import sqlite3

def connect():

       con = sqlite3.connect('fee.db')

       cur = con.cursor()

       cur.execute('CREATE TABLE IF NOT EXISTS fee(id INTEGER PRIMARY KEY, recpt integer, name text, admsn text, date integer, \

                    branch text, sem text, total integer, paid integer, due integer)')

       con.commit()

       con.close()

def insert(recpt = ' ', name = ' ', admsn = ' ', date = ' ', branch = ' ', sem = ' ', total = ' ', paid = ' ', due = ' '):

       con = sqlite3.connect('fee.db')

       cur = con.cursor()

       cur.execute('INSERT INTO fee VALUES (NULL,?,?,?,?,?,?,?,?,?)',(recpt,name,admsn,date,branch,sem,total,paid,due))

       con.commit()

       con.close()

def view():

       con = sqlite3.connect('fee.db')

       cur = con.cursor()

       cur.execute('SELECT \* FROM fee')

       row = cur.fetchall()

       return row

       con.commit()

def delete(id):

       con = sqlite3.connect('fee.db')

       cur = con.cursor()

       cur.execute('DELETE FROM fee WHERE id = ?',(id,))

       con.commit()

       con.close()

def update(id,recpt = ' ', name = ' ', admsn = ' ', date = ' ', branch = ' ', sem = ' ', total = ' ', paid = ' ', due = ' '):

       con = sqlite3.connect('fee.db')

       cur = con.cursor()

       cur.execute('UPDATE fee SET recpt = ? OR name = ? OR admsn = ? OR date = ? OR branch = ? OR sem = ? OR total = ? OR \

                    paid = ? OR due = ?',(recpt,name,admsn,date,branch,sem,total,paid,due))

       con.commit()

       con.close()

def search(recpt = ' ', name = ' ', admsn = ' ', date = ' ', branch = ' ', sem = ' ', total = ' ', paid = ' ', due = ' '):

       con = sqlite3.connect('fee.db')

       cur = con.cursor()

       cur.execute('SELECT \* FROM fee WHERE  recpt = ? OR name = ? OR admsn = ? OR date = ? OR branch = ? OR sem = ? OR \

                    total = ? OR paid = ? OR due = ?',(recpt,name,admsn,date,branch,sem,total,paid,due))

       row = cur.fetchall()

       return row

       con.commit()

connect()

from tkinter import\*

from tkinter import ttk

import tkinter.messagebox

import datetime

import Fee\_Backend

class Fee():

    def \_\_init\_\_(self, master):

        self.master = master

        self.master.title('Fee Report')

        self.master.geometry('1350x750')

        self.master.config(bg='Navajo white')

        # ==================================================Variables=================================================

        self.recpt = StringVar()

        self.name = StringVar()

        self.admsn = StringVar()

        self.date = StringVar()

        self.branch = StringVar()

        self.sem = StringVar()

        self.total = DoubleVar()

        self.paid = DoubleVar()

        self.due = DoubleVar()

        # ==================================================Functions=================================================

        def Tuple(event):

            try:

                global st

                index = self.list.curselection()[0]

                st = self.list.get(index)

                self.recpt\_entry.delete(0, END)

                self.recpt\_entry.insert(END, st[1])

                self.name\_entry.delete(0, END)

                self.name\_entry.insert(END, st[2])

                self.admsn\_entry.delete(0, END)

                self.admsn\_entry.insert(END, st[3])

                self.Date\_entry.delete(0, END)

                self.Date\_entry.insert(END, st[4])

                self.branch\_entry.delete(0, END)

                self.branch\_entry.insert(END, st[5])

                self.sem\_entry.delete(0, END)

                self.sem\_entry.insert(END, st[6])

                self.total\_entry.delete(0, END)

                self.total\_entry.insert(END, st[7])

                self.paid\_entry.delete(0, END)

                self.paid\_entry.insert(END, st[8])

                self.due\_entry.delete(0, END)

                self.due\_entry.insert(END, st[9])

            except IndexError:

                pass

        def Insert():

            if (len(self.admsn.get()) != 0):

                Fee\_Backend.insert(self.recpt.get(), self.name.get(), self.admsn.get(), self.date.get(),

                                   self.branch.get(), self.sem.get(), self.total.get(), self.paid.get(),

                                   self.due.get())

                self.list.delete(0, END)

                self.list.insert(END, (self.recpt.get(), self.name.get(), self.admsn.get(), self.date.get(),

                                       self.branch.get(), self.sem.get(), self.total.get(), self.paid.get(),

                                       self.due.get()))

        def View():

            self.list.delete(0, END)

            for row in Fee\_Backend.view():

                self.list.insert(END, row, str(' '))

        def Reset():

            self.recpt.set(' ')

            self.name.set(' ')

            self.admsn.set(' ')

            #self.date.set(' ')

            self.branch.set(' ')

            self.sem.set(' ')

            self.paid.set(' ')

            self.due.set(' ')

            self.Display.delete('1.0', END)

            self.list.delete(0, END)

        def Delete():

            Fee\_Backend.delete(st[0])

            Reset()

            View()

        def Receipt():

            self.Display.delete('1.0', END)

            self.Display.insert(END, '\t\tRECEIPT' + '\n\n')

            self.Display.insert(

                END, '\tReceipt No.\t     :' + self.recpt.get() + '\n')

            self.Display.insert(END, '\tStudent Name  :' +

                                self.name.get() + '\n')

            self.Display.insert(END, '\tAdmission No.\t:' +

                                self.admsn.get() + '\n')

            self.Display.insert(

                END, '\tDate\t          :' + self.date.get() + '\n')

            self.Display.insert(

                END, '\tBranch\t          :' + self.branch.get() + '\n')

            self.Display.insert(

                END, '\tSemester \t        :' + self.sem.get() + '\n\n')

            x1 = (self.var\_1.get())

            x2 = (self.paid.get())

            x3 = (x1 - x2)

            self.Display.insert(END, '\tTotal Amount  :' + str(x1) + '\n')

            self.Display.insert(END, '\tPaid Amount   :' + str(x2) + '\n')

            self.Display.insert(END, '\tBalance\t         :' + str(x3) + '\n')

            self.due.set(x3)

        def Search():

            self.list.delete(0, END)

            for row in Fee\_Backend.search(self.recpt.get(), self.name.get(), self.admsn.get(), self.date.get(),

                                          self.branch.get(), self.sem.get(), self.total.get(), self.paid.get(),

                                          self.due.get()):

                self.list.insert(END, row, str(' '))

        def Update():

            Fee\_Backend.delete(st[0])

            Insert()

        def Exit():

            Exit = tkinter.messagebox.askyesno(

                'Attention', 'Confirm, if you want to Exit')

            if Exit > 0:

                root.destroy()

                return

        # ==================================================Frames===================================================

        Main\_Frame = Frame(self.master, bg='Navajo white')

        Main\_Frame.grid()

        Title\_Frame = LabelFrame(

            Main\_Frame, width=1350, height=100, bg='Navajo white', relief='ridge', bd=15)

        Title\_Frame.pack(side=TOP)

        self.lblTitle = Label(Title\_Frame, font=('arial', 40, 'bold'), text='FEE REPORT',

                              bg='navajowhite', padx=13)

        self.lblTitle.grid(padx=400)

        Data\_Frame = Frame(Main\_Frame, width=1350, height=350,

                           bg='Navajo white', relief='ridge', bd=15)

        Data\_Frame.pack(side=TOP, padx=15)

        Frame\_1 = LabelFrame(Data\_Frame, width=850, height=350, bg='Navajo white', relief='ridge', bd=8,

                             text='Informations', font=('arial', 15, 'bold'))

        Frame\_1.pack(side=LEFT, padx=10)

        Frame\_2 = LabelFrame(Data\_Frame, width=495, height=350, bg='Navajo white', relief='ridge', bd=8,

                             text='Fee Receipt', font=('arial', 15, 'bold'))

        Frame\_2.pack(side=RIGHT, padx=10)

        List\_Frame = Frame(Main\_Frame, width=1350, height=150,

                           bg='Navajo white', relief='ridge', bd=15)

        List\_Frame.pack(side=TOP, padx=15)

        Button\_Frame = Frame(Main\_Frame, width=1350, height=80,

                             bg='Navajo white', relief='ridge', bd=15)

        Button\_Frame.pack(side=TOP)

        # ===================================================Labels================================================

        self.recpt\_label = Label(Frame\_1, text='Receipt No. : ', font=(

            'arial', 14, 'bold'), bg='Navajo white')

        self.recpt\_label.grid(row=0, column=0, padx=15, sticky=W)

        self.name\_label = Label(Frame\_1, text='Student Name : ', font=(

            'arial', 14, 'bold'), bg='Navajo white')

        self.name\_label.grid(row=1, column=0, padx=15, sticky=W)

        self.admsn\_label = Label(Frame\_1, text='Admission No. : ', font=(

            'arial', 14, 'bold'), bg='Navajo white')

        self.admsn\_label.grid(row=2, column=0, padx=15, sticky=W)

        self.Date\_label = Label(Frame\_1, text='Date : ', font=(

            'arial', 14, 'bold'), bg='Navajo white')

        self.Date\_label.grid(row=3, column=0, padx=15, sticky=W)

        self.branch\_label = Label(Frame\_1, text='Branch : ', font=(

            'arial', 14, 'bold'), bg='Navajo white')

        self.branch\_label.grid(row=4, column=0, padx=15, sticky=W)

        self.sem\_label = Label(Frame\_1, text='Semester : ', font=(

            'arial', 14, 'bold'), bg='Navajo white')

        self.sem\_label.grid(row=5, column=0, padx=15, sticky=W)

        self.total\_label = Label(Frame\_1, text='TOTAL AMOUNT : ', font=(

            'arial', 14, 'bold'), bg='Navajo white')

        self.total\_label.grid(row=2, column=2, padx=5, sticky=W)

        self.paid\_label = Label(Frame\_1, text='PAID AMOUNT : ', font=(

            'arial', 14, 'bold'), bg='Navajo white')

        self.paid\_label.grid(row=3, column=2, padx=5, sticky=W)

        self.due\_label = Label(Frame\_1, text='BALANCE : ', font=(

            'arial', 14, 'bold'), bg='Navajo white')

        self.due\_label.grid(row=4, column=2, padx=5, sticky=W)

        # ==================================================Entries=================================================

        self.var\_1 = DoubleVar(Frame\_1, value='36265')

        d1 = datetime.date.today()

        self.date.set(d1)

        self.recpt\_entry = Entry(Frame\_1, font=(

            'arial', 14), textvariable=self.recpt)

        self.recpt\_entry.grid(row=0, column=1, padx=15, pady=5)

        self.name\_entry = Entry(Frame\_1, font=(

            'arial', 14), textvariable=self.name)

        self.name\_entry.grid(row=1, column=1, padx=15, pady=5)

        self.admsn\_entry = Entry(Frame\_1, font=(

            'arial', 14), textvariable=self.admsn)

        self.admsn\_entry.grid(row=2, column=1, padx=15, pady=5)

        self.Date\_entry = Entry(Frame\_1, font=(

            'arial', 14), textvariable=self.date)

        self.Date\_entry.grid(row=3, column=1, padx=15, pady=5)

        self.branch\_entry = ttk.Combobox(Frame\_1, values=(' ', 'CSE', 'IT', 'ETC/ET&T', 'Mechanical', 'Civil', 'EE', 'EEE'),

                                         font=('arial', 14), width=19, textvariable=self.branch)

        self.branch\_entry.grid(row=4, column=1, padx=15, pady=5)

        self.sem\_entry = ttk.Combobox(Frame\_1, values=(' ', 'FIRST', 'SECOND', 'THIRD', 'FOURTH', 'FIFTH', 'SIXTH',

                                                       'SEVENTH', 'EIGHTH'), font=('arial', 14), width=19,

                                      textvariable=self.sem)

        self.sem\_entry.grid(row=5, column=1, padx=15, pady=5)

        self.total\_entry = Entry(Frame\_1, font=(

            'arial', 14), width=10, textvariable=self.var\_1, state='readonly')

        self.total\_entry.grid(row=2, column=3, padx=8, pady=5)

        self.paid\_entry = Entry(Frame\_1, font=(

            'arial', 14), width=10, textvariable=self.paid)

        self.paid\_entry.grid(row=3, column=3, pady=5)

        self.due\_entry = Entry(Frame\_1, font=(

            'arial', 14), width=10, textvariable=self.due)

        self.due\_entry.grid(row=4, column=3, pady=7)

        # ==================================================Frame\_2=================================================

        self.Display = Text(Frame\_2, width=42, height=12,

                            font=('arial', 14, 'bold'))

        self.Display.grid(row=0, column=0, padx=3)

        # =============================================List box and scrollbar===========================================

        sb = Scrollbar(List\_Frame)

        sb.grid(row=0, column=1, sticky='ns')

        self.list = Listbox(List\_Frame, font=(

            'arial', 13, 'bold'), width=140, height=8)

        self.list.bind('<<ListboxSelect>>', Tuple)

        self.list.grid(row=0, column=0)

        sb.config(command=self.list.yview)

        # ==================================================Buttons=================================================

        btnSave = Button(Button\_Frame, text='SAVE', font=(

            'arial', 14, 'bold'), width=10, command=Insert)

        btnSave.grid(row=0, column=0, padx=5, pady=5)

        btnDisplay = Button(Button\_Frame, text='DISPLAY', font=(

            'arial', 14, 'bold'), width=10, command=View)

        btnDisplay.grid(row=0, column=1, padx=5, pady=5)

        btnReset = Button(Button\_Frame, text='RESET', font=(

            'arial', 14, 'bold'), width=10, command=Reset)

        btnReset.grid(row=0, column=2, padx=5, pady=5)

        btnReset = Button(Button\_Frame, text='UPDATE', font=(

            'arial', 14, 'bold'), width=10, command=Update)

        btnReset.grid(row=0, column=3, padx=5, pady=5)

        btnSearch = Button(Button\_Frame, text='SEARCH', font=(

            'arial', 14, 'bold'), width=10, command=Search)

        btnSearch.grid(row=0, column=4, padx=5, pady=5)

        btnDelete = Button(Button\_Frame, text='DELETE', font=(

            'arial', 14, 'bold'), width=10, command=Delete)

        btnDelete.grid(row=0, column=5, padx=5, pady=5)

        btnReceipt = Button(Button\_Frame, text='RECEIPT', font=(

            'arial', 14, 'bold'), width=10, command=Receipt)

        btnReceipt.grid(row=0, column=6, padx=5, pady=5)

        btnExit = Button(Button\_Frame, text='EXIT', font=(

            'arial', 14, 'bold'), width=10, command=Exit)

        btnExit.grid(row=0, column=7, padx=5, pady=5)

root = Tk()

obj = Fee(root)

root.mainloop()

import sqlite3

def connect():

       con = sqlite3.connect('library.db')

       cur = con.cursor()

       cur.execute('CREATE TABLE IF NOT EXISTS library(x INTEGER PRIMARY KEY, Mtype text, refno integer, fname text, \

                     surname text, address text, post integer, mobno integer, ID text, title text, author text, \

                     borrow integer, due integer, loan integer)')

       con.commit()

       con.close()

def insert(Mtype = ' ', refno = ' ', fname = ' ', surname = ' ', address = ' ', post = ' ', mobno = ' ', ID = ' ', \

           title = ' ', author = ' ', borrow = ' ', due = ' ', loan = ' '):

       con = sqlite3.connect('library.db')

       cur = con.cursor()

       cur.execute('INSERT INTO library VALUES (NULL,?,?,?,?,?,?,?,?,?,?,?,?,?)',(Mtype,refno,fname,surname,address,post, \

                                                                                  mobno,ID,title,author,borrow,due,loan))

       con.commit()

       con.close()

def view():

       con = sqlite3.connect('library.db')

       cur = con.cursor()

       cur.execute('SELECT \* FROM library')

       row = cur.fetchall()

       return row

       con.close()

def delete(x):

       con = sqlite3.connect('library.db')

       cur = con.cursor()

       cur.execute('DELETE FROM library WHERE x = ?',(x,))

       con.commit()

       con.close()

def update(x, Mtype = ' ', refno = ' ', fname = ' ', surname = ' ', address = ' ', post = ' ', mobno = ' ', ID = ' ', \

           title = ' ', author = ' ', borrow = ' ', due = ' ', loan = ' '):

       con = sqlite3.connect('library.db')

       cur = con.cursor()

       cur.execute('UPDATE library SET Mtype = ? OR refno = ? OR fname = ? OR surname = ? OR address = ? OR post = ? OR \

       mobno = ? OR ID = ? OR title = ? OR author = ? OR borrow = ? OR due = ? OR loan = ?',(Mtype,refno,fname,surname,address, \

                                                                                             post,mobno,ID,title,author,borrow,due,loan))

       con.commit()

       con.close()

def search(Mtype = ' ', refno = ' ', fname = ' ', surname = ' ', address = ' ', post = ' ', mobno = ' ', ID = ' ', \

           title = ' ', author = ' ', borrow = ' ', due = ' ', loan = ' '):

       con = sqlite3.connect('library.db')

       cur = con.cursor()

       cur.execute('SELECT \* FROM library WHERE Mtype = ? OR refno = ? OR fname = ? OR surname = ? OR address = ? OR \

       post = ? OR mobno = ? OR ID = ? OR title = ? OR author = ? OR borrow = ? OR due = ? OR loan = ?',(Mtype,refno,\

                                                                                                         fname,surname,\

                                                                                                         address,post,mobno,\

                                                                                                         ID,title,author,\

                                                                                                         borrow,due,loan))

       row = cur.fetchall()

       return row

       con.close()

connect()

 from tkinter import\*

from tkinter import ttk

import random

from datetime import datetime

import tkinter.messagebox

import Library\_Backend

class Library:

       def \_\_init\_\_(self, root):

              self.root = root

              self.root.title('Library Management System')

              self.root.geometry('1350x750')

              self.root.config(bg = 'navajowhite')

       #===================================================Variables===================================================

              self.Mtype = StringVar()

              self.refno = StringVar()

              self.fname = StringVar()

              self.surname = StringVar()

              self.address = StringVar()

              self.post = StringVar()

              self.mobno = StringVar()

              self.ID = StringVar()

              self.title = StringVar()

              self.author = StringVar()

              self.borrow = StringVar()

              self.due = StringVar()

              self.loan = StringVar()

              self.yop = StringVar()

              self.edsn = StringVar()

       #================================================Functions======================================================

              def BookRec(event):

                     try:

                            global selected\_tuple

                            index = self.Listbox\_2.curselection()[0]

                            selected\_tuple = self.Listbox\_2.get(index)

                            self.Entry\_0.delete(0, END)

                            self.Entry\_0.insert(END, selected\_tuple[1])

                            self.Entry\_1.delete(0, END)

                            self.Entry\_1.insert(END, selected\_tuple[2])

                            self.Entry\_2.delete(0, END)

                            self.Entry\_2.insert(END, selected\_tuple[3])

                            self.Entry\_3.delete(0, END)

                            self.Entry\_3.insert(END, selected\_tuple[4])

                            self.Entry\_4.delete(0, END)

                            self.Entry\_4.insert(END, selected\_tuple[5])

                            self.Entry\_5.delete(0, END)

                            self.Entry\_5.insert(END, selected\_tuple[6])

                            self.Entry\_6.delete(0, END)

                            self.Entry\_6.insert(END, selected\_tuple[7])

                            self.Entry\_7.delete(0, END)

                            self.Entry\_7.insert(END, selected\_tuple[8])

                            self.Entry\_8.delete(0, END)

                            self.Entry\_8.insert(END, selected\_tuple[9])

                            self.Entry\_9.delete(0, END)

                            self.Entry\_9.insert(END, selected\_tuple[10])

                            self.Entry\_10.delete(0, END)

                            self.Entry\_10.insert(END, selected\_tuple[11])

                            self.Entry\_11.delete(0, END)

                            self.Entry\_11.insert(END, selected\_tuple[12])

                            self.Entry\_12.delete(0, END)

                            self.Entry\_12.insert(END, selected\_tuple[13])

                     except IndexError:

                            pass

              def Insert():

                     if(len(self.refno.get()) != 0):

                            Library\_Backend.insert(self.Mtype.get(), self.refno.get(), self.fname.get(), self.surname.get()\

                                                   , self.address.get(), self.post.get(), self.mobno.get(), self.ID.get()\

                                                   , self.title.get(), self.author.get(), self.borrow.get(), self.due.get()\

                                                   , self.loan.get())

                            self.Listbox\_2.delete(0, END)

                            self.Listbox\_2.insert(END , (self.Mtype.get(), self.refno.get(), self.fname.get(), self.surname.get()\

                                                         , self.address.get(), self.post.get(), self.mobno.get(), self.ID.get()\

                                                         , self.title.get(), self.author.get(), self.borrow.get(), self.due.get()\

                                                         , self.loan.get()))

              def Display():

                     self.Listbox\_2.delete(0, END)

                     for row in Library\_Backend.view():

                            self.Listbox\_2.insert(END, row, str(' '))

              def Exit():

                     Exit = tkinter.messagebox.askyesno('Library Management System','Confirm if you want to Exit')

                     if Exit > 0:

                            root.destroy()

                            return

              def Reset():

                     self.Mtype.set('')

                     self.refno.set('')

                     self.fname.set('')

                     self.surname.set('')

                     self.address.set('')

                     self.post.set('')

                     self.mobno.set('')

                     self.ID.set('')

                     self.title.set('')

                     self.author.set('')

                     self.borrow.set('')

                     self.due.set('')

                     self.loan.set('')

                     self.Display.delete('1.0',END)

                     self.Listbox\_2.delete(0, END)

              def Delete():

                     Library\_Backend.delete(selected\_tuple[0])

                     Reset()

                     Display()

              def Update():

                     Library\_Backend.delete(selected\_tuple[0])

                     Library\_Backend.insert(self.Mtype.get(), self.refno.get(), self.fname.get(), self.surname.get()\

                                                  , self.address.get(), self.post.get(), self.mobno.get(), self.ID.get()\

                                                  , self.title.get(), self.author.get(), self.borrow.get(), self.due.get()\

                                                  , self.loan.get())

                     self.Listbox\_2.delete(0, END)

                     self.Listbox\_2.insert(END,(self.Mtype.get(), self.refno.get(), self.fname.get(), self.surname.get()\

                                                  , self.address.get(), self.post.get(), self.mobno.get(), self.ID.get()\

                                                  , self.title.get(), self.author.get(), self.borrow.get(), self.due.get()\

                                                  , self.loan.get()))

              def Search():

                     self.Listbox\_2.delete(0, END)

                     for row in Library\_Backend.search(self.Mtype.get(), self.refno.get(), self.fname.get(), self.surname.get()\

                                                  , self.address.get(), self.post.get(), self.mobno.get(), self.ID.get()\

                                                  , self.title.get(), self.author.get(), self.borrow.get(), self.due.get()\

                                                  , self.loan.get()):

                            self.Listbox\_2.insert(END, row, str(' '))

              def Details():

                     self.Display.delete('1.0',END)

                     self.Display.insert(END, 'Book ID: ' + self.ID.get() + '\n')

                     self.Display.insert(END, 'Title: ' + self.title.get() + '\n')

                     self.Display.insert(END, 'Author:  ' +  self.author.get() + '\n')

                     self.Display.insert(END, 'Edition: ' + self.edsn.get() + '\n')

                     self.Display.insert(END, 'Year of Publision: \t' + self.yop.get() + '\n')

                     self.Display.insert(END, 'Date Borrowed: ' + self.borrow.get() + '\n')

                     self.Display.insert(END, 'Date Due:' + self.due.get() + '\n')

                     self.Display.insert(END, 'Days in Loan: ' + self.loan.get() + '\n')

       #=====================================================Frames=====================================================

              Main\_Frame = Frame(self.root, bg = 'navajowhite')

              Main\_Frame.grid()

              Title\_Frame\_1 = Frame(Main\_Frame, width = 1350, bg = 'navajowhite', relief = RIDGE, bd = 15, padx = 20)

              Title\_Frame\_1.pack(side = TOP)

              self.lblTitle = Label(Title\_Frame\_1, font = ('arial',40,'bold'), text = '\tLibrary Management System\t', \

                                    bg = 'navajowhite', padx = 13)

              self.lblTitle.grid()

              Button\_Frame = Frame(Main\_Frame, width = 1350, height = 50, relief = RIDGE, bd = 10, bg = 'navajowhite')

              Button\_Frame.pack(side = BOTTOM)

              Detail\_Frame = Frame(Main\_Frame, width = 1350, height = 100, relief = RIDGE, bd = 10, bg = 'navajowhite')

              Detail\_Frame.pack(side = BOTTOM)

              Data\_Frame = Frame(Main\_Frame, width = 1350, height = 400, relief = RIDGE, bd = 15, bg = 'navajowhite')

              Data\_Frame.pack(side = BOTTOM)

              Frame\_1 = LabelFrame(Data\_Frame, width = 800, height = 400, relief = RIDGE, bd = 10, bg = 'navajowhite', \

                              text = "Library Membership Info:", padx = 20, font = ('arial',15,'bold'))

              Frame\_1.pack(side = LEFT, padx = 3)

              Frame\_2 = LabelFrame(Data\_Frame, width = 550, height = 400, relief = RIDGE, bd = 10, bg = 'navajowhite', \

                              text = "Book Details:", padx = 20, font = ('arial',15,'bold'))

              Frame\_2.pack(side = RIGHT)

       #================================================Labels========================================================

              self.Label\_1 = Label(Frame\_1, text = 'Member type', font = ('arial',13,'bold'), pady = 2, \

                                   bg = 'navajowhite' )

              self.Label\_1.grid(row = 0, column = 0, sticky = W)

              self.Label\_2 = Label(Frame\_1, text = 'Reference No.', font = ('arial',13,'bold'), pady = 2, \

                                   bg = 'navajowhite' )

              self.Label\_2.grid(row = 1, column = 0, sticky = W)

              self.Label\_3 = Label(Frame\_1, text = 'First Name', font = ('arial',13,'bold'), pady = 2, \

                                   bg = 'navajowhite' )

              self.Label\_3.grid(row = 2, column = 0, sticky = W)

              self.Label\_4 = Label(Frame\_1, text = 'Surname', font = ('arial',13,'bold'), pady = 2, \

                                   bg = 'navajowhite' )

              self.Label\_4.grid(row = 3, column = 0, sticky = W)

              self.Label\_5 = Label(Frame\_1, text = 'Address', font = ('arial',13,'bold'), pady = 2, \

                                   bg = 'navajowhite' )

              self.Label\_5.grid(row = 4, column = 0, sticky = W)

              self.Label\_6 = Label(Frame\_1, text = 'Post Code', font = ('arial',13,'bold'), pady = 2, \

                                   bg = 'navajowhite' )

              self.Label\_6.grid(row = 5, column = 0, sticky = W)

              self.Label\_7 = Label(Frame\_1, text = 'Mobile No.', font = ('arial',13,'bold'), pady = 2, \

                                   bg = 'navajowhite' )

              self.Label\_7.grid(row = 6, column = 0, sticky = W)

              self.Label\_8 = Label(Frame\_1, text = 'Book ID', font = ('arial',13,'bold'), pady = 2, \

                                   bg = 'navajowhite' )

              self.Label\_8.grid(row = 0, column = 2, sticky = W)

              self.Label\_9 = Label(Frame\_1, text = 'Book Title', font = ('arial',13,'bold'), pady = 2, \

                                   bg = 'navajowhite' )

              self.Label\_9.grid(row = 1, column = 2, sticky = W)

              self.Label\_10 = Label(Frame\_1, text = 'Author', font = ('arial',13,'bold'), pady = 2, \

                                   bg = 'navajowhite' )

              self.Label\_10.grid(row = 2, column = 2, sticky = W)

              self.Label\_11 = Label(Frame\_1, text = 'Date Borrowed', font = ('arial',13,'bold'), pady = 2, \

                                   bg = 'navajowhite' )

              self.Label\_11.grid(row = 3, column = 2, sticky = W)

              self.Label\_13 = Label(Frame\_1, text = 'Date Due', font = ('arial',13,'bold'), pady = 2, \

                                   bg = 'navajowhite' )

              self.Label\_13.grid(row = 4, column = 2, sticky = W)

              self.Label\_13 = Label(Frame\_1, text = 'Days in Loan', font = ('arial',13,'bold'), pady = 2, \

                                   bg = 'navajowhite' )

              self.Label\_13.grid(row = 5, column = 2, sticky = W)

       #================================================Entries========================================================

              self.Entry\_0 = ttk.Combobox(Frame\_1, values = (' ','Student','Faculty','Staff Member'), \

                                          font = ('arial',13,'bold'), width = 23, textvariable = self.Mtype )

              self.Entry\_0.grid(row = 0, column = 1)

              self.Entry\_1 = Entry(Frame\_1, font = ('arial',13,'bold'), width = 25, textvariable = self.refno )

              self.Entry\_1.grid(row = 1, column = 1, padx = 15)

              self.Entry\_2 = Entry(Frame\_1, font = ('arial',13,'bold'), width = 25, textvariable = self.fname)

              self.Entry\_2.grid(row = 2, column = 1, padx = 15)

              self.Entry\_3 = Entry(Frame\_1, font = ('arial',13,'bold'), width = 25, textvariable = self.surname)

              self.Entry\_3.grid(row = 3, column = 1, padx = 15)

              self.Entry\_4 = Entry(Frame\_1, font = ('arial',13,'bold'), width = 25, textvariable = self.address)

              self.Entry\_4.grid(row = 4, column = 1, padx = 15)

              self.Entry\_5 = Entry(Frame\_1, font = ('arial',13,'bold'), width = 25, textvariable = self.post)

              self.Entry\_5.grid(row = 5, column = 1, padx = 15)

              self.Entry\_6 = Entry(Frame\_1, font = ('arial',13,'bold'), width = 25, textvariable = self.mobno)

              self.Entry\_6.grid(row = 6, column = 1, padx = 15)

              self.Entry\_7 = Entry(Frame\_1, font = ('arial',13,'bold'), width = 25, textvariable = self.ID)

              self.Entry\_7.grid(row = 0, column = 4, padx = 15)

              self.Entry\_8 = Entry(Frame\_1, font = ('arial',13,'bold'), width = 25, textvariable = self.title)

              self.Entry\_8.grid(row = 1, column = 4, padx = 15)

              self.Entry\_9 = Entry(Frame\_1, font = ('arial',13,'bold'), width = 25, textvariable = self.author)

              self.Entry\_9.grid(row = 2, column = 4, padx = 15)

              self.Entry\_10 = Entry(Frame\_1, font = ('arial',13,'bold'), width = 25, textvariable = self.borrow)

              self.Entry\_10.grid(row = 3, column = 4, padx = 15)

              self.Entry\_11 = Entry(Frame\_1, font = ('arial',13,'bold'), width = 25, textvariable = self.due)

              self.Entry\_11.grid(row = 4, column = 4, padx = 15)

              self.Entry\_12 = Entry(Frame\_1, font = ('arial',13,'bold'), width = 25, textvariable = self.loan)

              self.Entry\_12.grid(row = 5, column = 4, padx = 15)

       #=============================================Widgets=========================================================

              self.Display = Text(Frame\_2, font = ('arial',13,'bold'), width = 28, height = 11)

              self.Display.grid(row = 0, column = 2)

              List\_of\_Books = [' C',' C++',' Java',' Python',' PHP',' Java Script',' My SQL',' Data Structure',' Linux',\

                               ' Operating System',' Web Developement',' Data Science',' Algorithms',' Android', \

                               ' VB.net']

       #===========================================Function for Books Details=========================================

              def SelectedBook(event):

                     value = str(self.Listbox\_1.get(self.Listbox\_1.curselection()))

                     v = value

                     if (v == ' C'):

                            self.ID.set('ISBN 525341')

                            self.title.set('Programming using C')

                            self.author.set('Yashwant Kanetkar')

                            self.yop.set('2019')

                            self.edsn.set('5th')

                            import datetime

                            d1 = datetime.date.today()

                            d2 = datetime.timedelta(days = 14)

                            d3 = (d1 + d2)

                            self.borrow.set(d1)

                            self.loan.set('14')

                            self.due.set(d3)

                            Details()

                     elif (v == ' C++'):

                            self.ID.set('ISBN 345687')

                            self.title.set('Programming using C++')

                            self.author.set('Yashwant Kanetkar')

                            self.yop.set('2019')

                            self.edsn.set('4th')

                            import datetime

                            d1 = datetime.date.today()

                            d2 = datetime.timedelta(days = 10)

                            d3 = (d1 + d2)

                            self.borrow.set(d1)

                            self.loan.set('10')

                            self.due.set(d3)

                            Details()

                     elif (v == ' Java'):

                            self.ID.set('ISBN 643842')

                            self.title.set('Java Programming')

                            self.author.set('Joshua Bloch')

                            self.yop.set('2019')

                            self.edsn.set('7th')

                            import datetime

                            d1 = datetime.date.today()

                            d2 = datetime.timedelta(days = 13)

                            d3 = (d1 + d2)

                            self.borrow.set(d1)

                            self.loan.set('13')

                            self.due.set(d3)

                            Details()

                     elif (v == ' Python'):

                            self.ID.set('ISBN 564524')

                            self.title.set('Python Programming')

                            self.author.set('John Zelle')

                            self.yop.set('2019')

                            self.edsn.set('3rd')

                            import datetime

                            d1 = datetime.date.today()

                            d2 = datetime.timedelta(days = 13)

                            d3 = (d1 + d2)

                            self.borrow.set(d1)

                            self.loan.set('13')

                            self.due.set(d3)

                            Details()

                     elif (v == ' PHP'):

                            self.ID.set('ISBN 735893')

                            self.title.set('PHP Programming')

                            self.author.set('Alan Forbes')

                            self.yop.set('2019')

                            self.edsn.set('5th')

                            import datetime

                            d1 = datetime.date.today()

                            d2 = datetime.timedelta(days = 15)

                            d3 = (d1 + d2)

                            self.borrow.set(d1)

                            self.loan.set('15')

                            self.due.set(d3)

                            Details()

                     elif (v == ' Java Script'):

                            self.ID.set('ISBN 643842')

                            self.title.set('Java Script Programming')

                            self.author.set('Jon Duckett.')

                            self.yop.set('2019')

                            self.edsn.set('4th')

                            import datetime

                            d1 = datetime.date.today()

                            d2 = datetime.timedelta(days = 13)

                            d3 = (d1 + d2)

                            self.borrow.set(d1)

                            self.loan.set('13')

                            self.due.set(d3)

                            Details()

                     elif (v == ' My SQL'):

                            self.ID.set('ISBN 649635')

                            self.title.set('My SQL Programming')

                            self.author.set('Groff James')

                            self.yop.set('2019')

                            self.edsn.set('3rd')

                            import datetime

                            d1 = datetime.date.today()

                            d2 = datetime.timedelta(days = 20)

                            d3 = (d1 + d2)

                            self.borrow.set(d1)

                            self.loan.set('20')

                            self.due.set(d3)

                            Details()

                     elif (v == ' Data Structure'):

                            self.ID.set('ISBN 531588')

                            self.title.set('Data Structure')

                            self.author.set('Karumanchi Narasimha')

                            self.yop.set('2019')

                            self.edsn.set('5th')

                            import datetime

                            d1 = datetime.date.today()

                            d2 = datetime.timedelta(days = 11)

                            d3 = (d1 + d2)

                            self.borrow.set(d1)

                            self.loan.set('11')

                            self.due.set(d3)

                            Details()

                     elif (v == ' Linux'):

                            self.ID.set('ISBN 356853')

                            self.title.set('Linux Administration')

                            self.author.set('SOYINKA')

                            self.yop.set('2019')

                            self.edsn.set('1st')

                            import datetime

                            d1 = datetime.date.today()

                            d2 = datetime.timedelta(days = 6)

                            d3 = (d1 + d2)

                            self.borrow.set(d1)

                            self.loan.set('6')

                            self.due.set(d3)

                            Details()

                     elif (v == ' Operating System'):

                            self.ID.set('ISBN 536453')

                            self.title.set('OS Concepts ')

                            self.author.set('Silberschatz Abraham')

                            self.yop.set('2019')

                            self.edsn.set('4th')

                            import datetime

                            d1 = datetime.date.today()

                            d2 = datetime.timedelta(days = 12)

                            d3 = (d1 + d2)

                            self.borrow.set(d1)

                            self.loan.set('12')

                            self.due.set(d3)

                            Details()

                     elif (v == ' Web Developement'):

                            self.ID.set('ISBN 543548')

                            self.title.set('Web Developement ')

                            self.author.set('Paul McFedries')

                            self.yop.set('2019')

                            self.edsn.set('3rd')

                            import datetime

                            d1 = datetime.date.today()

                            d2 = datetime.timedelta(days = 15)

                            d3 = (d1 + d2)

                            self.borrow.set(d1)

                            self.loan.set('15')

                            self.due.set(d3)

                            Details()

                     elif (v == ' Data Science'):

                            self.ID.set('ISBN 835764')

                            self.title.set('Data Science Concept ')

                            self.author.set('David Stephenson')

                            self.yop.set('2019')

                            self.edsn.set('3rd')

                            import datetime

                            d1 = datetime.date.today()

                            d2 = datetime.timedelta(days = 15)

                            d3 = (d1 + d2)

                            self.borrow.set(d1)

                            self.loan.set('15')

                            self.due.set(d3)

                            Details()

                     elif (v == ' Algorithms'):

                            self.ID.set('ISBN 535674')

                            self.title.set('Basics of Algorithm ')

                            self.author.set('Karumanchi Narasimha')

                            self.yop.set('2019')

                            self.edsn.set('7th')

                            import datetime

                            d1 = datetime.date.today()

                            d2 = datetime.timedelta(days = 10)

                            d3 = (d1 + d2)

                            self.borrow.set(d1)

                            self.loan.set('10')

                            self.due.set(d3)

                            Details()

                     elif (v == ' Android'):

                            self.ID.set('ISBN 356452')

                            self.title.set('Android Programming')

                            self.author.set('Harwani B. M')

                            self.yop.set('2019')

                            self.edsn.set('4th')

                            import datetime

                            d1 = datetime.date.today()

                            d2 = datetime.timedelta(days = 9)

                            d3 = (d1 + d2)

                            self.borrow.set(d1)

                            self.loan.set('9')

                            self.due.set(d3)

                            Details()

       #===========================================List Box and Scroll Bar==========================================

              sb\_1 = Scrollbar(Frame\_2)

              sb\_1.grid(row =0, column = 1, sticky = 'ns')

              self.Listbox\_1 = Listbox(Frame\_2, font = ('arial',13,'bold'), width = 20, height = 10)

              self.Listbox\_1.bind('<<ListboxSelect>>', SelectedBook)

              self.Listbox\_1.grid(row = 0, column = 0)

              sb\_1.config(command = self.Listbox\_1.yview)

              sb\_2 = Scrollbar(Detail\_Frame)

              sb\_2.grid(row = 1, column = 1, sticky = 'ns')

              self.Listbox\_2 = Listbox(Detail\_Frame, font = ('arial',13,'bold'), width = 144, height = 11)

              self.Listbox\_2.bind('<<ListboxSelect>>', BookRec)

              self.Listbox\_2.grid(row = 1, column = 0)

              sb\_2.config(command = self.Listbox\_2.yview)

              for items in List\_of\_Books:

                     self.Listbox\_1.insert(END, items)

       #=============================================Buttons=========================================================

              Button\_1 = Button(Button\_Frame, text = 'SAVE', font = ('arial',15,'bold'), width = 10, command = Insert)

              Button\_1.grid(row = 0, column = 0, padx = 8, pady = 5)

              Button\_2 = Button(Button\_Frame, text = 'DISPLAY', font = ('arial',15,'bold'), width = 10, command = Display)

              Button\_2.grid(row = 0, column = 1, padx = 8)

              Button\_3 = Button(Button\_Frame, text = 'RESET', font = ('arial',15,'bold'), width = 10, command = Reset)

              Button\_3.grid(row = 0, column = 2, padx = 8)

              Button\_4 = Button(Button\_Frame, text = 'UPDATE', font = ('arial',15,'bold'), width = 10, command = Update)

              Button\_4.grid(row = 0, column = 3, padx = 8)

              Button\_5 = Button(Button\_Frame, text = 'SEARCH', font = ('arial',15,'bold'), width = 10, command = Search)

              Button\_5.grid(row = 0, column = 4, padx = 8)

              Button\_6 = Button(Button\_Frame, text = 'DELETE', font = ('arial',15,'bold'), width = 10, command = Delete)

              Button\_6.grid(row = 0, column = 5, padx = 8)

              Button\_7 = Button(Button\_Frame, text = 'EXIT', font = ('arial',15,'bold'), width = 10, command = Exit)

              Button\_7.grid(row = 0, column = 6, padx = 8)

if \_\_name\_\_ == '\_\_main\_\_':

       root = Tk()

       applicaton = Library(root)

       root.mainloop()

from tkinter import\*

import tkinter.messagebox                               # for messagebox

import os                                               # for stringvariable

from tkinter import ttk                                 # for combobox

import random                                           # for reference

import time

import datetime

def main():

    root = Tk()

    app = Window\_1(root)

class Window\_1:

    def \_\_init\_\_(self, master):

        self.master = master

        self.master.title("Login Window")

        self.master.geometry('1350x750')

        self.master.config(bg = 'lightskyblue')

        self.Frame = Frame(self.master, bg = 'lightskyblue')

        self.Frame.pack()

        self.Username = StringVar()                             # x = StringVar()  Holds a string; default value is " "

        self.Password = StringVar()

        self.Lbl\_Title = Label(self.Frame, text = 'Login Menu', font = ('arial',55,'bold'), bg = 'lightskyblue', fg = 'Black')

        self.Lbl\_Title.grid(row = 0, column = 0, columnspan =3, pady = 40)

        self.Login\_Frame\_1 = LabelFrame(self.Frame, width = 1350, height = 600, relief = 'ridge', bg = 'lightskyblue', bd = 15,

                                        font = ('arial',20,'bold'))

        self.Login\_Frame\_1.grid(row = 1, column =0)

        self.Login\_Frame\_2 = LabelFrame(self.Frame, width = 1000, height = 600, relief = 'ridge',bg = 'lightskyblue', bd = 15,

                                        font = ('arial',20,'bold'))

        self.Login\_Frame\_2.grid(row = 2, column = 0)

        #===================================================LABEL and ENTRIES=======================================================================

        self.Label\_Username = Label(self.Login\_Frame\_1, text = 'Username', font = ('arial',20,'bold'), bg = 'lightskyblue', bd = 20)

        self.Label\_Username.grid(row = 0, column = 0)

        self.text\_Username = Entry(self.Login\_Frame\_1, font = ('arial',20,'bold'), textvariable = self.Username)

        self.text\_Username.grid(row = 0, column = 1, padx = 50)

        self.Label\_Password = Label(self.Login\_Frame\_1, text = 'Password', font = ('arial',20,'bold'), bg = 'lightskyblue', bd = 20)

        self.Label\_Password.grid(row = 1, column = 0)

        self.text\_Password = Entry(self.Login\_Frame\_1, font = ('arial',20,'bold'), show = '\*', textvariable = self.Password)

        self.text\_Password.grid(row = 1, column = 1)

        #=============================================================BUTTONS=======================================================================

        self.btnLogin = Button(self.Login\_Frame\_2, text = 'Login', width = 10, font = ('airia',15,'bold'), command = self.Login)

        self.btnLogin.grid(row = 3, column = 0, padx = 8, pady = 20)

        self.btnReset = Button(self.Login\_Frame\_2, text = 'Reset', width = 10, font = ('airia',15,'bold'), command = self.Reset)

        self.btnReset.grid(row = 3, column = 1, padx = 8, pady = 20)

        self.btnExit = Button(self.Login\_Frame\_2, text = 'Exit', width = 10, font = ('airia',15,'bold'), command = self.Exit)

        self.btnExit.grid(row = 3, column = 2, padx = 8, pady = 20)

        #======================================================Code for Login Button==================================================================

    def Login(self):

        u = (self.Username.get())

        p = (self.Password.get())

        if (u == str('Prashant') and p == str(12345678)):

            self.\_\_menu\_\_()

        else:

            tkinter.messagebox.askyesno("Login","Error : Wrong Password")

            self.Username.set("")

            self.Password.set("")

            #self.text\_Username.focus()

        #======================================================Code for Reset Button==================================================================

    def Reset(self):

         self.Username.set("")

         self.Password.set("")

         self.text\_Username.focus()

        #======================================================Code for Exit Button==================================================================

    def Exit(self):

        self.Exit = tkinter.messagebox.askokcancel("Login System", "Confirm if you want to Exit")

        if self.Exit > 0:

            self.master.destroy()

            return

    def \_\_menu\_\_(self):

        filename = 'Menu.py'

        os.system(filename)

        os.system('notepad'+filename)

    '''def new\_window(self):

        self.new\_Window = Toplevel(self.master)

        self.app = Window\_2(self.new\_Window)'''

class Window\_2:

    def \_\_init\_\_(self, master):

        self.master = master

        self.master.title("Login Main Menu")

        self.master.geometry('1350x750')

        self.master.config(bg = 'sky blue')

        self.Frame = Frame(self.master, bg = 'lightskyblue')

        self.Frame.pack()

if \_\_name\_\_ == '\_\_main\_\_':                                    # https://micropyramid.com/blog/understand-self-and-\_\_init\_\_-method-in-python-class/

    main()

from tkinter import\*

import sqlite3

def connect():

       con = sqlite3.connect('Marks.db')

       cur = con.cursor()

       cur.execute('CREATE TABLE IF NOT EXISTS Marks (id INTEGER PRIMARY KEY, name text, roll integer, fname text, mname \

                     text, DOB integer, gender text, scl text, email text, m1 integer, m2 integer, m3 integer, m4 integer, \

                     m5 integer, gt integer, per integer, cgpa integer, grade text, div text, result text)')

       con.commit()

       con.close()

def insert(name = ' ',roll = ' ',fname = ' ',mname = ' ',DOB = ' ',gender = ' ',scl = ' ',email = ' ',m1 = ' ',m2 = ' ', \

           m3 = ' ',m4 = ' ',m5 = ' ',gt = ' ',per = ' ',cgpa = ' ',grade = ' ', div = ' ', result = ' '):

       con = sqlite3.connect('Marks.db')

       cur = con.cursor()

       cur.execute('INSERT INTO Marks VALUES (NULL,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)',(name,roll,fname,mname,DOB,gender, \

                                                                                        scl,email,m1,m2,m3,m4,m5,gt,per,\

                                                                                        cgpa,grade,div,result))

       con.commit()

       con.close()

'''def view():

       con = sqlite3.connect('Marks.db')

       cur = con.cursor()

       cur.execute('SELECT \* FROM Marks')

       con.commit()

       con.close()

def delete(id):

       con = sqlite3.connect('Marks.db')

       cur = con.cursor()

       cur.execute('DELETE FROM Marks WHERE id = ?)',(id,))

       con.commit()

       con.close()'''

def update(id,name = ' ',roll = ' ',fname = ' ',mname = ' ',DOB = ' ',gender = ' ',scl = ' ',email = ' ',m1 = ' ',m2 = ' ', \

           m3 = ' ',m4 = ' ',m5 = ' ',gt = ' ',per = ' ',cgpa = ' ',grade = ' ', div = ' ', result = ' '):

       con = sqlite3.connect('Marks.db')

       cur = con.cursor()

       cur.execute('UPDATE Marks SET name = ? OR roll = ? OR fname =  ? OR mname = ? OR DOB = ? OR gender = ? OR \

                     scl = ? OR email = ? OR m1 = ? OR m2 = ? OR  m3 = ? OR m4 = ? OR m5 = ? OR gt = ? OR per = ? OR \

                     cgpa = ? OR grade = ? OR div = ? OR result = ?',(name,roll,fname,mname,DOB,gender,scl,email,m1,m2,m3, \

                                                                      m4,m5,gt,per,cgpa,grade))

       con.commit()

       con.close()

def search(roll):

       con = sqlite3.connect('Marks.db')

       cur = con.cursor()

       cur.execute('SELECT \* FROM Marks WHERE roll = ?',(roll,))

       row = cur.fetchall()

       return row

connect()

from tkinter import \*

import random

import Marksheet\_Backend

import tkinter.messagebox

from tkinter import ttk

def marksheet():

       root = Tk()

       root.title('Marksheet')

       root.geometry('1350x750')

       root.config(bg = 'Navajo white')

       #================================================Variables======================================================

       name = StringVar()

       roll = StringVar()

       fname = StringVar()

       mname = StringVar()

       DOB = StringVar()

       gender = StringVar()

       scl = StringVar()

       email = StringVar()

       m1 = DoubleVar()

       m2 = DoubleVar()

       m3 = DoubleVar()

       m4 = DoubleVar()

       m5 = DoubleVar()

       gt = DoubleVar()

       per = DoubleVar()

       cgpa = DoubleVar()

       grade = StringVar()

       div = StringVar()

       result = StringVar()

       #==============================================Functions==========================================================

       def Add():

              if (len(roll.get()) != 0):

                     Marksheet\_Backend.insert(name.get(),roll.get(),fname.get(),mname.get(),DOB.get(),gender.get(), \

                                              scl.get(),email.get(),m1.get(),m2.get(),m3.get(),m4.get(),m5.get(), \

                                              gt.get(),per.get(),cgpa.get(),grade.get(),div.get(),result.get())

       def Update():

              if (len(roll.get()) != 0):

                     Marksheet\_Backend.update(name.get(),roll.get(),fname.get(),mname.get(),DOB.get(),gender.get(), \

                                              scl.get(),email.get(),m1.get(),m2.get(),m3.get(),m4.get(),m5.get(), \

                                              gt.get(),per.get(),cgpa.get(),grade.get(),div.get(),result.get())

       def Exit():

              Exit = tkinter.messagebox.askyesno('Marksheet','Confirm if you want to Exit')

              if Exit > 0:

                     root.destroy()

                     return

       def Compute():

              x1 = (m1.get());      x2 = (m2.get());    x3 = (m3.get());      x4 = (m4.get());    x5 = (m5.get())

              if x1 > 100:

                     tkinter.messagebox.askokcancel('Attention','Please enter Correct Marks')

                     return

              if x2 > 100:

                     tkinter.messagebox.askokcancel('Attention','Please enter Correct Marks')

                     return

              if x3 > 100:

                     tkinter.messagebox.askokcancel('Attention','Please enter Correct Marks')

                     return

              if x4 > 100:

                     tkinter.messagebox.askokcancel('Attention','Please enter Correct Marks')

                     return

              if x5 > 100:

                     tkinter.messagebox.askokcancel('Attention','Please enter Correct Marks')

                     return

              tot = x1+x2+x3+x4+x5

              gt.set(tot)

              Per = ((x1+x2+x3+x4+x5) \* 100)/500

              per.set(Per)

              cg = (((x1+x2+x3+x4+x5) \* 100)/500) / 9.5

              cgpa.set(round(cg,1))

              if cg > 10:

                     cgpa.set(10)

              if (((x1+x2+x3+x4+x5) \* 100)/500) <= 40:

                     grd = 'G'

              elif (((x1+x2+x3+x4+x5) \* 100)/500) <= 50:

                     grd = 'F'

              elif (((x1+x2+x3+x4+x5) \* 100)/500) <= 60:

                     grd = 'E'

              elif (((x1+x2+x3+x4+x5) \* 100)/500) <= 70:

                     grd = 'D'

              elif (((x1+x2+x3+x4+x5) \* 100)/500) <= 80:

                     grd = 'C'

              elif (((x1+x2+x3+x4+x5) \* 100)/500) <= 90:

                     grd = 'B'

              else:

                     grd = 'A'

              grade.set(grd)

              count = 0

              if x1 < 33:

                     count = count + 1

              if x2 < 33:

                     count = count + 1

              if x3 < 33:

                     count = count + 1

              if x4 < 33:

                     count = count + 1

              if x5 < 33:

                     count = count + 1

              if (count == 0):

                     result.set('PASS')

              elif (count == 1 or count == 2 ):

                     result.set('SUPPLY')

              else:

                     result.set('FAIL')

              if Per <= 45 and result != "FAIL":

                     div.set('THIRD')

              elif Per <= 60 and result != "FAIL":

                     div.set('SECOND')

              elif Per <= 100:

                     div.set('FIRST')

       def Reset():

              name.set(' ')

              roll.set(' ')

              fname.set(' ')

              mname.set(' ')

              DOB.set(' ')

              gender.set(' ')

              scl.set(' ')

              email.set(' ')

              m1.set(' ')

              m2.set(' ')

              m3.set(' ')

              m4.set(' ')

              m5.set(' ')

              gt.set(' ')

              per.set(' ')

              cgpa.set(' ')

              grade.set(' ')

              div.set(' ')

              result.set(' ')

       #========================================================Frame\_1===============================================================

       Frame\_1 = LabelFrame(root, width = 1200, height = 400, font = ('arial',20,'bold'), bg = 'Navajo white', bd = 10, \

                            text = 'Student Details', relief = 'ridge')

       Frame\_1.grid(row = 1, column = 0, pady = 20, padx = 20)

       #=================================================Labels and Entries for Frame\_1===============================================

       Label\_Name = Label(Frame\_1, text = 'Name', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_Name.grid(row = 0, column = 0, padx = 80)

       Entry\_Name = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = name)

       Entry\_Name.grid(row = 0, column = 1, padx = 5, pady = 5)

       Label\_Roll\_no = Label(Frame\_1, text = 'Roll Number', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_Roll\_no.grid(row = 0, column = 3, padx = 80)

       Entry\_Roll\_no = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = roll)

       Entry\_Roll\_no.grid(row = 0, column = 4, padx = 40)

       Label\_Father\_Name = Label(Frame\_1, text = 'Father Name', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_Father\_Name.grid(row = 1, column = 0, padx = 80)

       Entry\_Father\_Name = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = fname)

       Entry\_Father\_Name.grid(row = 1, column = 1, padx = 5, pady = 10)

       Label\_Mother\_Name = Label(Frame\_1, text = 'Mother Name', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_Mother\_Name.grid(row = 1, column = 3, padx = 80)

       Entry\_Mother\_Name = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = mname)

       Entry\_Mother\_Name.grid(row = 1, column = 4, padx = 5)

       Label\_DOB = Label(Frame\_1, text = 'Date of Birth', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_DOB.grid(row = 2, column = 0, padx = 80)

       Entry\_DOB = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = DOB)

       Entry\_DOB.grid(row = 2, column = 1, padx = 5, pady = 5)

       Label\_Gender = Label(Frame\_1, text = 'Gender', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_Gender.grid(row = 2, column = 3, padx = 80)

       Entry\_Gender = ttk.Combobox(Frame\_1, values = (' ','Male','Female','Others'), font = ('arial',15), width = 23, textvariable = gender)

       Entry\_Gender.grid(row = 2, column = 4, padx = 5, pady = 5)

       Label\_School = Label(Frame\_1, text = 'School Name', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_School.grid(row = 3, column = 0, padx = 80)

       Entry\_School = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = scl)

       Entry\_School.grid(row = 3, column = 1, padx = 5, pady = 5)

       Label\_Email = Label(Frame\_1, text = 'Email ID', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_Email.grid(row = 3, column = 3, padx = 80)

       Entry\_Email = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = email)

       Entry\_Email.grid(row = 3, column = 4, padx = 5, pady = 5)

       #========================================================Frame\_2==================================================================

       Frame\_2 = LabelFrame(root, width = 1200, height = 400, font = ('arial',20,'bold'), bg = 'Navajo white', bd = 10 \

                            , text = 'Grades Point Obtained', relief = 'ridge')

       Frame\_2.grid(row = 3, column = 0)

       #======================================================Labels of Frame\_2===========================================================

       Label\_Subject = Label(Frame\_2, text = 'SUBJECT', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_Subject.grid(row = 3, column = 0, padx = 50, pady = 10)

       Label\_obt\_Marks = Label(Frame\_2, text = 'MARKS OBTAINED', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_obt\_Marks.grid(row = 3, column = 1, padx = 20)

       Label\_Subject = Label(Frame\_2, text = 'PASSING MARKS', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_Subject.grid(row = 3, column = 2, padx = 20)

       Label\_obt\_Marks = Label(Frame\_2, text = 'TOTAL MARKS', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_obt\_Marks.grid(row = 3, column = 3, padx = 20)

       Label\_1 = Label(Frame\_2, text = 'MATHEMATICS', font = ('arial',14), bg = 'Navajo white')

       Label\_1.grid(row = 4, column = 0)

       Label\_2 = Label(Frame\_2, text = 'PHYSICS', font = ('arial',14), bg = 'Navajo white')

       Label\_2.grid(row = 5, column = 0)

       Label\_3 = Label(Frame\_2, text = 'CHEMISTRY', font = ('arial',14), bg = 'Navajo white')

       Label\_3.grid(row = 6, column = 0)

       Label\_4 = Label(Frame\_2, text = 'HINDI', font = ('arial',14), bg = 'Navajo white')

       Label\_4.grid(row = 7, column = 0)

       Label\_5 = Label(Frame\_2, text = 'ENGLISH', font = ('arial',14), bg = 'Navajo white')

       Label\_5.grid(row = 8, column = 0)

       Label\_6 = Label(Frame\_2, text = 'GRAND TOTAL', font = ('arial',16), bg = 'Navajo white')

       Label\_6.grid(row = 9, column = 0)

       Label\_7 = Label(Frame\_2, text = 'PERCENTAGE', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_7.grid(row = 10, column = 0)

       Label\_8 = Label(Frame\_2, text = 'CGPA', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_8.grid(row = 10, column = 2)

       Label\_9 = Label(Frame\_2, text = 'GRADE', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_9.grid(row = 10, column = 4)

       Label\_10 = Label(Frame\_2, text = 'DIVISION', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_10.grid(row = 11, column = 0)

       Label\_10 = Label(Frame\_2, text = 'RESULT', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_10.grid(row = 11, column = 2)

       #======================================================Entries of Frame\_2===========================================================

       var\_1 = StringVar(Frame\_2, value = '33')

       var\_2 = StringVar(Frame\_2, value = '100')

       var\_3 = StringVar(Frame\_2, value = '500')

       Entry\_\_1 = Entry(Frame\_2, font = ('arial',16), width = 5, textvariable = m1)

       Entry\_\_1.grid(row = 4, column = 1)

       Entry\_\_2 = Entry(Frame\_2, font = ('arial',16), width = 5, textvariable = m2)

       Entry\_\_2.grid(row = 5, column = 1)

       Entry\_\_3 = Entry(Frame\_2, font = ('arial',16), width = 5, textvariable = m3)

       Entry\_\_3.grid(row = 6, column = 1)

       Entry\_\_4 = Entry(Frame\_2, font = ('arial',16), width = 5, textvariable = m4)

       Entry\_\_4.grid(row = 7, column = 1)

       Entry\_\_5 = Entry(Frame\_2, font = ('arial',16), width = 5, textvariable = m5)

       Entry\_\_5.grid(row = 8, column = 1)

       Entry\_\_6 = Entry(Frame\_2, font = ('arial',14), width = 5, textvariable = gt, state = 'readonly')

       Entry\_\_6.grid(row = 9, column = 1, pady = 8)

       Entry\_\_7 = Entry(Frame\_2, font = ('arial',14,'bold'), width = 5, textvariable = per, state = 'readonly')

       Entry\_\_7.grid(row = 10, column = 1, pady = 8)

       Entry\_\_8 = Entry(Frame\_2, font = ('arial',14,'bold'), width = 5, textvariable = cgpa, state = 'readonly')

       Entry\_\_8.grid(row = 10, column = 3, pady = 8)

       Entry\_\_9 = Entry(Frame\_2, font = ('arial',14,'bold'), width = 5, textvariable = grade, state = 'readonly')

       Entry\_\_9.grid(row = 10, column = 5, padx = 20, pady = 8)

       Entry\_\_10 = Entry(Frame\_2, font = ('arial',14,'bold'), width = 8, textvariable = div, state = 'readonly')

       Entry\_\_10.grid(row = 11, column = 1, padx = 20, pady = 8)

       Entry\_\_11 = Entry(Frame\_2, font = ('arial',14,'bold'), width = 7, textvariable = result, state = 'readonly')

       Entry\_\_11.grid(row = 11, column = 3, padx = 20, pady = 8)

       Entry\_1\_2 = Entry(Frame\_2, textvariable = var\_1, font = ('arial',16), width = 5, state = 'readonly')

       Entry\_1\_2.grid(row = 4, column = 2, pady = 5)

       Entry\_1\_3 = Entry(Frame\_2, textvariable = var\_2, font = ('arial',16), width = 5, state = 'readonly')

       Entry\_1\_3.grid(row = 4, column = 3)

       Entry\_2\_2 = Entry(Frame\_2, textvariable = var\_1, font = ('arial',16), width = 5, state = 'readonly')

       Entry\_2\_2.grid(row = 5, column = 2, pady = 5)

       Entry\_2\_3 = Entry(Frame\_2, textvariable = var\_2, font = ('arial',16), width = 5, state = 'readonly')

       Entry\_2\_3.grid(row = 5, column = 3)

       Entry\_3\_2 = Entry(Frame\_2, textvariable = var\_1, font = ('arial',16), width = 5, state = 'readonly')

       Entry\_3\_2.grid(row = 6, column = 2, pady = 5)

       Entry\_3\_3 = Entry(Frame\_2, textvariable = var\_2, font = ('arial',16), width = 5, state = 'readonly')

       Entry\_3\_3.grid(row = 6, column = 3)

       Entry\_4\_2 = Entry(Frame\_2, textvariable = var\_1, font = ('arial',16), width = 5, state = 'readonly')

       Entry\_4\_2.grid(row = 7, column = 2, pady = 5)

       Entry\_4\_3 = Entry(Frame\_2, textvariable = var\_2, font = ('arial',16), width = 5, state = 'readonly')

       Entry\_4\_3.grid(row = 7, column = 3)

       Entry\_5\_2 = Entry(Frame\_2, textvariable = var\_1, font = ('arial',16), width = 5, state = 'readonly')

       Entry\_5\_2.grid(row = 8, column = 2, pady = 5)

       Entry\_5\_3 = Entry(Frame\_2, textvariable = var\_2, font = ('arial',16), width = 5, state = 'readonly')

       Entry\_5\_3.grid(row = 8, column = 3)

       Entry\_6\_3 = Entry(Frame\_2, textvariable = var\_3, font = ('arial',16), width = 5, state = 'readonly')

       Entry\_6\_3.grid(row = 9, column = 3)

       #======================================================Buttons===========================================================

       Btn\_Compute = Button(Frame\_2, text = 'COMPUTE', font = ('arial',12,'bold'), width = 10, command = Compute)

       Btn\_Compute.grid(row = 4, column = 4, padx = 50, pady = 6)

       Btn\_Save = Button(Frame\_2, text = 'SAVE', font = ('arial',12,'bold'), width = 10, command = Add)

       Btn\_Save.grid(row = 5, column = 4, padx = 50, pady = 6)

       Btn\_Update = Button(Frame\_2, text = 'UPDATE', font = ('arial',12,'bold'), width = 10, command = Update)

       Btn\_Update.grid(row = 6, column = 4, padx = 50, pady = 6)

       Btn\_Cancel = Button(Frame\_2, text = 'RESET', font = ('arial',12,'bold'), width = 10, command = Reset)

       Btn\_Cancel.grid(row = 7, column = 4, padx = 50, pady = 6)

       Btn\_Exit = Button(Frame\_2, text = 'EXIT', font = ('arial',12,'bold'), width = 10, command = Exit)

       Btn\_Exit.grid(row = 8, column = 4, padx = 50, pady = 6)

       root.mainloop()

def search\_result\_marksheet(row):

       root = Tk()

       root.title('Marksheet')

       root.geometry('1350x750')

       root.config(bg = 'Navajo white')

       #==============================================Functions==========================================================

       def Compute():

              x1 = (m1.get());      x2 = (m2.get());    x3 = (m3.get());      x4 = (m4.get());    x5 = (m5.get())

              tot = x1+x2+x3+x4+x5

              gt.set(tot)

              Per = ((x1+x2+x3+x4+x5) \* 100)/500

              per.set(Per)

              cg = (((x1+x2+x3+x4+x5) \* 100)/500) / 9.5

              cgpa.set(round(cg,1))

              if (((x1+x2+x3+x4+x5) \* 100)/500) <= 40:

                     grd = 'G'

              elif (((x1+x2+x3+x4+x5) \* 100)/500) <= 50:

                     grd = 'F'

              elif (((x1+x2+x3+x4+x5) \* 100)/500) <= 60:

                     grd = 'E'

              elif (((x1+x2+x3+x4+x5) \* 100)/500) <= 70:

                     grd = 'D'

              elif (((x1+x2+x3+x4+x5) \* 100)/500) <= 80:

                     grd = 'C'

              elif (((x1+x2+x3+x4+x5) \* 100)/500) <= 90:

                     grd = 'B'

              else:

                     grd = 'A'

              grade.set(grd)

              count = 0

              if x1 < 33:

                     count = count + 1

              if x2 < 33:

                     count = count + 1

              if x3 < 33:

                     count = count + 1

              if x4 < 33:

                     count = count + 1

              if x5 < 33:

                     count = count + 1

              if (count == 0):

                     result.set('PASS')

              elif (count == 1 or count == 2 ):

                     result.set('SUPPLY')

              else:

                     result.set('FAIL')

              if Per <= 45 and result != "FAIL":

                     div.set('THIRD')

              elif Per <= 60 and result != "FAIL":

                     div.set('SECOND')

              elif Per <= 100:

                     div.set('FIRST')

       #========================================================Frame\_1===============================================================

       Frame\_1 = LabelFrame(root, width = 1200, height = 400, font = ('arial',20,'bold'), bg = 'Navajo white', bd = 10, \

                            text = 'Student Details', relief = 'ridge')

       Frame\_1.grid(row = 1, column = 0, pady = 20, padx = 20)

       name = StringVar(Frame\_1,value=row[0][1])

       roll = StringVar(Frame\_1,value=row[0][2])

       fname = StringVar(Frame\_1,value=row[0][3])

       mname = StringVar(Frame\_1,value=row[0][4])

       DOB = StringVar(Frame\_1,value=row[0][5])

       gender = StringVar(Frame\_1,value=row[0][6])

       scl = StringVar(Frame\_1,value=row[0][7])

       email = StringVar(Frame\_1,value=row[0][8])

       #=================================================Labels and Entries for Frame\_1===============================================

       Label\_Name = Label(Frame\_1, text = 'Name', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_Name.grid(row = 0, column = 0, padx = 80)

       Entry\_Name = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = name)

       Entry\_Name.grid(row = 0, column = 1, padx = 5, pady = 5)

       Label\_Roll\_no = Label(Frame\_1, text = 'Roll Number', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_Roll\_no.grid(row = 0, column = 3, padx = 80)

       Entry\_Roll\_no = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = roll)

       Entry\_Roll\_no.grid(row = 0, column = 4, padx = 40)

       Label\_Father\_Name = Label(Frame\_1, text = 'Father Name', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_Father\_Name.grid(row = 1, column = 0, padx = 80)

       Entry\_Father\_Name = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = fname)

       Entry\_Father\_Name.grid(row = 1, column = 1, padx = 5, pady = 10)

       Label\_Mother\_Name = Label(Frame\_1, text = 'Mother Name', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_Mother\_Name.grid(row = 1, column = 3, padx = 80)

       Entry\_Mother\_Name = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = mname)

       Entry\_Mother\_Name.grid(row = 1, column = 4, padx = 5)

       Label\_DOB = Label(Frame\_1, text = 'Date of Birth', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_DOB.grid(row = 2, column = 0, padx = 80)

       Entry\_DOB = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = DOB)

       Entry\_DOB.grid(row = 2, column = 1, padx = 5, pady = 5)

       Label\_Gender = Label(Frame\_1, text = 'Gender', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_Gender.grid(row = 2, column = 3, padx = 80)

       Entry\_Gender = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = gender)

       Entry\_Gender.grid(row = 2, column = 4, padx = 5, pady = 5)

       Label\_School = Label(Frame\_1, text = 'School Name', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_School.grid(row = 3, column = 0, padx = 80)

       Entry\_School = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = scl)

       Entry\_School.grid(row = 3, column = 1, padx = 5, pady = 5)

       Label\_Email = Label(Frame\_1, text = 'Email ID', font = ('arial',15,'bold'), bg = 'Navajo white')

       Label\_Email.grid(row = 3, column = 3, padx = 80)

       Entry\_Email = Entry(Frame\_1, font = ('arial',15), width = 25, textvariable = email)

       Entry\_Email.grid(row = 3, column = 4, padx = 5, pady = 5)

       #========================================================Frame\_2==================================================================

       Frame\_2 = LabelFrame(root, width = 1200, height = 400, font = ('arial',20,'bold'), bg = 'Navajo white', bd = 10 \

                            , text = 'Grades Point Obtained', relief = 'ridge')

       Frame\_2.grid(row = 3, column = 0)

       m1 = DoubleVar(Frame\_2,row[0][9])

       m2 = DoubleVar(Frame\_2,row[0][10])

       m3 = DoubleVar(Frame\_2,row[0][11])

       m4 = DoubleVar(Frame\_2,row[0][12])

       m5 = DoubleVar(Frame\_2,row[0][13])

       gt = DoubleVar(Frame\_2,row[0][14])

       per = DoubleVar(Frame\_2,row[0][15])

       cgpa = DoubleVar(Frame\_2,row[0][16])

       grade = StringVar(Frame\_2,row[0][17])

       div = StringVar(Frame\_2,row[0][18])

       result = StringVar(Frame\_2,row[0][19])

       #======================================================Labels of Frame\_2===========================================================

       Label\_Subject = Label(Frame\_2, text = 'SUBJECT', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_Subject.grid(row = 3, column = 0, padx = 50, pady = 10)

       Label\_obt\_Marks = Label(Frame\_2, text = 'MARKS OBTAINED', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_obt\_Marks.grid(row = 3, column = 1, padx = 20)

       Label\_Subject = Label(Frame\_2, text = 'PASSING MARKS', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_Subject.grid(row = 3, column = 2, padx = 20)

       Label\_obt\_Marks = Label(Frame\_2, text = 'TOTAL MARKS', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_obt\_Marks.grid(row = 3, column = 3, padx = 20)

       Label\_1 = Label(Frame\_2, text = 'MATHEMATICS', font = ('arial',14), bg = 'Navajo white')

       Label\_1.grid(row = 4, column = 0)

       Label\_2 = Label(Frame\_2, text = 'PHYSICS', font = ('arial',14), bg = 'Navajo white')

       Label\_2.grid(row = 5, column = 0)

       Label\_3 = Label(Frame\_2, text = 'CHEMISTRY', font = ('arial',14), bg = 'Navajo white')

       Label\_3.grid(row = 6, column = 0)

       Label\_4 = Label(Frame\_2, text = 'HINDI', font = ('arial',14), bg = 'Navajo white')

       Label\_4.grid(row = 7, column = 0)

       Label\_5 = Label(Frame\_2, text = 'ENGLISH', font = ('arial',14), bg = 'Navajo white')

       Label\_5.grid(row = 8, column = 0)

       Label\_6 = Label(Frame\_2, text = 'GRAND TOTAL', font = ('arial',16), bg = 'Navajo white')

       Label\_6.grid(row = 9, column = 0)

       Label\_7 = Label(Frame\_2, text = 'PERCENTAGE', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_7.grid(row = 10, column = 0)

       Label\_8 = Label(Frame\_2, text = 'CGPA', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_8.grid(row = 10, column = 2)

       Label\_9 = Label(Frame\_2, text = 'GRADE', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_9.grid(row = 10, column = 4)

       Label\_10 = Label(Frame\_2, text = 'DIVISION', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_10.grid(row = 11, column = 0)

       Label\_10 = Label(Frame\_2, text = 'RESULT', font = ('arial',16,'bold'), bg = 'Navajo white')

       Label\_10.grid(row = 11, column = 2)

       #======================================================Entries of Frame\_2===========================================================

       var\_1 = StringVar(Frame\_2, value = '33')

       var\_2 = StringVar(Frame\_2, value = '100')

       var\_3 = StringVar(Frame\_2, value = '500')

       Entry\_\_1 = Entry(Frame\_2, font = ('arial',16), width = 5, textvariable = m1)

       Entry\_\_1.grid(row = 4, column = 1)

       Entry\_\_2 = Entry(Frame\_2, font = ('arial',16), width = 5, textvariable = m2)

       Entry\_\_2.grid(row = 5, column = 1)

       Entry\_\_3 = Entry(Frame\_2, font = ('arial',16), width = 5, textvariable = m3)

       Entry\_\_3.grid(row = 6, column = 1)

       Entry\_\_4 = Entry(Frame\_2, font = ('arial',16), width = 5, textvariable = m4)

       Entry\_\_4.grid(row = 7, column = 1)

       Entry\_\_5 = Entry(Frame\_2, font = ('arial',16), width = 5, textvariable = m5)

       Entry\_\_5.grid(row = 8, column = 1)

       Entry\_\_6 = Entry(Frame\_2, font = ('arial',14), width = 5, textvariable = gt)

       Entry\_\_6.grid(row = 9, column = 1, pady = 8)

       Entry\_\_7 = Entry(Frame\_2, font = ('arial',14,'bold'), width = 5, textvariable = per)

       Entry\_\_7.grid(row = 10, column = 1, pady = 8)

       Entry\_\_8 = Entry(Frame\_2, font = ('arial',14,'bold'), width = 5, textvariable = cgpa)

       Entry\_\_8.grid(row = 10, column = 3, pady = 8)

       Entry\_\_9 = Entry(Frame\_2, font = ('arial',14,'bold'), width = 5, textvariable = grade)

       Entry\_\_9.grid(row = 10, column = 5, padx = 20, pady = 8)

       Entry\_\_10 = Entry(Frame\_2, font = ('arial',14,'bold'), width = 8, textvariable = div)

       Entry\_\_10.grid(row = 11, column = 1, padx = 20, pady = 8)

       Entry\_\_11 = Entry(Frame\_2, font = ('arial',14,'bold'), width = 7, textvariable = result)

       Entry\_\_11.grid(row = 11, column = 3, padx = 20, pady = 8)

       Entry\_1\_2 = Entry(Frame\_2, textvariable = var\_1, font = ('arial',16), width = 5)

       Entry\_1\_2.grid(row = 4, column = 2, pady = 5)

       Entry\_1\_3 = Entry(Frame\_2, textvariable = var\_2, font = ('arial',16), width = 5)

       Entry\_1\_3.grid(row = 4, column = 3)

       Entry\_2\_2 = Entry(Frame\_2, textvariable = var\_1, font = ('arial',16), width = 5)

       Entry\_2\_2.grid(row = 5, column = 2, pady = 5)

       Entry\_2\_3 = Entry(Frame\_2, textvariable = var\_2, font = ('arial',16), width = 5)

       Entry\_2\_3.grid(row = 5, column = 3)

       Entry\_3\_2 = Entry(Frame\_2, textvariable = var\_1, font = ('arial',16), width = 5)

       Entry\_3\_2.grid(row = 6, column = 2, pady = 5)

       Entry\_3\_3 = Entry(Frame\_2, textvariable = var\_2, font = ('arial',16), width = 5)

       Entry\_3\_3.grid(row = 6, column = 3)

       Entry\_4\_2 = Entry(Frame\_2, textvariable = var\_1, font = ('arial',16), width = 5)

       Entry\_4\_2.grid(row = 7, column = 2, pady = 5)

       Entry\_4\_3 = Entry(Frame\_2, textvariable = var\_2, font = ('arial',16), width = 5)

       Entry\_4\_3.grid(row = 7, column = 3)

       Entry\_5\_2 = Entry(Frame\_2, textvariable = var\_1, font = ('arial',16), width = 5)

       Entry\_5\_2.grid(row = 8, column = 2, pady = 5)

       Entry\_5\_3 = Entry(Frame\_2, textvariable = var\_2, font = ('arial',16), width = 5)

       Entry\_5\_3.grid(row = 8, column = 3)

       Entry\_6\_3 = Entry(Frame\_2, textvariable = var\_3, font = ('arial',16), width = 5)

       Entry\_6\_3.grid(row = 9, column = 3)

       #======================================================Buttons===========================================================

       Btn\_Exit = Button(Frame\_2, text = 'EXIT', font = ('arial',12,'bold'), width = 10, command = root.destroy)

       Btn\_Exit.grid(row = 8, column = 4, padx = 50, pady = 6)

       root.mainloop()

if \_\_name\_\_ == '\_\_main\_\_':

       marksheet()

from tkinter import\*

import random

import os

def \_\_marksheet\_\_():

       filename = 'Search\_Page.py'

       os.system(filename)

       os.system('notepad'+filename)

def \_\_Library\_\_():

       filename = 'Library\_Frontend.py'

       os.system(filename)

       os.system('notepad'+filename)

def \_\_information\_\_():

       filename = 'Std\_info\_FrontEnd.py'

       os.system(filename)

       os.system('notepad'+filename)

def \_\_FeeReport\_\_():

       filename = 'Fee\_Frontend.py'

       os.system(filename)

       os.system('notepad'+filename)

def menu():

       root = Tk()

       root.title('Menu')

       root.geometry('1350x750')

       root.config(bg = 'navajo white')

       title\_Frame = LabelFrame(root, font = ('arial',50,'bold'), width = 1000, height = 100, bg = 'navajo white', relief = 'raise', bd = 13)

       title\_Frame.grid(row = 0, column = 0, pady = 50)

       title\_Label = Label(title\_Frame, text = 'MENU', font = ('arial',30,'bold'), bg = 'navajo white')

       title\_Label.grid(row = 0, column = 0, padx = 150)

       #========================================================FRAMES===================================================================

       Frame\_1 = LabelFrame(root, font = ('arial',17,'bold'), width = 1000, height = 100, bg = 'navajo white', relief = 'ridge', bd = 10)

       Frame\_1.grid(row = 1, column = 0, padx = 280)

       Frame\_2 = LabelFrame(root, font = ('arial',17,'bold'), width = 1000, height = 100, bg = 'navajo white', relief = 'ridge', bd = 10)

       Frame\_2.grid(row = 2, column = 0, padx = 130, pady = 7)

       Frame\_3 = LabelFrame(root, font = ('arial',17,'bold'), width = 1000, height = 100, bg = 'navajo white', relief = 'ridge', bd = 10)

       Frame\_3.grid(row = 3, column = 0, pady = 7)

       Frame\_4 = LabelFrame(root, font = ('arial',17,'bold'), width = 1000, height = 100, bg = 'navajo white', relief = 'ridge', bd = 10)

       Frame\_4.grid(row = 4, column = 0, pady = 7)

       #========================================================LABELS===================================================================

       Label\_1 = Label(Frame\_1, text = 'STUDENT PROFILE', font = ('arial',25,'bold'), bg = 'navajo white')

       Label\_1.grid(row = 0, column = 0, padx = 50, pady = 5)

       Label\_2 = Label(Frame\_2, text = 'FEE REPORT', font = ('arial',25,'bold'), bg = 'navajo white')

       Label\_2.grid(row = 0, column = 0, padx = 100, pady = 5)

       Label\_3 = Label(Frame\_3, text = 'LIBRARY SYSTEM', font = ('arial',25,'bold'), bg = 'navajo white')

       Label\_3.grid(row = 0, column = 0, padx = 60, pady = 5)

       Label\_4 = Label(Frame\_4, text = 'MARKSHEET', font = ('arial',25,'bold'), bg = 'navajo white')

       Label\_4.grid(row = 0, column = 0, padx = 101, pady = 5)

       #========================================================BUTTONS===================================================================

       Button\_1 = Button(Frame\_1, text = 'VIEW', font = ('arial',16,'bold'), width = 8, command = \_\_information\_\_)

       Button\_1.grid(row = 0, column = 3, padx = 50)

       Button\_2 = Button(Frame\_2, text = 'VIEW', font = ('arial',16,'bold'), width = 8, command = \_\_FeeReport\_\_)

       Button\_2.grid(row = 0, column = 3, padx = 50)

       Button\_3 = Button(Frame\_3, text = 'VIEW', font = ('arial',16,'bold'), width = 8, command = \_\_Library\_\_)

       Button\_3.grid(row = 0, column = 3, padx = 50)

       Button\_4 = Button(Frame\_4, text = 'VIEW', font = ('arial',16,'bold'), width = 8, command = \_\_marksheet\_\_)

       Button\_4.grid(row = 0, column = 3, padx = 50)

       root.mainloop()

if \_\_name\_\_ == '\_\_main\_\_':

       menu()

from tkinter import \*

import random

import Marksheet\_Backend

import Marksheet\_Frontend

import tkinter.messagebox

import os

class Window\_1():

       def \_\_init\_\_(self, master):

              self.master = master

              self.master.title('Search Page')

              self.master.geometry('1360x750')

              self.master.config(bg = 'navajowhite')

              self.roll = StringVar()

              frame = LabelFrame(self.master, width = 1000, height = 100, font = ('arial',30,'bold'), relief = 'ridge', bd = 15, bg = 'wheat')

              frame.grid(row = 1, column = 0, padx = 200, pady = 200)

              label = Label(frame, text = 'Enter Roll Number', font = ('arial',25,'bold'), bg = 'wheat' )

              label.grid(row = 0, column = 0, padx = 100, pady = 10)

              entry = Entry(frame, font = ('arial',25,'bold'), textvariable = self.roll)

              entry.grid(row = 0, column = 1, padx = 30, pady = 20)

              def Search():

                     if(len(self.roll.get()) != 0):

                           row = Marksheet\_Backend.search(int(self.roll.get()))

                           print(row)

                           Marksheet\_Frontend.search\_result\_marksheet(row)

                     else:

                            tkinter.messagebox.askokcancel('Attention','Please enter valid Roll No.')

                            return

              def new():

                     filename = 'Marksheet\_Frontend.py'

                     os.system(filename)

                     os.system('notepad'+filename)

              btnSearch = Button(frame, text = 'SEARCH', width = 15, font = ('arial',15,'bold'), command=Search)

              btnSearch.grid(row = 1, column = 0, padx = 50)

              btnNew = Button(frame, text = 'CREATE NEW', width = 15, font = ('arial',15,'bold'), command=new)

              btnNew.grid(row = 1, column = 1, padx = 50, pady = 20 )

root = Tk()

root.title("Login Form")

Window\_1(root)

root.mainloop()

import sqlite3

def connect():

       conn = sqlite3.connect("student.db")

       cur = conn.cursor()

       cur.execute("CREATE TABLE IF NOT EXISTS student (id INTEGER PRIMARY KEY, name text, fname text, mname text, \

                     address text, mobno integer,email text, dob integer, gender text)")

       conn.commit()

       conn.close()

def insert(name = " ", fname = " ", mname = " ", address = " ", mobno = " ", email = " ", dob = " ", gender = " "):

       conn = sqlite3.connect("student.db")

       cur = conn.cursor()

       cur.execute("INSERT INTO student VALUES (NULL,?,?,?,?,?,?,?,?)", (name, fname, mname, address , mobno, email, dob, gender))

       conn.commit()

       conn.close()

def view():

       conn = sqlite3.connect("student.db")

       cur = conn.cursor()

       cur.execute("SELECT \* FROM student")

       rows = cur.fetchall()

       return rows

       conn.close()

def delete(id):

       conn = sqlite3.connect("student.db")

       cur = conn.cursor()

       cur.execute("DELETE FROM student WHERE id = ?", (id,))

       conn.commit()

       conn.close()

def update(id,name = " ", fname = " ", mname = " ", address = " ", mobno = " ", email = " ", dob = " ", gender = " "):

       conn = sqlite3.connect("student.db")

       cur = conn.cursor()

       cur.execute("UPDATE student SET name = ? OR fname = ? OR mname = ? OR address = ? OR mobno = ? OR email = ? OR dob = ? OR gender = ?", \

                   (name, fname, mname, address , mobno, email, dob, gender))

       conn.commit()

       conn.close()

def search(name = " ", fname = " ", mname = " ", address = " ", mobno = " ", email = " ", dob = " ", gender = " "):

       conn = sqlite3.connect("student.db")

       cur = conn.cursor()

       cur.execute("SELECT \* FROM student WHERE name = ? OR fname = ? OR mname = ? OR address = ? OR mobno = ? OR email = ? OR dob = ? \

                     OR gender = ?", (name, fname, mname, address , mobno, email, dob, gender))

       rows = cur.fetchall()

       return rows

       conn.close()

connect()

from tkinter import\*

import tkinter.messagebox

import random

import Std\_info\_BackEnd

from tkinter import ttk

class Std\_info():

       def \_\_init\_\_(self, master):

              self.master = master

              self.master.title('Student Information')

              self.master.geometry('1350x750')

              self.master.config(bg = 'navajowhite')

              def information():

              #========================================================Variables=====================================================================

                     self.name = StringVar()

                     self.fname = StringVar()

                     self.mname = StringVar()

                     self.address = StringVar()

                     self.mobno = StringVar()

                     self.email = StringVar()

                     self.dob = StringVar()

                     self.gender = StringVar()

               #==========================================================Functions====================================================================

                     def StudentRec(events):

                            try:

                                   global selected\_tuple

                                   index = self.listbox.curselection()[0]

                                   selected\_tuple = self.listbox.get(index)

                                   self.Entry\_name.delete(0, END)

                                   self.Entry\_name.insert(END, selected\_tuple[1])

                                   self.Entry\_fname.delete(0, END)

                                   self.Entry\_fname.insert(END, selected\_tuple[2])

                                   self.Entry\_mname.delete(0, END)

                                   self.Entry\_mname.insert(END, selected\_tuple[3])

                                   self.Entry\_address.delete(0, END)

                                   self.Entry\_address.insert(END, selected\_tuple[4])

                                   self.Entry\_mobno.delete(0, END)

                                   self.Entry\_mobno.insert(END, selected\_tuple[5])

                                   self.Entry\_emailID.delete(0, END)

                                   self.Entry\_emailID.insert(END, selected\_tuple[6])

                                   self.Entry\_dob.delete(0, END)

                                   self.Entry\_dob.insert(END, selected\_tuple[7])

                                   self.Entry\_gender.delete(0, END)

                                   self.Entry\_gender.insert(END, selected\_tuple[8])

                            except IndexError:

                                   pass

                     def Add():

                            if(len(self.name.get()) != 0):

                               Std\_info\_BackEnd.insert(self.name.get(), self.fname.get(), self.mname.get(), self.address.get(), self.mobno.get(), self.email.get(), self.dob.get(), \

                                                       self.gender.get())

                               self.listbox.delete(0, END)

                               self.listbox.insert(END, (self.name.get(), self.fname.get(), self.mname.get(), self.address.get(), self.mobno.get(), self.email.get(), self.dob.get(), \

                                                       self.gender.get()))

                     def Display():

                               self.listbox.delete(0, END)

                               for row in Std\_info\_BackEnd.view():

                                      self.listbox.insert(END, row, str(' '))

                     def Exit():

                            Exit = tkinter.messagebox.askyesno("Login System", "Confirm if you want to Exit")

                            if Exit > 0:

                                   self.master.destroy()

                                   return

                     def Reset():

                            self.name.set('')

                            self.fname.set('')

                            self.mname.set('')

                            self.address.set('')

                            self.mobno.set('')

                            self.email.set('')

                            self.dob.set('')

                            self.gender.set('')

                            self.listbox.delete(0, END)

                     def Delete():

                            if(len(self.name.get()) != 0):

                               Std\_info\_BackEnd.delete(selected\_tuple[0])

                               Reset()

                               Display()

                     def Search():

                            self.listbox.delete(0, END)

                            for row in Std\_info\_BackEnd.search(self.name.get(), self.fname.get(), self.mname.get(), self.address.get(), self.mobno.get(), self.email.get(), self.dob.get(),self.gender.get()):

                                   self.listbox.insert(END, row, str(' '))

                     def Update():

                            if(len(self.name.get()) != 0):

                               Std\_info\_BackEnd.delete(selected\_tuple[0])

                            if(len(self.name.get()) != 0):

                               Std\_info\_BackEnd.insert(self.name.get(), self.fname.get(), self.mname.get(), self.address.get(), self.mobno.get(), self.email.get(), self.dob.get(), \

                                                       self.gender.get())

                               self.listbox.delete(0, END)

                               self.listbox.insert(END, (self.name.get(), self.fname.get(), self.mname.get(), self.address.get(), self.mobno.get(), self.email.get(), self.dob.get(), \

                                                       self.gender.get()))

                     #============================================================Frames=====================================================================

                     self.Main\_Frame = LabelFrame(self.master, width = 1300, height = 500, font = ('arial',20,'bold'), \

                                                  bg = 'navajowhite',bd = 15, relief = 'ridge')

                     self.Main\_Frame.grid(row = 0, column = 0, padx = 10, pady = 20)

                     self.Frame\_1 = LabelFrame(self.Main\_Frame, width = 600, height = 400, font = ('arial',15,'bold'), \

                                               relief = 'ridge', bd = 10, bg = 'navajowhite', text = 'STUDENT INFORMATION ')

                     self.Frame\_1.grid(row = 1, column = 0, padx = 10)

                     self.Frame\_2 = LabelFrame(self.Main\_Frame, width = 750, height = 400, font = ('arial',15,'bold'), \

                                               relief = 'ridge', bd = 10, bg = 'navajowhite', text = 'STUDENT DATABASE')

                     self.Frame\_2.grid(row = 1, column = 1, padx = 5)

                     self.Frame\_3 = LabelFrame(self.master, width = 1200, height = 100, font = ('arial',10,'bold'), \

                                               bg = 'navajowhite', relief = 'ridge', bd = 13)

                     self.Frame\_3.grid(row = 2, column = 0, pady = 10)

                     #========================================================Labels of Frame\_1========================================================

                     self.Label\_name = Label(self.Frame\_1, text = 'Name', font = ('arial',20,'bold'),  bg = 'navajowhite')

                     self.Label\_name.grid(row = 0, column = 0, sticky = W, padx = 20, pady = 10)

                     self.Label\_fname = Label(self.Frame\_1, text = 'Father Name', font = ('arial',20,'bold'),  bg = 'navajowhite')

                     self.Label\_fname.grid(row = 1, column = 0, sticky = W, padx = 20)

                     self.Label\_mname = Label(self.Frame\_1, text = 'Mother Name', font = ('arial',20,'bold'),  bg = 'navajowhite')

                     self.Label\_mname.grid(row = 2, column = 0, sticky = W, padx = 20)

                     self.Label\_address = Label(self.Frame\_1, text = 'Address', font = ('arial',20,'bold'),  bg = 'navajowhite')

                     self.Label\_address.grid(row = 3, column = 0, sticky = W, padx = 20)

                     self.Label\_mobno = Label(self.Frame\_1, text = 'Mobile Number', font = ('arial',20,'bold'),  bg = 'navajowhite')

                     self.Label\_mobno.grid(row = 4, column = 0, sticky = W, padx = 20)

                     self.Label\_emailID = Label(self.Frame\_1, text = 'Email ID', font = ('arial',20,'bold'),  bg = 'navajowhite')

                     self.Label\_emailID.grid(row = 5, column = 0, sticky = W, padx = 20)

                     self.Label\_dob = Label(self.Frame\_1, text = 'Date of Birth', font = ('arial',20,'bold'),  bg = 'navajowhite')

                     self.Label\_dob.grid(row = 6, column = 0, sticky = W, padx = 20)

                     self.Label\_gender = Label(self.Frame\_1, text = 'Gender', font = ('arial',20,'bold'),  bg = 'navajowhite')

                     self.Label\_gender.grid(row = 7, column = 0, sticky = W, padx = 20, pady = 10)

                     #========================================================Entries of Frame\_1========================================================

                     self.Entry\_name = Entry(self.Frame\_1, font = ('arial',17,'bold'), textvariable = self.name)

                     self.Entry\_name.grid(row = 0, column = 1, padx = 10, pady = 5)

                     self.Entry\_fname = Entry(self.Frame\_1, font = ('arial',17,'bold'), textvariable = self.fname)

                     self.Entry\_fname.grid(row = 1, column = 1, padx = 10, pady = 5)

                     self.Entry\_mname = Entry(self.Frame\_1, font = ('arial',17,'bold'), textvariable = self.mname)

                     self.Entry\_mname.grid(row = 2, column = 1, padx = 10, pady = 5)

                     self.Entry\_address = Entry(self.Frame\_1, font = ('arial',17,'bold'), textvariable = self.address)

                     self.Entry\_address.grid(row = 3, column = 1, padx = 10, pady = 5)

                     self.Entry\_mobno = Entry(self.Frame\_1, font = ('arial',17,'bold'), textvariable = self.mobno)

                     self.Entry\_mobno.grid(row = 4, column = 1, padx = 10, pady = 5)

                     self.Entry\_emailID = Entry(self.Frame\_1, font = ('arial',17,'bold'), textvariable = self.email)

                     self.Entry\_emailID.grid(row = 5, column = 1, padx = 10, pady = 5)

                     self.Entry\_dob = Entry(self.Frame\_1, font = ('arial',17,'bold'), textvariable = self.dob)

                     self.Entry\_dob.grid(row = 6, column = 1, padx = 10, pady = 5)

                     self.Entry\_gender = ttk.Combobox(self.Frame\_1, values = (' ','Male','Female','Others'),\

                                                      font = ('arial',17,'bold'), textvariable = self.gender, width = 19)

                     self.Entry\_gender.grid(row = 7, column = 1, padx = 10, pady = 5)

                     #========================================================Buttons of self.Frame\_3=========================================================

                     self.btnSave = Button(self.Frame\_3, text = 'SAVE', font = ('arial',17,'bold'), width = 8, command = Add)

                     self.btnSave.grid(row = 0, column = 0, padx = 10, pady = 10)

                     self.btnDisplay = Button(self.Frame\_3, text = 'DISPLAY', font = ('arial',17,'bold'), width = 8, command = Display)

                     self.btnDisplay.grid(row = 0, column = 1, padx = 10, pady = 10)

                     self.btnReset = Button(self.Frame\_3, text = 'RESET', font = ('arial',17,'bold'), width = 8, command = Reset)

                     self.btnReset.grid(row = 0, column = 2, padx = 10, pady = 10)

                     self.btnUpdate = Button(self.Frame\_3, text = 'UPDATE', font = ('arial',17,'bold'), width = 8, command = Update)

                     self.btnUpdate.grid(row = 0, column = 3, padx = 10, pady = 10)

                     self.btnDelete = Button(self.Frame\_3, text = 'DELETE', font = ('arial',17,'bold'), width = 8, command = Delete)

                     self.btnDelete.grid(row = 0, column = 4, padx = 10, pady = 10)

                     self.btnSearch = Button(self.Frame\_3, text = 'SEARCH', font = ('arial',17,'bold'), width = 8, command = Search )

                     self.btnSearch.grid(row = 0, column = 5, padx = 10, pady = 10)

                     self.btnExit = Button(self.Frame\_3, text = 'EXIT', font = ('arial',17,'bold'), width = 8, command = Exit)

                     self.btnExit.grid(row = 0, column = 6, padx = 10, pady = 10)

                     #===============================================List Box and self.scrollbar========================================================

                     self.scrollbar = Scrollbar(self.Frame\_2)

                     self.scrollbar.grid(row = 0, column = 1, sticky = 'ns')

                     self.listbox = Listbox(self.Frame\_2, width = 75, height = 20 , font = ('arial',12,'bold'))

                     self.listbox.bind('<<ListboxSelect>>', StudentRec)

                     self.listbox.grid(row = 0, column = 0)

                     self.scrollbar.config(command = self.listbox.yview)

              information()

root = Tk()

obj = Std\_info(root)

root.mainloop()