Report on Attendance Management System by Dimuthu Fernando

Problem: Design GUI based Python application with Class Teacher as admin and Subject Teacher as normal user module. The class teacher should be able to decide subjects, theory teachers for each subject, batch wise laboratory teachers for each subject etc. The class teacher should be able to grant attendance for activities done by students, medical reasons etc. The subject teacher should be able to enter theory and laboratory attendance as well as can see student attendance percentage. The admin must be able to generate less attendance students list as per the given criteria.

Code:

```
import tkinter as tk from tkinter
import * from tkinter import
messagebox import os
import sqlite3 as sql



root=Tk() root.title('Application')
root.geometry('700x700') l=('Class
Teacher','Subject Teacher')
variable=tk.StringVar(root)
variable.set(l[0])
opt=tk.OptionMenu(root,variable,*l) opt.pack()



class ClassTeacher(tk.Tk):

    def __init__(self):
        tk.Tk.__init__(self)
self.geometry('1500x2000')
        self.title('Class Teacher')
self.frame=Frame(self,width=100, height=400,bd=2)
self.frame.grid(row=0,column=0)
##      self.frame1=Frame(self,width=100, height=400,bd=2)
##      self.frame1.grid(row=0,column=2)
        self.sems=(1,2,3,4,5,6,7,8)
self.var=tk.IntVar(self)        self.var.set(self.sems[0])
        self.label=tk.Label(self.frame,text='Set the subjects and teachers')
self.label1=tk.Label(self.frame,text='Enter Semester:')
self.opt=tk.OptionMenu(self.frame,self.var,*self.sems)
self.label2=tk.Label(self.frame,text='Enter number of subjects:')        self.e1=tk.Entry(self.frame)
        self.b1=Button(self.frame,text='Ok',command=self.on_ok)
self.label.grid(row=0,column=0)        self.label1.grid(row=1,column=0)
self.opt.grid(row=1,column=1)        self.label2.grid(row=2,column=0)
self.e1.grid(row=2,column=1)        self.b1.grid(row=4,column=1)
```

```
   def on_ok(self):         self.sem=int(self.var.get())         self.n=int(self.e1.get())
self.lab1=[tk.Label(self.frame,text='Enter subject name:') for x in range(self.n)]
self.lab2=[tk.Label(self.frame,text='Enter theory teacher name:') for x in range(self.n)]
self.lab3=[tk.Label(self.frame,text='Enter Lab-A teacher name:') for x in range(self.n)]
self.lab4=[tk.Label(self.frame,text='Enter Lab-B teacher name:') for x in range(self.n)]

        self.en1=[tk.Entry(self.frame) for x in range(self.n)]
self.en2=[tk.Entry(self.frame) for x in range(self.n)]
self.en3=[tk.Entry(self.frame) for x in range(self.n)]
self.en4=[tk.Entry(self.frame) for x in range(self.n)]         for
i in range(self.n):
            self.lab1[i].grid(row=i+5,column=0)
self.en1[i].grid(row=i+5,column=1)             self.lab2[i].grid(row=i+5,column=2)
self.en2[i].grid(row=i+5,column=3)             self.lab3[i].grid(row=i+5,column=4)
self.en3[i].grid(row=i+5,column=5)             self.lab4[i].grid(row=i+5,column=6)
            self.en4[i].grid(row=i+5,column=7)

        self.bu1=Button(self.frame,text='Submit',command=self.submit2)
self.bu1.grid(row=5+self.n,column=1)

   def generate(self):         if
os.path.isfile(self.entry1.get()+'.txt'):
            print('exists')
pdf = FPDF()
pdf.add_page()
            pdf.set_font('Arial',size=15)
file1=open(self.entry1.get()+'.txt','r')             for
i in file1:
                pdf.cell(200,10,txt=i,ln=1,align='C')
pdf.output(self.entry1.get()+'.pdf')             os.startfile(self.entry1.get()+'.pdf')

else:
            self.label6=tk.Label(self.frame1,text='Student with the enrollment number hasn\'t yet
entered').grid(row=4,column=0)
   def reset(self):
f=open('data.json','r')
semester=json.load(f)
f.close()
self.var.set(self.sems[0])
self.e1.delete(0,'end')         for
i in range(self.n):
self.lab1[i].destroy()
self.lab2[i].destroy()
self.bu1.destroy()
self.en1[i].destroy()
self.en2[i].destroy()
```

```
    def submit2(self):
self.sem=int(self.var.get())        for
i in range(self.n):
            strvar="Sem : "+str(self.sem)+"\nSubject : "+str(self.en1[i].get())+"\nTheory Teacher
name : "+str(self.en2[i].get())+"\nLab-A Teacher name : "+str(self.en3[i].get())+"\nLab-B
Teacher name : "+str(self.en4[i].get())

            messagebox.showinfo("information",strvar)




    def submit(self):
        messagebox.showinfo("information","Information")

        f=open('data.json','w')        if
self.sem==int(self.var.get()):
for i in range(int(self.e1.get())):
                subjects[self.en1[i].get()]=int(self.en2[i].get())
semester[int(self.var.get())-1].update(subjects)            print(semester)
            json.dump(semester,f)
f.close()            self.reset()
##          if self.label3:
##              self.label3.destroy() ##
else:
##          self.label3=tk.Label(self.frame,text='The semester you selected doesn\'t match to the
one when you started filling form.')
##          self.label3.grid(row=12,column=2)
class SubjectTeacher(tk.Tk):            def
__init__(self,*args,**kwargs):                  tk.Tk.
__init__(self,*args,**kwargs)
                    container=tk.Frame(self)

                   container.pack(side="top",fill="both",expand=True)
          container.grid_rowconfigure(0,weight=1)
                    container.grid_columnconfigure(0,weight=1)

                   self.frames=dict()
                   for F in
(StartPage,NewRecord,ManageAttendance,DeleteRecord,EditRecord,AddSubjects,TodayData):
                              frame=F(container,self)
                              self.frames[F]=frame
                              frame.grid(row=0,column=0,sticky="nsew")

                   self.show_frame(StartPage)
          def show_frame(self,cont):
          frame=self.frames[cont]
                    frame.tkraise()
```

```
class StartPage(tk.Frame):    def
__init__(self,parent,controller):
        tk.Frame.__init__(self,parent)


                    label1=tk.Label(self,text="Hi!, what do you desire?",font=("Arial",24))

                    bt1=tk.Button(self,text="Start a new
record",font=("Arial",16),height=2,width=17,command=lambda:controller.show_frame(NewRec
ord))
                    bt2=tk.Button(self,text="Manage
attendance",font=("Arial",16),height=2,width=17,command=lambda:controller.show_frame(Ma
nageAttendance))
                    bt3=tk.Button(self,text="Delete a
record",font=("Arial",16),height=2,width=17,command=lambda:controller.show_frame(DeleteR
ecord))
                    bt4=tk.Button(self,text="Edit the
record",font=("Arial",16),height=2,width=17,command=lambda:controller.show_frame(EditRec
ord))
                  label1.pack()
        bt1.pack()
        bt2.pack()
        bt3.pack()

                                        if w[2]==0 and w[3]==0: bt4.pack()

class NewRecord(tk.Frame):            def
__init__(self,parent,controller):
        tk.Frame.__init__(self,parent)
                  label1=tk.Label(self,text="New Record",font=("Arial",24))
        label2=tk.Label(self,text="if you want a new record, previous one will be
deleted,continue?",font=("Arial",12))


        bt2=tk.Button(self,text="YES",font=("Arial",16),height=2,width=17,command=lambda:c
ontroller.show_frame(AddSubjects))

        bt3=tk.Button(self,text="NO",font=("Arial",16),height=2,width=17,command=lambda:co
ntroller.show_frame(StartPage))
                  label1.pack()
        label2.pack()
        bt2.pack()
        bt3.pack()



class ManageAttendance(tk.Frame):
        def __init__(self,parent,controller):
        tk.Frame.__init__(self,parent)

                    label1=tk.Label(self,text="Manage Attendance",font=("Arial",24))
                    label1.pack()
                    bt2=tk.Button(self,text="show
  status",font=("Arial",16),height=2,width=17,command=lambda:self.showstatus(controller))
          bt3=tk.Button(self,text="Today's
  data",font=("Arial",16),height=2,width=17,command=lambda:controller.show_frame(TodayDat
 a))

            bt1=tk.Button(self,text="home",font=("Arial",16),height=2,width=17,command=lambda:
  controller.show_frame(StartPage))
          bt2.pack()
          bt3.pack()
          bt1.pack()       def
  showstatus(self,controller):
          try:
                              conn=sql.connect("attend")
                              cur=conn.cursor()
                                    cur.execute('SELECT * FROM
                   attable')        text="" for w in cur:
                                            per="0"
                                    else:
                                             per=w[2]/(w[2]+w[3])
                                             per=per*100
                                             per=str(int(per))
                                    text=text+"sub id "+str(w[0])+" "+w[1]+" "+per+"%\n"
                            messagebox.showinfo("status", text)                except:
                            messagebox.showinfo("alert!", "There is no record")
           class DeleteRecord(tk.Frame):          def
  __init__(self,parent,controller):                  tk.Frame.__init__(self,parent)
                   label1=tk.Label(self,text="Delete Record",font=("Times",24))
                   label2=tk.Label(self,text="This action will delete the
  record,continue?",font=("Times",12))

          bt2=tk.Button(self,text="YES",font=("Times",16),height=2,width=17,command=lambda:
  self.delrecord(controller))

          bt1=tk.Button(self,text="NO",font=("Times",16),height=2,width=17,command=lambda:c
  ontroller.show_frame(StartPage))
                   label1.pack()
          label2.pack()             bt2.pack()
                   bt1.pack()       def
  delrecord(self,controller):
          conn=sql.connect('attend')
          cur=conn.cursor()
                   cur.execute('DROP TABLE IF EXISTS attable')
          conn.commit()
                    conn.close()
                    messagebox.showinfo("alert!", "records deleted")
                    controller.show_frame(StartPage)
```

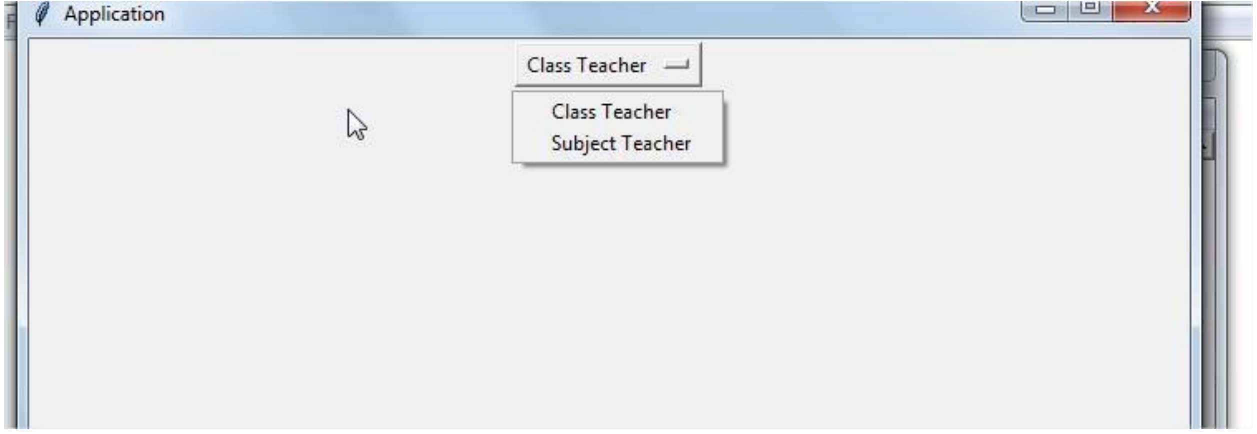
```
class EditRecord(tk.Frame):
        def __init__(self,parent,controller):
        tk.Frame.__init__(self,parent)
                label1=tk.Label(self,text="Edit Record",font=("Arial",24))




          bt1=tk.Button(self,text="home",font=("Arial",16),height=2,width=17,command=lambda:
controller.show_frame(StartPage))
                label1.pack()
                lb2=tk.Label(self,text="input the corresponding subject id",font=("Arial",10))
txt1=tk.Entry(self) lb2.pack()                    txt1.pack()
               lb3=tk.Label(self,text="number of times attended",font=("Arial",10))
        txt2=tk.Entry(self)
               lb4=tk.Label(self,text="number of times bunked",font=("Arial",10))
               txt3=tk.Entry(self)                lb3.pack()                 txt2.pack()
        lb4.pack()
                txt3.pack()

        bt3=tk.Button(self,text="Update",font=("Arial",16),height=2,width=17,command=lambd
a:self.update(txt1.get(),txt2.get(),txt3.get()))
                bt2=tk.Button(self,text="showid of
subjects",font=("Arial",16),height=2,width=17,command=lambda:self.showid(controller))
               bt2.pack()
        bt3.pack()
        bt1.pack()      def
update(self,i,p,b):
                i=int(i)
                             if p=="" or
p==" " or p=="\n":
                      p=0
        else:
                      p=int(p)                   if
b=="" or b==" " or b=="\n":
                      b=0
        else:
                      b=int(b)
               try:

                      conn=sql.connect("attend")
        cur=conn.cursor()
                      cur.execute("SELECT * FROM attable WHERE subid=?",(i,))
        kk=cur.fetchone()
                       np=p

                       nb=b

                       cur.execute("UPDATE attable SET attended = ? WHERE subid=
?",(np,i))     cur.execute("UPDATE   attable   SET   bunked   =   ?   WHERE   subid=   ?",(nb,i))
                    conn.commit()
                              conn.close()
                              messagebox.showinfo("alert!", "Updated")

                    except:
                              messagebox.showinfo("alert!", "There is no record")
                  def
showid(self,controller):
        try:
                              conn=sql.connect("attend")
                              cur=conn.cursor()
                              cur.execute('SELECT * FROM attable')
                            text=""
                            for w in cur:
                                        text=text+"sub id "+str(w[0])+" "+w[1]+"\n"
                              messagebox.showinfo("subject id", text)
                            conn.commit()
                            conn.close()
        except:
                            messagebox.showinfo("alert!", "There is no record")
class AddSubjects(tk.Frame):           def __init__(self,parent,controller):
        tk.Frame.__init__(self,parent)
                  label1=tk.Label(self,text="add subjects name seperated by
commas(,)",font=("Arial",12))
                    txt1=tk.Text(self,font=("Arial",16),width=48,height=3)

                  bt2=tk.Button(self,text="Add
subjects!",font=("Arial",16),height=1,width=17,command=lambda:self.addsub(txt1.get("1.0",tk.
END),controller))

          bt1=tk.Button(self,text="home",font=("Arial",16),height=2,width=17,command=lambda:
controller.show_frame(StartPage))
        label1.pack()
        txt1.pack()
        bt2.pack()
                  bt1.pack()       def
addsub(self,a,controller):

                    conn=sql.connect('attend')
                    cur=conn.cursor()
                    cur.execute('DROP TABLE IF EXISTS attable')

                    a=a[0:len(a)-1]
                    a=a.split(",")

                    if len(a)==1 and a[0]=="" :
                              messagebox.showinfo("alert!", "Please enter the subjects")
                    else:
                              sid=1
```

```
                                cur.execute('CREATE TABLE attable(subid INTEGER,subject
TEXT,attended INTEGER,bunked INTEGER)')
for sub in a:
                                        cur.execute('INSERT INTO attable
(subid,subject,attended,bunked) VALUES(?,?,?,?)',(sid,sub,0,0))
                                        sid=sid+1
                                conn.commit()
                                conn.close()
                                messagebox.showinfo("Congratulations!","subjects are added")
                                controller.show_frame(StartPage)



class TodayData(tk.Frame):          def
__init__(self,parent,controller):
        tk.Frame.__init__(self,parent)
                    label1=tk.Label(self,text="Enter data of today",font=("Arial",24))
                    label1.pack()
                    bt2=tk.Button(self,text="showid of
subjects",font=("Arial",16),height=2,width=17,command=lambda:self.showid(controller))

        bt1=tk.Button(self,text="home",font=("Arial",16),height=2,width=17,command=lambda:
controller.show_frame(StartPage))
                  lb2=tk.Label(self,text="input the corresponding subject id",font=("Arial",10))
                  txt1=tk.Entry(self)          lb2.pack()          txt1.pack()
                  lb3=tk.Label(self,text="number of lectures attended",font=("Arial",10))
        txt2=tk.Entry(self)
                  lb4=tk.Label(self,text="number of lectues bunked",font=("Arial",10))
                  txt3=tk.Entry(self)          lb3.pack()          txt2.pack()
        lb4.pack()
                    txt3.pack()
                    bt3=tk.Button(self,text="add to
record",font=("Arial",16),height=2,width=17,command=lambda:self.addrecord(txt1.get(),txt2.ge
t(),txt3.get()))
                  bt3.pack()
        bt2.pack()
        bt1.pack()      def
showid(self,controller):
                    try:
                                conn=sql.connect("attend")
                                cur=conn.cursor()
                                cur.execute('SELECT * FROM attable') text=""
        for w in cur:                              text=text+"sub id
"+str(w[0])+" "+w[1]+"\n"
                                messagebox.showinfo("subject id", text)
                              conn.commit()
                              conn.close()
        except:
                              messagebox.showinfo("alert!", "There is no record")
        def addrecord(self,i,p,b):
```

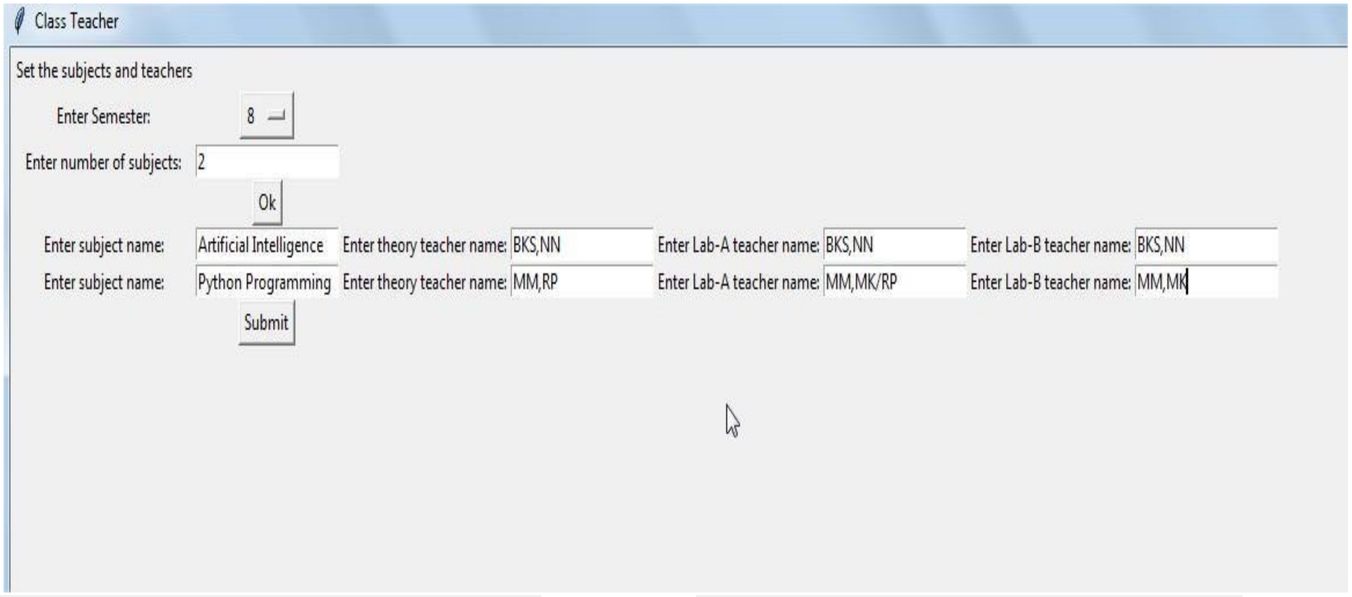
```
                    i=int(i)
                                    if p=="" or
p==" " or p=="\n":
                            p=0
        else:
                            p=int(p)                    if
b=="" or b==" " or b=="\n":
                            b=0
        else:
                            b=int(b)
                  try:

                             conn=sql.connect("attend")
                             cur=conn.cursor()
                            cur.execute("SELECT * FROM attable WHERE subid=?",(i,))
                            kk=cur.fetchone()                          np=kk[2]+p

                             nb=kk[3]+b

                             cur.execute("UPDATE attable SET attended = ? WHERE subid=
?",(np,i))
                            cur.execute("UPDATE attable SET bunked = ? WHERE subid= ?",(nb,i))
                            conn.commit()                              conn.close()
                            messagebox.showinfo("alert!", "Done")
        except:
                             messagebox.showinfo("alert!", "There is no record")

def get_variable(*args):     if
variable.get()=='Class Teacher':
      c=ClassTeacher()
#print(subjects)        c.mainloop()

   else:
      s=SubjectTeacher()
s.mainloop()

variable.trace('w',get_variable)

root.mainloop()
```

Output :

Select user:



Class Teacher:


Class Teacher

Set the subjects and teachers

Enter Semester: 8

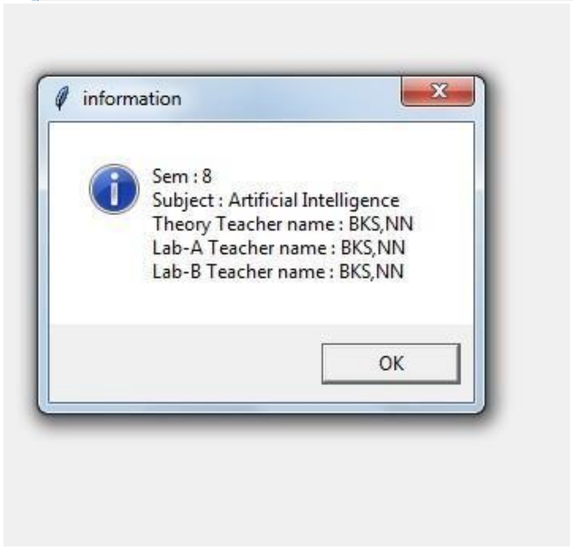
Enter number of subjects: 2

Ok

Enter subject name: Artificial Intelligence Enter theory teacher name: BKS,NN Enter Lab-A teacher name: BKS,NN Enter Lab-B teacher name: BKS,NN

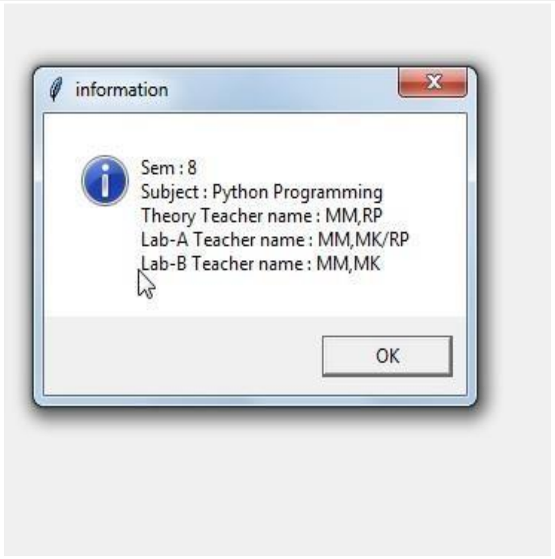
Enter subject name: Python Programming Enter theory teacher name: MM,RP Enter Lab-A teacher name: MM,MK/RP Enter Lab-B teacher name: MM,MK

Submit


information

Sem : 8
Subject : Artificial Intelligence
Theory Teacher name : BKS,NN
Lab-A Teacher name : BKS,NN
Lab-B Teacher name : BKS,NN

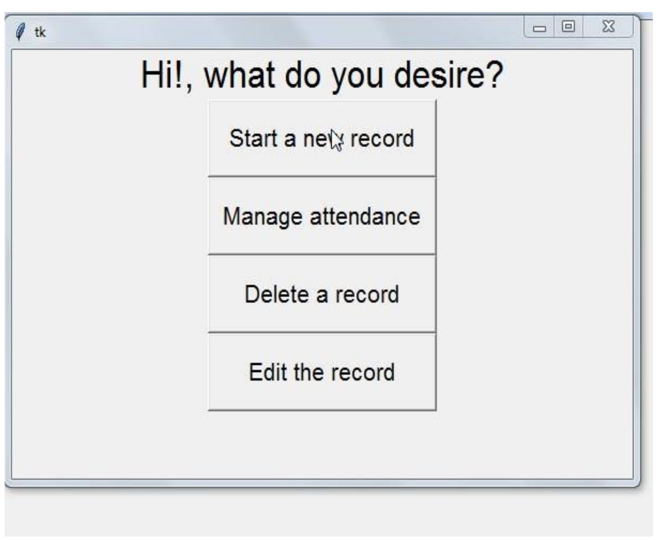
OK

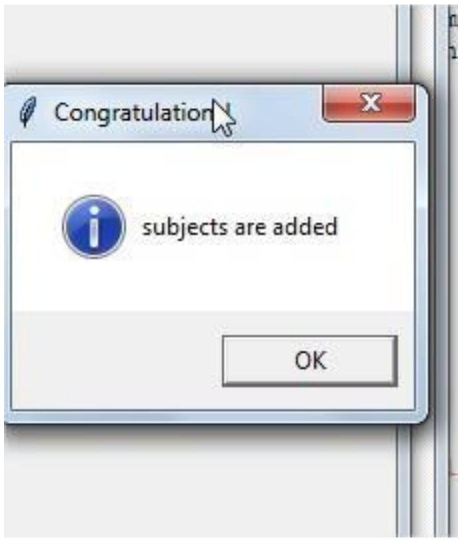

information

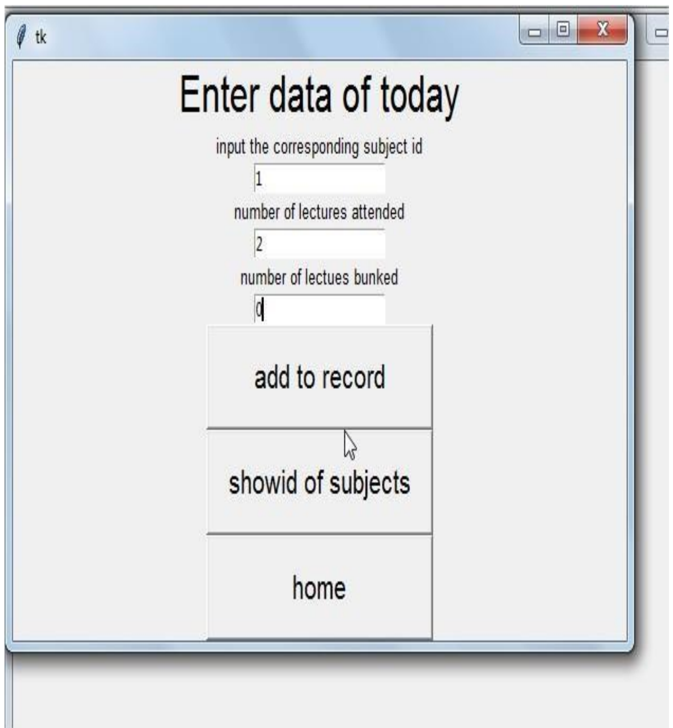
Sem : 8
Subject : Python Programming
Theory Teacher name : MM,RP
Lab-A Teacher name : MM,MK/RP
Lab-B Teacher name : MM,MK

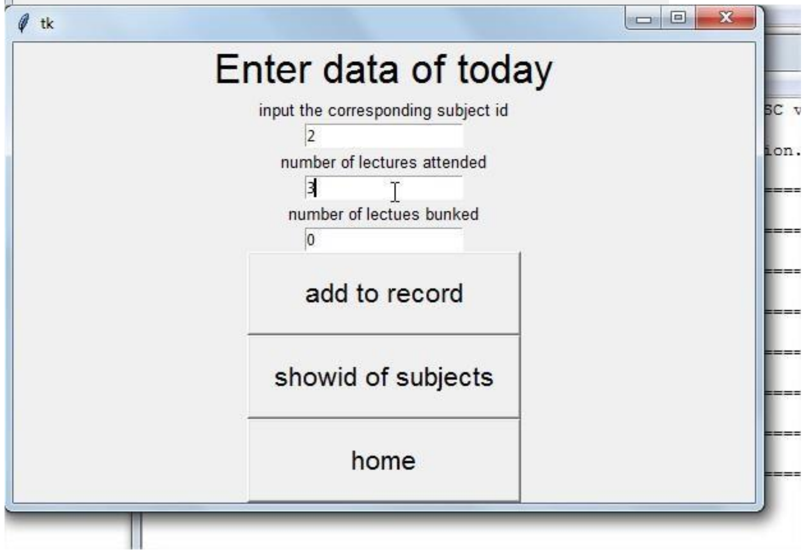
OK

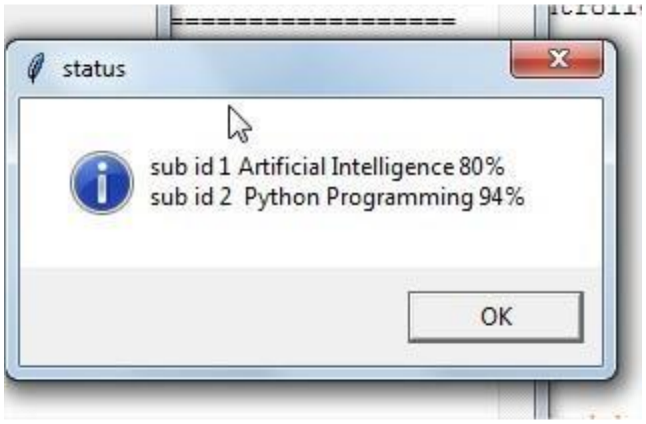
Subject Teacher:


tk
Hi!, what do you desire?
Start a new record
Manage attendance
Delete a record
Edit the record


Congratulation
subjects are added
OK


tk
Enter data of today
input the corresponding subject id
1
number of lectures attended
2
number of lectues bunked
0
add to record
showid of subjects
home


tk
Enter data of today
input the corresponding subject id
2
number of lectures attended
3
number of lectues bunked
0
add to record
showid of subjects
home



```
sub id 1 Artificial Intelligence 80%
sub id 2  Python Programming 94%
```