enter current meter reading:",end=" ")

      cmr1 =int(input())

      mycursor.execute('insert into ebill values("%s","%s","%i","%i","%i")'%(tcode1,cname1,mno1,pmr1,cmr1))

      print("Data inserted successfuly")

      con.commit()

   elif(c==2):
```

```
    print("Enter meter number to update:",end=" ")
    mno1=int(input())
    sql="select * from ebill where mno='%d'"
    mycursor.execute(sql %mno1)
    myresult=mycursor.fetchall()
    for x in myresult:
        print(x)
    if len(myresult)==0:
        print(f"Customer with this meter number{mno1} does not exist")
    else:
        print("Enter update tariff code:",end=" ")
        tcode1=input()
        print("Enter update customer name:",end=" ")
        cname1 =str(input())
        print("Enter previos meter reading:",end=" ")
        pmr1 = int(input())
        print("Enter current meter reading:",end=" ")
        cmr1 =int(input())
        sql="UPDATE ebill SET tcode='%s',cname='%s',pmr=%i,cmr=%i where mno=%i"
        mycursor.execute(sql % (tcode1,cname1,pmr1,cmr1,mno1))
        con.commit()
        print(mycursor.rowcount,"records(s) affected")
elif(c==3):
    print("Enter meter number to calculate bill:",end=" ")
    mno1=int(input())
    sql="SELECT * FROM ebill where mno='%d'"
    mycursor.execute(sql%mno1)
    myresult=mycursor.fetchall()
    if len(myresult)==0:
        print(f"customer number {mno1}does not exist")
    else:
        for row in  myresult:
```

```
            tcode1=row[0]
            cname1 = row[1]
            mno1 = row[2]
            pmr1 = row[3]
            cmr1 = row[4]
            uc=cmr1-pmr1
        if tcode1=="LT1"or tcode1=="lt1":
            if uc>=0 and uc<=30:
                bill=uc*2.0
            elif uc >=31 and uc <=100:
                bill =(uc-30)*3.5+60
            elif uc >=101 and uc <=200:
                bill =(uc-100)*4.5+305
        else:
            bill=(uc-200)*5.0+755
        if tcode1 == "LT2" or tcode1 == "lt2":
            if uc >= 0 and uc <= 30:
                bill =uc * 3.5
            elif uc >= 31 and uc <= 100:
                bill =(uc-30) * 5.0+105
            elif uc >= 101 and uc <= 200:
                bill =(uc-100)* 6.0+455
        else:
            bill =(uc-200)*7.5+1055
            print(f'Tariff code is:{tcode1}')
            print(f'Customer number is:{cname1}')
            print(f'Meter number is:{mno1}')
            print(f'Previous meter reading is:{pmr1}')
            print(f'Current meter reading is:{cmr1}')
            print(f'Total bill is:{bill}')
    elif(c==4):
        break
```

```
    else:
        print("Invalid choice")
```

OUTPUT:

1. For insert data to consumer table
2. To update customer details
3. To calcuate customer bill
4. For exist

Enter your choice:1

Enter tariff code: LT1

Enter customer name: Shamith

Enter meter number: 654

Enter previous meter reading: 50

Enter current meter reading: 75

Data inserted successfuly

1. For insert data to consumer table
2. To update customer details
3. To calcuate customer bill
4. For exist

Enter your choice:2

Enter meter number to update: 844

Customer with this meter number844 does not exist

1. For insert data to consumer table
2. To update customer details
3. To calcuate customer bill
4. For exist

Enter your choice:2

Enter meter number to update: 654

('LT1', 'Shamith', 654, 50, 75)

Enter update tariff code: LT2

Enter update customer name: Shamith Dharmapal saliyan

Enter previos meter reading: 75

Enter current meter reading: 100

1 Records(s) affected

1. For insert data to consumer table
2. To update customer details
3. To calcuate customer bill
4. For exist

Enter your choice:3

Enter meter number to calculate bill: 654

Tariff code is:LT2

Customer number is:Shamith Dharmapal saliyan

Meter number is:654

Previous meter reading is:75

Current meter reading is:100

Total bill is:-257.5

1. For insert data to consumer table
2. To update customer details
3. To calcuate customer bill
4. For exist

Enter your choice:654

Invalid choice

1. For insert data to consumer table
2. To update customer details
3. To calcuate customer bill
4. For exist

Enter your choice:4

/* ********** */

Aim :8. Consider following data and draw the bar graph using matplot library.(Use CSV or Excel).Add the data Using GUI.

Batsman	2017	2018	2019	2020
Virat Kohli	2501	1855	2203	1223
Steve Smith	2340	2250	2003	1153
Babar Azam	1750	2147	1896	1008
Rohit Sharma	1463	1896	1854	1638
Kane Williamson	1256	1854	1874	1974
Jos Butler	1125	1769	1769	1436

Date :

/* ********** */

```
from tkinter import*
import openpyxl
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

def calculate_ci():
    wBook=openpyxl.load_workbook("practise1.xlsx")
    b=wBook["Sheet1"]
    p=player.get()
    y1=y2017.get()
    y2=y2018.get()
    y3=y2019.get()
    y4=y2020.get()
    data=[p,y1,y2,y3,y4]
    b.append(data)
    wBook.save("practise1.xlsx")
    print('All data inserted successfully')

def graph():
    e1=pd.read_excel("practise1.xlsx","Sheet1")
```

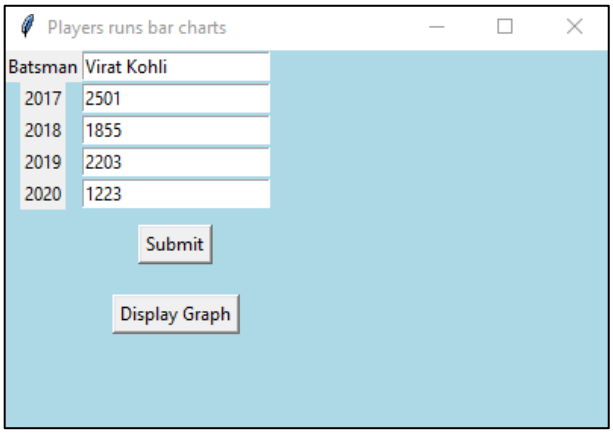
```
    X=np.arange(len(e1['player']))
    plt.figure(figsize=(10,6))
    width=0.15
    plt.bar(x=e1['player'],height=e1[2017],width=width,label='2017',color="red")
    plt.bar(X+0.20, height=e1[2018], width=width,color="green",label='2018')
    plt.bar(X+0.40, height=e1[2019], width=width, color="pink", label='2019')
    plt.bar(X+0.60, height=e1[2020], width=width, color="yellow", label='2020')
    plt.legend()
    plt.title('Players runs bar chart')
    plt.xlabel('Players')
    plt.ylabel('Runs')
    plt.savefig('Players runs.png')
    plt.show()
root=Tk()
root.configure(background='light blue')
root.geometry("400x250")
root.title("Compound Inter")
label1=Label(root,text="player:").grid(row=1,column=0)
label2=Label(root,text="2017:").grid(row=2,column=0)
label3=Label(root,text="2018:").grid(row=3,column=0)
label4=Label(root,text="2019:").grid(row=4,column=0)
label5=Label(root,text="2020:").grid(row=5,column=0)
player=Entry(root)
y2017=Entry(root)
y2018=Entry(root)
y2019=Entry(root)
y2020=Entry(root)
player.grid(row=1,column=1)
y2017.grid(row=2,column=1)
y2018.grid(row=3,column=1)
y2019.grid(row=4,column=1)
y2020.grid(row=5,column=1)
```

```
button1=Button(root,text="Submit",command=calculate_ci).grid(row=6,column=1,pady=10)

button2=Button(root,text="Display Graph",command=graph).grid(row=7,column=1,pady=10)

root.mainloop()
```

OUTPUT:



	A	B	C	D	E
1	Batsman	2017	2018	2019	2020
2	Virat Kohli	2501	1855	2203	1223
3	Steve Smith	2340	2250	2003	1153
4	Babar Azam	1750	2147	1896	1008
5	Rohit Sharma	1463	1985	1854	1638
6	Kane Williamson	1256	1785	1874	1974
7	Jos Butler	1125	1853	1769	1436