Delhi Public School, Navi Mumbai

COMPUTER SCIENCE PROJECT FILE

PROJECT NAME : T.O.H.F.A.
STUDENT NAME : AAYUSH RAJESH
ROLL NO : 15606576

Index

ACKNOWLEDGEMENT

I, Aayush Rajesh, would like to thank my Computer Science teacher, Mrs. Radhika Sridhar, who taught me Python. It was because of her excellence in the language and her constant support that I was able to do this project in the first place.
Also, I would like to thank our Principal sir, Mr. J. Mohanty, and our Vice-Principal ma'am, Mrs. Sravani Rao. This project led to me learning a lot more about Python language than I did. It also inculcated the values of teamwork and time management in me, which is very valuable in the development of an individual.
I would also like to thank my group members, Aryan Tiwari and Satyam Rath, without whom this project wouldn't have been possible. Lastly, I would like to thank all my classmates who have contributed to this project in some way or the other.

REQUIREMENTS

HARDWARE REQUIREMENTS:

- CD Drive/USB port
- 1 GB RAM or higher
- External plugged in audio device

SOFTWARE REQUIREMENTS:

- Spyder Python 3.6+
- MySQL Prompt
- Python Modules:
  - mysql.connector
  - speech_recognition
  - datetime
  - time
  - pickle
  - sys
  - tkinter
  - pyttsx3
  - msgpack

About Python and MySQL

PYTHON: Python is an interpreted, object-oriented, high-level programming language. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive.
Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace.

MYSQL: MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

SYNOPSIS

T.O.H.F.A. – **T**eacher **O**riented **H**elping and **F**unctioning **A**ssistant

TOHFA, as the full form above suggests, is a program developed solely keeping the teachers in mind and aims to reduce the work done by teachers by providing them with an easy-to-operate program to store data about students.

The program enables teachers to store, update, delete and view marks and attendance records for students of a particular class. MySQL connectivity, Tkinter and File Handling has been used in the program to ensure data security.

Functions have been used to ensure that the code is easy to read; while majority of the code has been audio enabled, meaning that both output and input can be received/given in audio format. Further, the menu structure has been used for a better user interface.

AIMS AND OBJECTIVES

- To allow different teachers to access their records by maintaining different accounts and the secure storage of user details through binary files, which are corrupted of forceful opening.
- To allow teachers to enter and access records of attendance and marks with ease.
- To ensure the safe storage of the records through the MySQL server.
- To provide a simple and quick analysis of records stored to assist teachers in analysis and overall statistics.
- To reduce the workload of teachers by providing an easy-to-use system, assisted by a voice assistant.

PROJECT CODE

```
print('''REQUIREMENTS
Module Requirements:
 1.mysql.connector
 2.speech_recognition
 3.datetime
 4.time
 5.pickle
 6.sys
 7.tkinter
 8.pyttsx3
 9.msgpack
External Hardware Requirements:
 1.Plugged-in Audio Device''')
dummy=input('Press ENTER to continue')

try:  #Checking for module requirement
    import mysql.connector as m
    import speech_recognition as sr
    import datetime
    import time
    import pickle as p
    import sys
    from tkinter import *
    import pyttsx3
    engine=pyttsx3.init()
    voices = engine.getProperty('voices')
    engine.setProperty('voice', voices[1].id)
except ModuleNotFoundError:
    print('Module Requirement not matched. Exiting program.')
    sys.exit()
    time.sleep(2)

def speech_out(text):  #Function to speak and print text
    print()
    engine.say(text)
    print(text)
    engine.runAndWait()
```

```
print("\033[1;32;47m                    Welcome to ")
engine.say('Welcome to')
engine.runAndWait()
engine.setProperty('voice', voices[0].id)
time.sleep(2)
logo=''' ________    _______              _________   _________
    |      |      |     |      |     |          |        |
    |      |      |     |      |     |______     |        |
    |      |      |     |------|     |          |_________|
    |      |      |     |      |     |          |        |
    | .    |_______| .  |      | .   |    .     |        | .'''

logo1='''                    T - Teacher
                    O - Oriented
                    H - Helping (and)
                    F - Functioning
                    A - Assistant'''
print(logo)
engine.say('TOHFA')
engine.runAndWait()
print()
time.sleep(1)
print("\033[1;31;47m",logo1)
engine.setProperty('voice', voices[1].id)
engine.say('Teacher Oriented Helping and Functioning assistant.')
engine.runAndWait()
print()
print("Who do you want to talk to? Nova, the male assistant or Starfire, the female assistant?")
print()
while True:
   ask=input("Enter 'n' for Nova, 's' for Starfire: ")
   if ask.upper()[0]=='N':
         engine.setProperty('voice', voices[0].id)
         break
   elif ask.upper()[0]=='S':
         engine.setProperty('voice', voices[1].id)
         break
   else:
      print("Invalid entry. Try again: ")
      continue
time.sleep(0.5)
#Creating/Using binary file to store user data
f1=open('USERDATA','ab')
p.dump('\n',f1)
```

```
f1.close()
#Setting up MySQL connection
speech_out('Enter name of host of your MySQL connection')
host_name=input("Enter: ")
speech_out('Enter name of user of your MySQL connection')
user_name=input("Enter: ")
speech_out('Enter password of your MySQL connection')
password_SQL=input("Enter: ")

db=m.connect(host=host_name,user=user_name,password=password_SQL)
if db.is_connected():
   pass
else:
   speech_out('Invalid MySQL connection. QUITTING PROGRAM.....')
   sys.exit()
   time.sleep(2)
c=db.cursor()
password=''

def NewUser():   #Function to create username-password for new user
      global username,password
      while True:
         username=input('Enter username: ')
         f1=open('USERDATA','rb+')
         f1.seek(0)
         i=p.load(f1)
         userstatus=True
         try:
            while True:
               if username in i:
                  speech_out('Username Already Exists. Try Again')
                  userstatus=False
                  break
               i=p.load(f1)
         except EOFError:
            pass

         if userstatus==False:
            continue
         elif userstatus==True:
            pass
         print()
         #Creating Tkinter textbox to acquire password
         master = Tk()
```

```
          def save():
             password = Password.get()
             master.destroy()
             return password
          Label1 = Label(master, text="For username '"+ username+"', enter password and \n press the
'submit password' button.")
          Label1.pack()
          Password = Entry(master, bd=5, width=20, show="*")
          Password.pack()
          Button1 = Button(master, text='Submit Password', command=save)
          Button1.pack()
          master.mainloop()
          p.dump(username+':'+password,f1)
          f1.close()
          break

def ExistUser():   #Function to accept username-password of existing user
       global username,user_status,password
       username=input('Enter your username: ')
       print()
       #Creating Tkinter textbox to acqiuire password
       master = Tk()
       def save():
          password = Password.get()
          master.destroy()
          return password
       Label1 = Label(master, text="For username '"+ username+"', enter password and \n press the
'submit password' button.")
       Label1.pack()
       Password = Entry(master, bd=5, width=20, show="*")
       Password.pack()
       Button1 = Button(master, text='Submit Password', command=save)
       Button1.pack()
       master.mainloop()
       #Checking user details
       f1=open('USERDATA','rb')
       f1.seek(0)
       i=p.load(f1)
       userstatus=False
       passstatus=False
       try:
          while i:
             if username+':'+password==i:
                userstatus=True
                passstatus=True
                break
            i=p.load(f1)
        except EOFError:
            pass
        f1.close()
        #If details wrong, offering options to exercise
        if userstatus!=True or passstatus!=True:
            print()
            speech_out('Invalid username or password')
            speech_out('Would you like to create a new account or enter your username and password again
or Exit?')
            choice=input("Type  'New' for new, 'Try again' for trying again or 'Exit' to exit: ")
            if choice.upper()=='NEW':
                user_status='NEW'
                NewUser()
            elif choice.upper()=='TRY AGAIN':
                user_status='EXISTING'
                ExistUser()
            else:
                speech_out('QUITTING PROGRAM ...........')
                sys.exit()
                time.sleep(2)

speech_out('New User or Existing User? ')
user_status=input("Enter 'n' for new, 'e' for existing user: ")
print()

if user_status.upper()=='N':
    NewUser()
else:
    ExistUser()

speech_out('Logging on to '+username+'....')
time.sleep(2)
if user_status.upper()[0]=='N':
    c.execute('create database '+username) #Setting database as user name
    c.execute('commit;')
    c.execute('use '+username)
else:
    c.execute('use '+username) #Setting database as user name

def Speech():  #Function to accept speech
    r=sr.Recognizer()
    with sr.Microphone() as source:
        print('Speak >')
        audio=r.listen(source)
    try:
        return r.recognize_google(audio)
    except:
        pass
dummy=''
d={}
dm={}
n2=0
mark_dict={}

def create_Attendance(table_name):   #Function to create table for attendance
    speech_out('Enter number of students in the class: ')
    n=int(input("Enter (in numbers): "))
    command='create table '+table_name+'\n(\n Date date primary key,'
    for i in range(1,n+1):
        line='\nRN'+str(i)+' int ,'
        command+=line
    command=command.rstrip(',')
    command=command+');'
    c.execute(command)
    c.execute('commit;')
    speech_out('Table created.')

def Attendance(table_name):  #Function to record attendance
    l=[]
    c.execute ("desc "+table_name+";")
    for i in c:
        l=l+[i[0]]
    n=len(l)-1
    speech_out('Do you want to enter attendance for today or previous date? ')
    choice=input("Enter 't' for today, any other key for any other day: ")
    if choice.upper()=='T':
        date=datetime.datetime.today().strftime('%Y-%m-%d')
    else:
        speech_out('Enter date in YYYY-MM-DD format: ')
        date=input("Enter: ")
    record={'Date':date}
    speech_out("Enter mode of registering.")
    mode=input("Press 's' for speech, or any other key for text input: ")
    a=1
    if mode.upper()=='S':    #Accepting attendance in speech
```

```
      while True:
         if a>n:
            speech_out('Roll number limit reached. Continue only to change existing attendance')
#Warning user about reaching max. possible entries

         speech_out('Enter roll number and be ready to speak or type EXIT to exit: ')
         roll_no=input("Enter: ")

         if roll_no.upper()=='EXIT':
            break

         elif not roll_no.isdigit():
            speech_out('Invalid roll number')
            continue
         roll_no='RN'+roll_no

         text=Speech()
         if text==None:
            speech_out('Invalid Entry. Try again.')
            continue
         text=text.split()
         status=-1
         for i in text:
            if i.upper()=='PRESENT':
               status=1
            if i.upper()=='ABSENT':
               status=0
         if status==-1:
            speech_out('Invalid Entry. Please try again.')
            continue
         record[roll_no]=status #Storing attendance in dictionary
         a+=1
   else:
      while True:  #Accepting attendance in text
         if a>n:
            speech_out('Roll number limit reached. Continue only to change existing attendance')

         speech_out('Enter roll number or type EXIT to exit: ')
         roll_no=input("Enter: ")
         if roll_no.upper()=='EXIT':
            break
         elif not roll_no.isdigit():
            speech_out('Invalid roll number')
            continue
```

```
        roll_no='RN'+roll_no

        text=input('Enter status. P/A: ')
        if text.upper()[0]=='P':
          status=1
        elif text.upper()[0]=='A':
          status=0
        else:
          speech_out('Invalid Entry. Try Again')
          continue

        record[roll_no]=status #Storing attendance in dictionary
        a+=1
    record=list(record.values())
    command='insert into '+table_name+' values\n('
    for j in record:
      if type(j)==str:
        command=command+'\''+j+'\','
      else:
        command=command+str(j)+','
    command=command.rstrip(',')
    command+=');'
    c.execute(command)
    c.execute('commit;')
    speech_out('Attendance Stored.')

def markstable_name():# To create table name for a table storing marks
    global dummy
    global n2
    speech_out("Enter the standard (eg 10 for tenth standard): ")
    standard=input("Enter: ")
    speech_out("Enter the section: ")
    section=input("Enter: ")
    speech_out("Enter the exam (eg. 'Half yearly' for half yearly exams, 'weekly 1' for first weekly exams,
etc.): ")
    exam1=input("Enter: ")
    examl=exam1.split()
    exam=''
    for i in range(len(examl)):
      if i!=len(examl)-1:
        exam+=examl[i]+'_'
      else:
        exam+=examl[i]
    speech_out("Enter the year in YYYY-YY format (eg. 2018-19): ")
```

```
ay1=input("Enter: ")
check=False
l=ay1.split('-')
while check==False:
    if len(l[0])==4:
        if l[0].isdigit():
            check=True
            break
        else:
            a=l[0].isdigit()
            while a==False:
                speech_out("Error in starting year.")
                speech_out("Enter starting year in YYYY format: ")
                v=input("Enter: ")
                l[0]=v
                if len(v)!=4:
                    speech_out("Error in format. Try again.")
                    continue
                a=l[0].isdigit()
    else:
        speech_out("Error: Invalid length of starting year entered.")
        speech_out("Enter starting year in YYYY format: ")
        v=input("Enter: ")
        l[0]=v
        continue
check=False
while check==False:
    if len(l[1])==2:
        if l[1].isdigit():
            check=True
            break
        else:
            a=l[1].isdigit()
            while a==False:
                speech_out("Error in ending year.")
                speech_out("Enter ending year in YY format:")
                v=input("Enter: ")
                l[1]=v
                if len(v)!=2:
                    speech_out("Error in format. Try again.")
                    continue
                a=l[1].isdigit()
    else:
        speech_out("Error: Invalid length of ending year entered. Enter again")
```

```
            speech_out("Enter ending year in YY format: ")
            v=input("Enter: ")
            l[1]=v
            continue
    ay=''
    for i in range(2):
        if i==0:
            ay=ay+l[i]+'_'
        elif i==1:
            ay=ay+l[i]
    table_name=standard+'_'+section+'_'+exam+'_'+ay
    return table_name

def mark_entry(mark_table):#For entering marks into a created table
    global d,dm
    global mark_dict
    l=[]
    c.execute ("desc "+mark_table+";")
    for i in c:
        l=l+[i[0]]
    l=l[1:]
    for i in range(len(l)):
        d[i+1]=l[i]
    speech_out("Enter the number of students for whom the marks are to be entered.")
    n=int(input("Enter: "))
    for i in range(n):
        speech_out("Enter roll no: ")
        rollno=int(input("Enter: "))
        confirm=True
        while confirm!="NO" or confirm!="no" or confirm!="No" or confirm!="nO" or confirm!="":
            lm=[]

            for i in range(len(l)):
                print(d)
                print("For subject",i+1,": ")
                marks=int(input("Enter marks: "))
                try:
                    while marks>dm[i]:
                        print('Maximum marks: ',dm[i])
                        speech_out("Error: Marks entered are more than maximum marks.")
                        marks=int(input("Please enter valid marks: "))
                except KeyError:
                        pass
                lm=lm+[marks]
            print(lm)
            speech_out("Above are the marks subject wise.")
            speech_out("Confirm? (Reply with Yes or No)")
            confirm=input("Your reply: ")
            if confirm=='NO' or confirm=='no' or confirm=='No':
                ask=input("Change marks for one subject, change marks for all or confirm?")
                if ask=="One" or "one" or "ONE":
                    sno=int(input("Enter subject number: "))
                    marks=int(input("Enter marks: "))
                    lm[sno-1]=int(marks)
                    print(lm)
                    break
                elif ask=='ALL' or 'all' or 'All':
                    continue
                else:
                    break
            else:
                break
            mark_dict[rollno]=lm
        m1=''
        for i in range(len(lm)):
            if i==len(lm)-1:
                m1=m1+str(lm[i])
            else:
                m1=m1+str(lm[i])+','
        c.execute("insert into "+mark_table+" values\n"+'('+str(rollno)+','+m1+");")
    c.execute("commit;")

def markstable(table_name):#To create a table which will store marks
    global dummy
    global n2
    global dm
    c.execute("show tables;")
    for i in c.fetchall():
        j=str(i)
        k=j.strip(",;()'"")
        if table_name==k:
            print("Table by the name '",table_name,"' already exists.")
            speech_out("Try again")
            break
    else:
        create="create table "+table_name+"\n(Roll_No int(2) primary key);"
        c.execute(create)
    a='a'
    while a=='a' or a=='A':
      speech_out("Enter the number of subjects to be added: ")
      n=int(input("Enter: "))
      for i in range(n):
        print("For subject",i+1,":")
        print()
        print("For subject name (In one word only): ")
        speech_out("Press 's' key and enter for speaking or press any other key and enter for manually
entering subject name.")
        opt=input("Enter your choice: ")
        if opt=='s' or opt=='S':
          dummy=input("Press enter when ready to speak.")
          s=str(Speech())
          if s=='None':
            speech_out("Speech wasn't recognized. Sorry for the inconvinience.")
            s=input("Enter subject name (through keyboard) (In one word only): ")
            s.lower()
          else:
             s=s.lower()
        else:
          speech_out("Enter subject name (In one word only): ")
          s=input("Enter: ")
        speech_out("Enter maximum marks: ")
        mm=int(input("Enter: "))
        dm[i]=mm
        d[n2+i+1]=s
        print(d)
        q='alter table '+table_name+' add\n('+s+' int);'
        c.execute(q)
      speech_out("If more columns are to be added, press 'a' and enter key. Else, press any other key
to finalize the table.")
      a=input("Enter: ")
      n2=n2+n
    c.execute('commit;')
    speech_out("Table created.")

def attendance_tablename(): #Function to create name for table storing attendance
  speech_out("Enter the standard (example: 10 for tenth standard)")
  standard=input("Enter: ")
  speech_out("Enter the section: ")
  section=input("Enter: ")
  speech_out("Enter the year in YYYY-YY format (eg. 2018-19): ")
  ay1=input("Enter: ")
```

```
check=False
l=ay1.split('-')
#Checking formatting of session duration
while check==False:
    if len(l[0])==4:
        if l[0].isdigit():
            check=True
            break
        else:
            a=l[0].isdigit()
            while a==False:
                speech_out("Error in starting year.")
                speech_out("Enter starting year in YYYY format: ")
                v=input("Enter: ")
                l[0]=v
                if len(v)!=4:
                    speech_out("Error in format. Try again.")
                    continue
                a=l[0].isdigit()
    else:
        speech_out("Error: Invalid length of starting year entered.")
        speech_out("Enter starting year in YYYY format: ")
        v=input("Enter: ")
        l[0]=v
        continue
check=False
while check==False:
    if len(l[1])==2:
        if l[1].isdigit():
            check=True
            break
        else:
            a=l[1].isdigit()
            while a==False:
                speech_out("Error in ending year.")
                speech_out("Enter ending year in YY format:")
                v=input("Enter: ")
                l[1]=v
                if len(v)!=2:
                    speech_out("Error in format. Try again.")
                    continue
                a=l[1].isdigit()
    else:
        speech_out("Error: Invalid length of ending year entered. Enter again")
```

```
            speech_out("Enter ending year in YY format:")
            v=input("Enter: ")
            l[1]=v
            continue
    ay=''
    for i in range(2):
        if i==0:
            ay=ay+l[i]+'_'
        elif i==1:
            ay=ay+l[i]
    table_name=standard+'_'+section+'_'+'attendance'+'_'+ay
    return table_name

def delete_table(table_name): #Function to delete a table
    c.execute("drop table "+table_name+";")
    print("Table '"+table_name+"' deleted successfully.")
    c.execute('commit;')

def deletemark(table_name): #Function to delete marks for a particular student
    l=[]
    c.execute("select * from "+table_name+";")
    speech_out("Enter roll no. for which record has to be deleted (eg. 1)")
    rollno=int(input("Enter: "))
    for i in c:
        l=l+[i[0]]
    for i in l:
        if i==rollno:
            c.execute("delete from "+table_name+" where Roll_No="+str(rollno)+";")
            speech_out("Record successfully deleted.")
            break
    else:
        speech_out("Roll No. is not a part of the table. Try again.")
    c.execute('commit;')

def del_Attendance(table_name): #Function to delete record from Attendance table
    n=int(input('Enter number of days whose records you want to delete:'))
    for i in range(n):
        date=input('Enter date to delete in YYYY-MM-DD format:') #Deletion according to date, as date is
primary key
        command='delete from '+table_name+' where Date=\''+date+'\';'
        c.execute(command)
        c.execute('commit;')
        print('Attendance for',date,'deleted successfully')
```

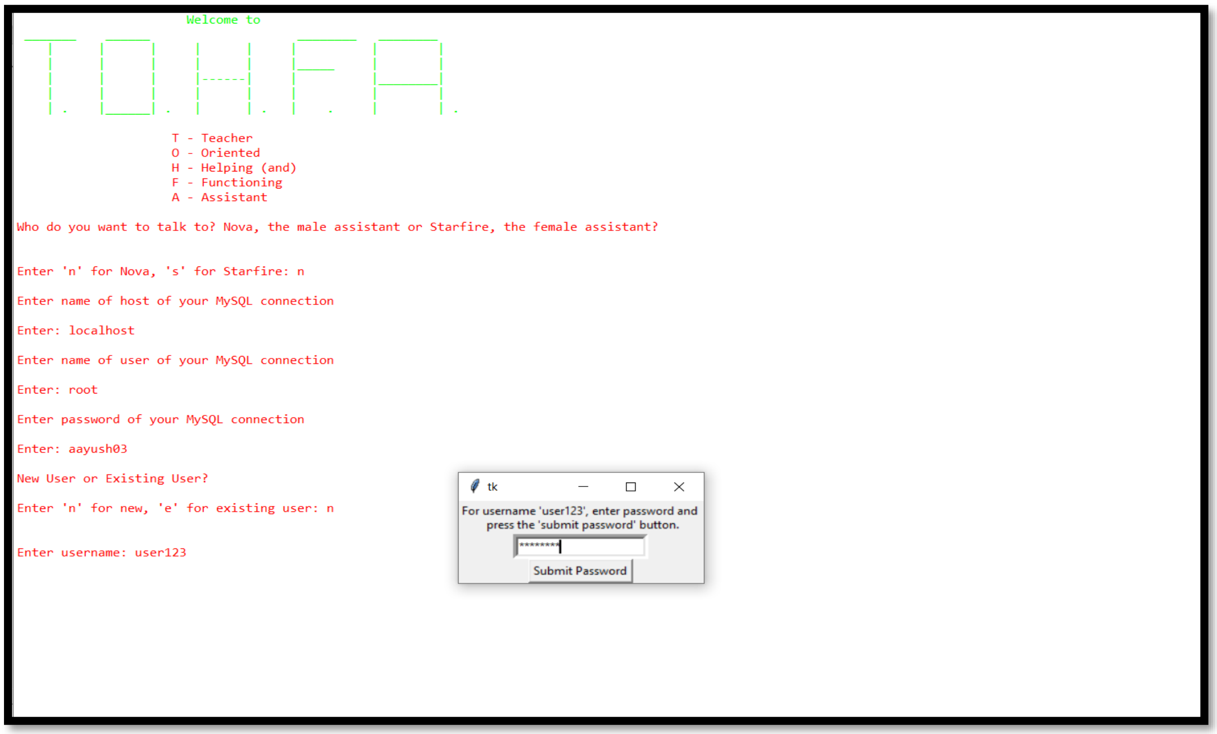
```
def displaymarks(table_name):#Function to display marks
    select1 = 'select*from '+table_name
    speech_out('''
Enter 1 to view the entire table
Enter 2 to view marks for a particular student
Enter 3 to view marks for a particular subject''')
    q=int(input("Enter choice number: "))

    if q == 1:#Display of entire table
        print('MARKS OF '+table_name)
        c.execute("desc "+table_name)
        l = []
        for i in c:
            l = l +[i[0]]
        c.execute(select1)
        print(l)
        for i in c:
            print(i)

    elif q==2:#Display of marks for a particular student
        c.execute("desc "+table_name)
        l = []
        for i in c:
            l = l +[i[0]]

        speech_out('Enter the Roll Number of the student')
        t = input("Enter: ")
        select2 = 'select*from '+ table_name + ' where Roll_No ='+t
        c.execute(select2)
        j = c.fetchall()
        print(j)
        print('MARKS OF ROLL NO. ',t)
        n = len(l)
        for i in range(1,n):
            t1 = l[i]
            t2 = j[0][i]
            speech_out(str(t2)+' marks have been secured in the subject '+str(t1))

        print('The average marks of Roll No.',t,'is',(sum(j[0])-int(t))/(len(j[0])-1))

    elif q==3:#Display of marks for a particular subject
        speech_out('Enter the subject whose marks is to be displayed')
        u=input("Enter: ")
        select3 = 'select ' +u+ ' from '+ table_name
        select4 = 'select Roll_No from '+table_name+';'
        c.execute(select4)
        l=[]
        for i in c.fetchall():
            l=l+[i[0]]
        print(l)
        c.execute(select3)
        j= c.fetchall()
        listm = []
        print('MARKS OF THE SUBJECT ',u)
        for i in range(len(j)):
            k = j[i]
            k1 = str(k)
            k2 = k1.strip("(),")
            speech_out('ROLLNO.  '+str(l[i])+' has secured '+str(k2)+' marks')
            listm.append(int(k2))
        print('THE AVERAGE MARKS IS',sum(listm)/len(listm))

def displayattendance(table_name):#Function to display attendance
    select1 = 'select*from '+ table_name
    speech_out('''
Enter 1 to view the entire table
Enter 2 to view attendance for a particular date
Enter 3 to view attendance for a particular student''')
    speech_out("Enter choice number")
    q=int(input("Enter: "))

    if q == 1:#Display of entire table
        speech_out('ATTENDACE OF '+str(table_name))
        c.execute(select1)
        for i in c:
            print(i)

    elif q==2:#Display of attendance on a particular date
        date1 = input('Enter the date in the format of YYYY-MM-DD')
        select2 = 'select*from '+table_name+' where date = ' '\'' +date1+ '\'';'
        speech_out('ATTENDANCE ON '+ str(date1))
        c.execute(select2)
        j = c.fetchall()
        print(j)
        count1 = 0
        count2 = 0
        l = len(j[0])
        for i in range(1,l):
            if j[0][i] == 1:
                count1 = count1+1
        speech_out('THE TOTAL NUMBER OF STUDENTS PRESENT IS '+str(count1)+', THE NUMBER OF
ABSENTEES IS '+str(l-count1-1)+'OUT OF A TOTAL OF '+str(l-1)+'STUDENTS')
    elif q==3:#Display of attendance for a particular student
        speech_out('Enter the roll number of the student')
        t=input("Enter: ")
        speech_out('THE ATTENDANCE OF ROLL NO '+str(t)+' IS')
        select3 = 'select RN'+t+ ' from ' + table_name

        #select3 is for the attendance
        c.execute(select3)
        j = c.fetchall()

        count1 = 0
        count2 = 0
        for i in j:
            count1 = count1 + 1
        for i in j:
            if i == (1,):
                count2 = count2 + 1
        print(j)
        speech_out('ROLL NO '+str(t)+' HAS BEEN PRESENT FOR '+str(count2)+' DAYS OUT OF
'+str(count1)+' DAYS' )

optlist='''List of operations:
1. Creation of table to store marks
2. Entering marks into an already created table
3. Deleteting a table storing marks
4. Deleting marks for a particular student
5. Creation of table to store attendance
6. Marking attendance into an already created table
7. Deleting a table storing attendance
8. Deleting attendance for a particular date
9. Enquiry regarding marks
10. Enquiry regarding attendance'''
speech_out(optlist)
choice='1'
while choice in ['1','2','3','4','5','6','7','8','9','10']:
    choice=input("Enter choice number or any other key to exit: ")
    if choice=='1':
        table_name=markstable_name() #Generating table name to work on
        markstable(table_name)
        print(optlist)
```

```
   print()
   continue

if choice=='2':
   table_name=markstable_name() #Generating table name to work on
   #Checking if table exists in database. If exists, function is called
   c.execute("show tables;")
   for i in c.fetchall():
       j=str(i)
       k=j.strip(",;()'"")
       if table_name==k:
          mark_entry(table_name)
          print("Marks stored into table",table_name,".")
          print(optlist)
          print()
          break
   else:
      speech_out("Table name doesn't exist.")
      speech_out("Enter 'n' to create new table, press any other key to exit this option.")
      choice1=input("Enter: ")
      if choice1=='n' or choice1=='N':
         markstable(table_name)
         speech_out("Table created.")
         print(optlist)
         print()
         continue
      else:
         print(optlist)
         print()
   continue

if choice=='3':
   table_name=markstable_name() #Generating table name to work on
   #Checking if table exists in database. If exists, function is called
   c.execute("show tables;")
   for i in c.fetchall():
       j=str(i)
       k=j.strip(",;()'"")
       if table_name==k:
          print("You are about to delete: "+table_name)
          speech_out("Press 'y' to confirm deletion, press any other key to cancel it.")
          choice2=input("Enter: ")
          if choice2[0]=='y' or choice2[0]=='Y':
             delete_table(table_name)
```

```
                print(optlist)
                print()
                break
            else:
                speech_out("Action cancelled.")
                print(optlist)
                print()
                break
    else:
        print("Table '"+table_name+"' doesn't exist.")
        print(optlist)
        print()
    continue

if choice=='4':
    table_name=markstable_name() #Generating table name to work on
    #Checking if table exists in database. If exists, function is called
    c.execute("show tables;")
    for i in c.fetchall():
         j=str(i)
         k=j.strip(",;()''")
         if table_name==k:
             deletemark(table_name)
             break
    else:
       print("Table '"+table_name+"' doesn't exist.")
    print(optlist)
    print()
    continue

if choice=='5':
    table_name=attendance_tablename() #Generating table name to work on
    #Checking if table exists in database. If not existant, function is called
    c.execute("show tables;")
    for i in c.fetchall():
         j=str(i)
         k=j.strip(",;()''")
         if table_name==k:
             print("Table by the name '",table_name,"' already exists. Try again")
             break
    else:
        create_Attendance(table_name)
        print("Table '"+table_name+"' created.")
        print(optlist)
        print()
    continue

if choice=='6':
    table_name=attendance_tablename() #Generating table name to work on
    #Checking if table exists in database. If exists, function is called
    c.execute("show tables;")
    for i in c.fetchall():
        j=str(i)
        k=j.strip(",;()'"")
        if table_name==k:
            Attendance(table_name)
            print(optlist)
            print()
            break
    else:
        print("Table '"+table_name+"' doesn't exist.")
        print(optlist)
        print()
    continue

if choice=='7':
    table_name=attendance_tablename() #Generating table name to work on
    #Checking if table exists in database. If exists, function is called
    c.execute("show tables;")
    for i in c.fetchall():
        j=str(i)
        k=j.strip(",;()'"")
        if table_name==k:
            print("You are about to delete: "+table_name)
            speech_out("Press 'y' to confirm deletion, press any other key to cancel it.")
            choice2=input("Enter: ")
            print()
            if choice2=='y' or choice2=='Y':
                delete_table(table_name)
                print(optlist)
                print()
                break
            else:
                speech_out("Action cancelled.")
                print(optlist)
                print()
                break
    else:
        print("Table '"+table_name+"' doesn't exist.")
        print(optlist)
        print()
    continue

if choice=='8':
    table_name=attendance_tablename() #Generating table name to work on
    #Checking if table exists in database. If exists, function is called
    c.execute("show tables;")
    for i in c.fetchall():
        j=str(i)
        k=j.strip(",;()'"")
        if table_name==k:
            del_Attendance(table_name)
            break
    else:
       print("Table '"+table_name+"' doesn't exist.")
    print(optlist)
    print()
    continue

if choice=='9':
    table_name=markstable_name() #Generating table name to work on
    #Checking if table exists in database. If exists, function is called
    c.execute("show tables;")
    for i in c.fetchall():
        j=str(i)
        k=j.strip(",;()'"")
        if table_name==k:
            displaymarks(table_name)
            break
    else:
       print("Table '"+table_name+"' doesn't exist.")
    print(optlist)
    print()
    continue

if choice=='10':
    table_name=attendance_tablename() #Generating table name to work on
    #Checking if table exists in database. If exists, function is called
    c.execute("show tables;")
    for i in c.fetchall():
        j=str(i)
        k=j.strip(",;()'"")
```

```
         if table_name==k:
             displayattendance(table_name)
             break
    else:
       print("Table '"+table_name+"' doesn't exist.")
    print(optlist)
    print()
    continue

else:
    speech_out("Thank you for using TOHFA!")
    break
```

OUTPUT SCREENS

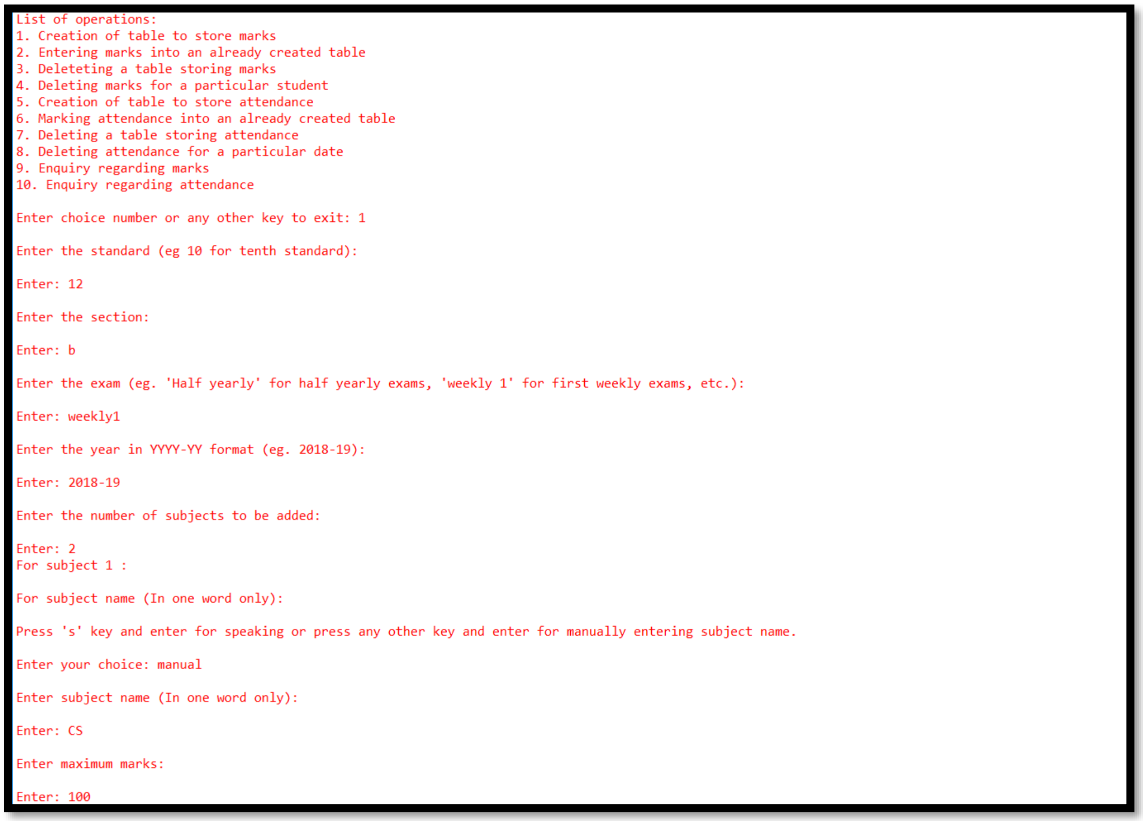


```
                       Welcome to
 _______    ______                    ________   ________
    |      |      |     |      |     |          |        |
    |      |      |     |      |     |_____     |        |
    |      |      |     |------|     |          |________|
    |      |      |     |      |     |          |        |
    | .    |______| .   |      | .   |    .     |        | .

                     T - Teacher
                     O - Oriented
                     H - Helping (and)
                     F - Functioning
                     A - Assistant

Who do you want to talk to? Nova, the male assistant or Starfire, the female assistant?


Enter 'n' for Nova, 's' for Starfire: n

Enter name of host of your MySQL connection

Enter: localhost

Enter name of user of your MySQL connection

Enter: root

Enter password of your MySQL connection

Enter: aayush03

New User or Existing User?

Enter 'n' for new, 'e' for existing user: n


Enter username: user123
```

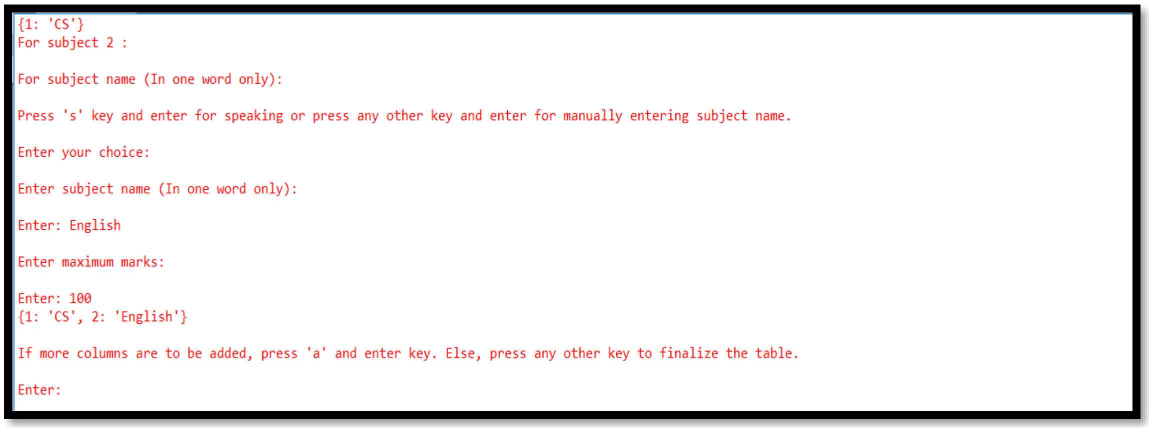
tk

For username 'user123', enter password and press the 'submit password' button.

Main Login Screen

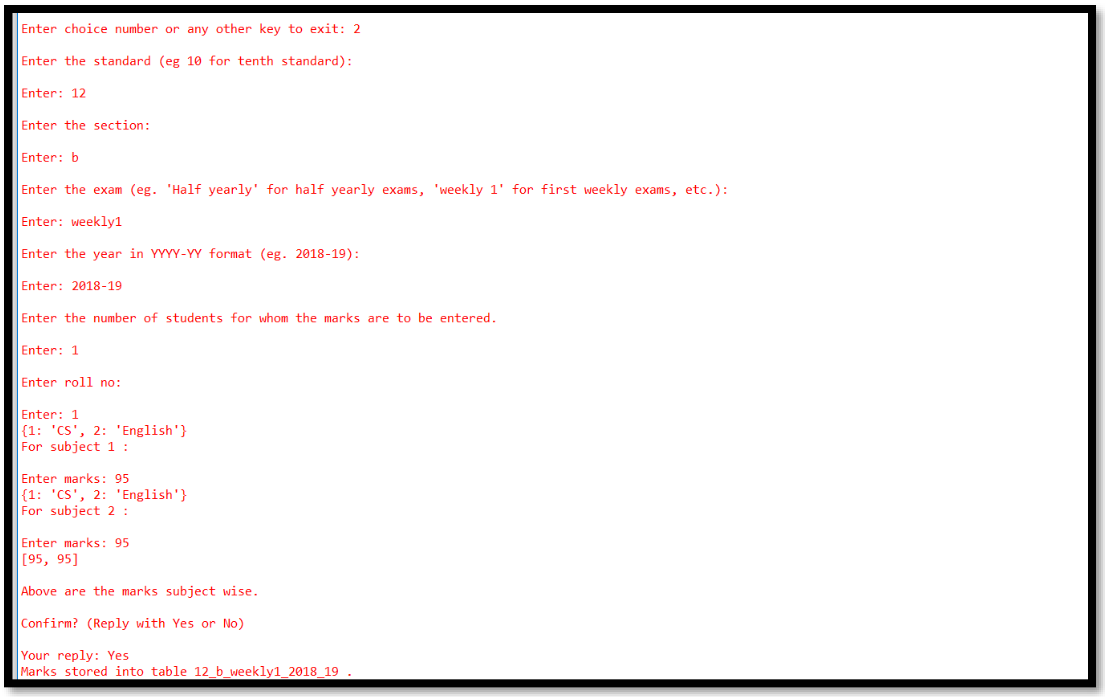


```
List of operations:
1. Creation of table to store marks
2. Entering marks into an already created table
3. Deleteting a table storing marks
4. Deleting marks for a particular student
5. Creation of table to store attendance
6. Marking attendance into an already created table
7. Deleting a table storing attendance
8. Deleting attendance for a particular date
9. Enquiry regarding marks
10. Enquiry regarding attendance

Enter choice number or any other key to exit: 1

Enter the standard (eg 10 for tenth standard):

Enter: 12

Enter the section:

Enter: b

Enter the exam (eg. 'Half yearly' for half yearly exams, 'weekly 1' for first weekly exams, etc.):

Enter: weekly1

Enter the year in YYYY-YY format (eg. 2018-19):

Enter: 2018-19

Enter the number of subjects to be added:

Enter: 2
For subject 1 :

For subject name (In one word only):

Press 's' key and enter for speaking or press any other key and enter for manually entering subject name.

Enter your choice: manual

Enter subject name (In one word only):

Enter: CS

Enter maximum marks:

Enter: 100
```



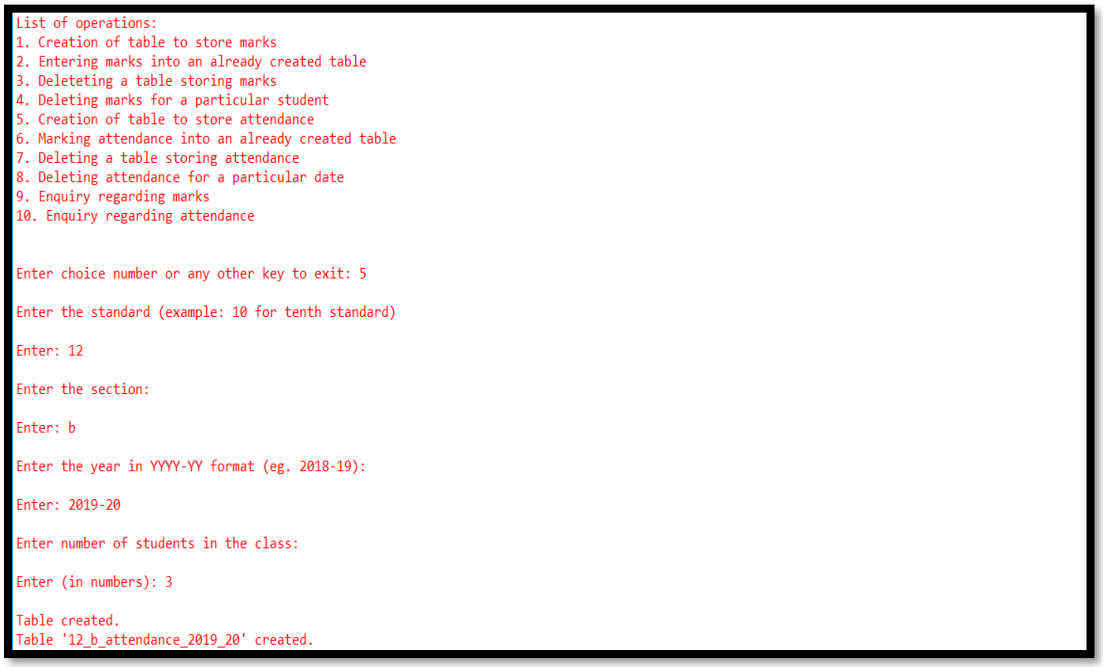
```
{1: 'CS'}
For subject 2 :

For subject name (In one word only):

Press 's' key and enter for speaking or press any other key and enter for manually entering subject name.

Enter your choice:

Enter subject name (In one word only):

Enter: English

Enter maximum marks:

Enter: 100
{1: 'CS', 2: 'English'}

If more columns are to be added, press 'a' and enter key. Else, press any other key to finalize the table.

Enter:
```

Option 1



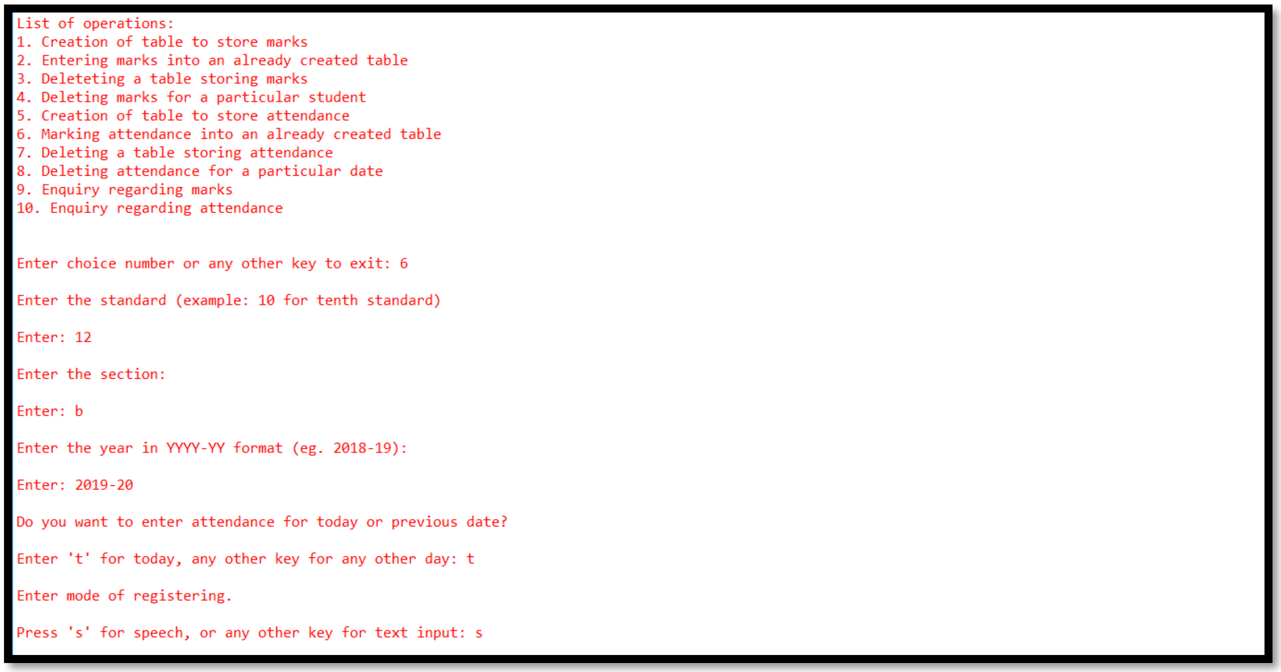
```
Enter choice number or any other key to exit: 2

Enter the standard (eg 10 for tenth standard):

Enter: 12

Enter the section:

Enter: b

Enter the exam (eg. 'Half yearly' for half yearly exams, 'weekly 1' for first weekly exams, etc.):

Enter: weekly1

Enter the year in YYYY-YY format (eg. 2018-19):

Enter: 2018-19

Enter the number of students for whom the marks are to be entered.

Enter: 1

Enter roll no:

Enter: 1
{1: 'CS', 2: 'English'}
For subject 1 :

Enter marks: 95
{1: 'CS', 2: 'English'}
For subject 2 :

Enter marks: 95
[95, 95]

Above are the marks subject wise.

Confirm? (Reply with Yes or No)

Your reply: Yes
Marks stored into table 12_b_weekly1_2018_19 .
```

Option 2

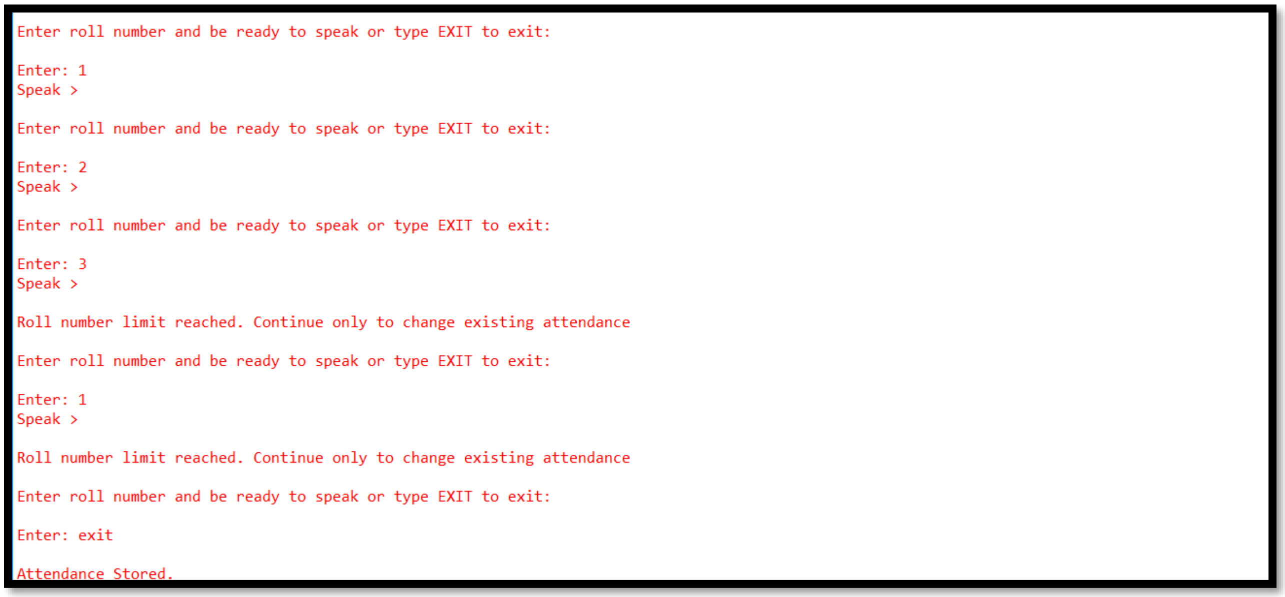


```
List of operations:
1. Creation of table to store marks
2. Entering marks into an already created table
3. Deleteting a table storing marks
4. Deleting marks for a particular student
5. Creation of table to store attendance
6. Marking attendance into an already created table
7. Deleting a table storing attendance
8. Deleting attendance for a particular date
9. Enquiry regarding marks
10. Enquiry regarding attendance


Enter choice number or any other key to exit: 5

Enter the standard (example: 10 for tenth standard)

Enter: 12

Enter the section:

Enter: b

Enter the year in YYYY-YY format (eg. 2018-19):

Enter: 2019-20

Enter number of students in the class:

Enter (in numbers): 3

Table created.
Table '12_b_attendance_2019_20' created.
```

Option 5

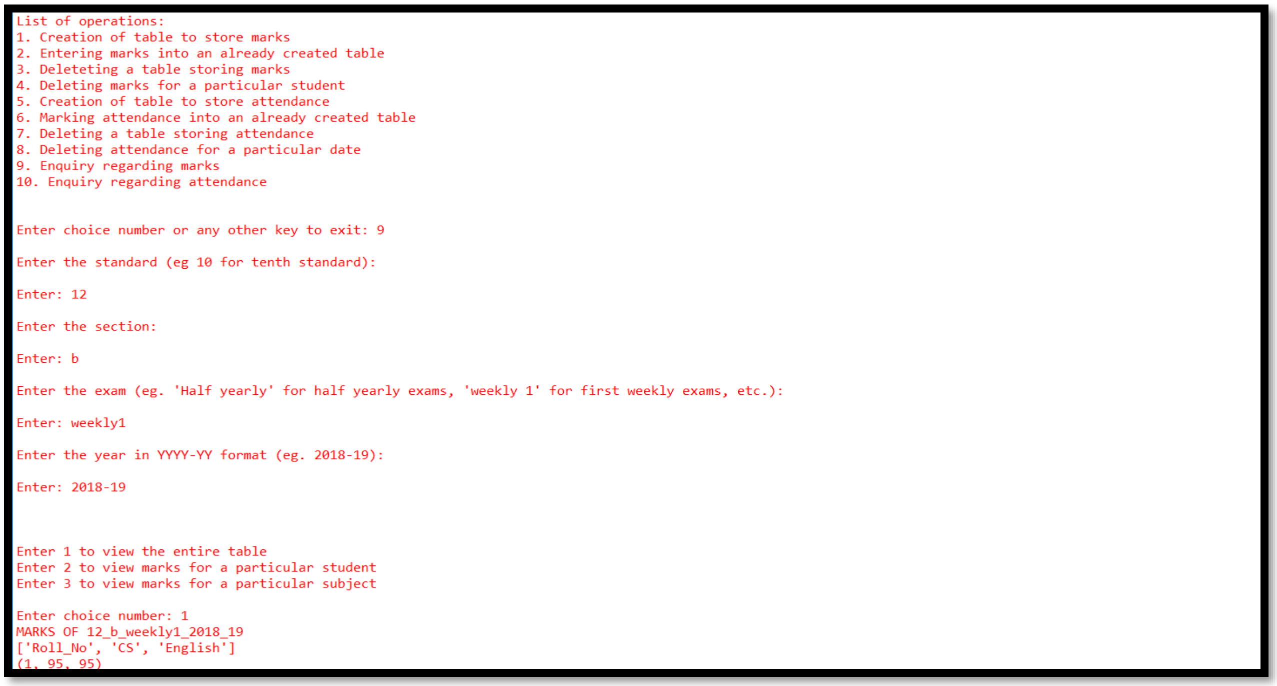


```
List of operations:
1. Creation of table to store marks
2. Entering marks into an already created table
3. Deleteting a table storing marks
4. Deleting marks for a particular student
5. Creation of table to store attendance
6. Marking attendance into an already created table
7. Deleting a table storing attendance
8. Deleting attendance for a particular date
9. Enquiry regarding marks
10. Enquiry regarding attendance


Enter choice number or any other key to exit: 6

Enter the standard (example: 10 for tenth standard)

Enter: 12

Enter the section:

Enter: b

Enter the year in YYYY-YY format (eg. 2018-19):

Enter: 2019-20

Do you want to enter attendance for today or previous date?

Enter 't' for today, any other key for any other day: t

Enter mode of registering.

Press 's' for speech, or any other key for text input: s
```

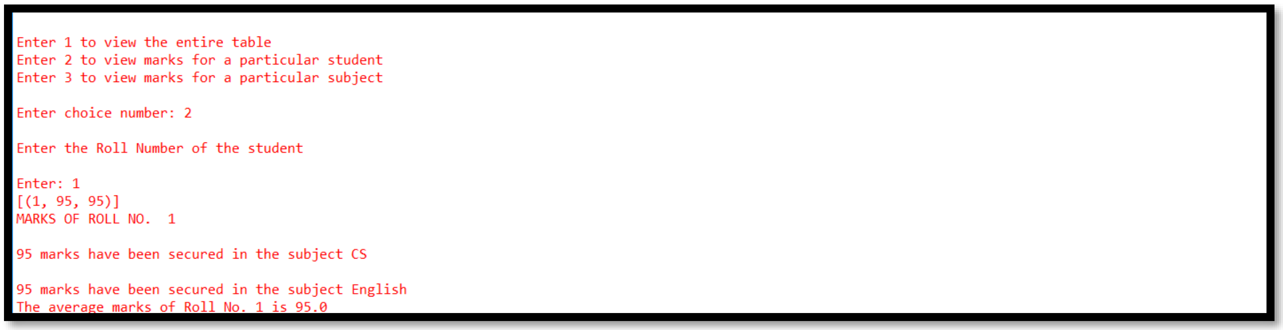


```
Enter roll number and be ready to speak or type EXIT to exit:

Enter: 1
Speak >

Enter roll number and be ready to speak or type EXIT to exit:

Enter: 2
Speak >

Enter roll number and be ready to speak or type EXIT to exit:

Enter: 3
Speak >

Roll number limit reached. Continue only to change existing attendance

Enter roll number and be ready to speak or type EXIT to exit:

Enter: 1
Speak >

Roll number limit reached. Continue only to change existing attendance

Enter roll number and be ready to speak or type EXIT to exit:

Enter: exit

Attendance Stored.
```

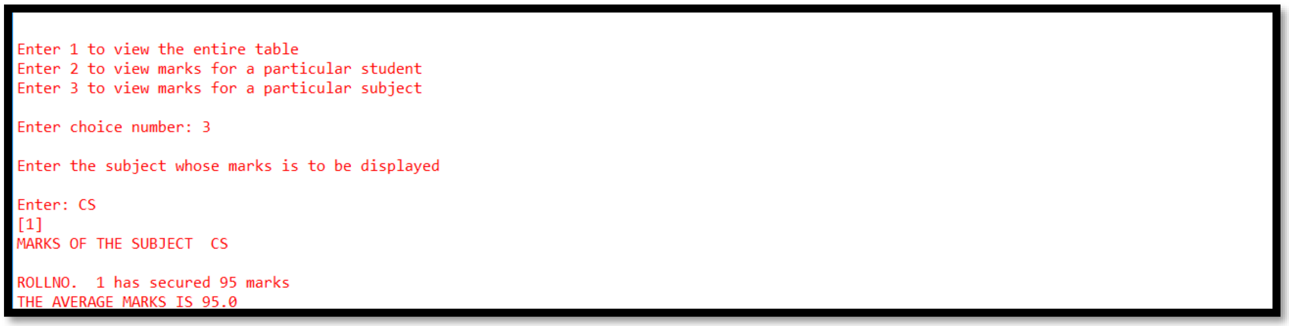
Option 6



```
List of operations:
1. Creation of table to store marks
2. Entering marks into an already created table
3. Deleteting a table storing marks
4. Deleting marks for a particular student
5. Creation of table to store attendance
6. Marking attendance into an already created table
7. Deleting a table storing attendance
8. Deleting attendance for a particular date
9. Enquiry regarding marks
10. Enquiry regarding attendance


Enter choice number or any other key to exit: 9

Enter the standard (eg 10 for tenth standard):

Enter: 12

Enter the section:

Enter: b

Enter the exam (eg. 'Half yearly' for half yearly exams, 'weekly 1' for first weekly exams, etc.):

Enter: weekly1

Enter the year in YYYY-YY format (eg. 2018-19):

Enter: 2018-19



Enter 1 to view the entire table
Enter 2 to view marks for a particular student
Enter 3 to view marks for a particular subject

Enter choice number: 1
MARKS OF 12_b_weekly1_2018_19
['Roll_No', 'CS', 'English']
(1, 95, 95)
```

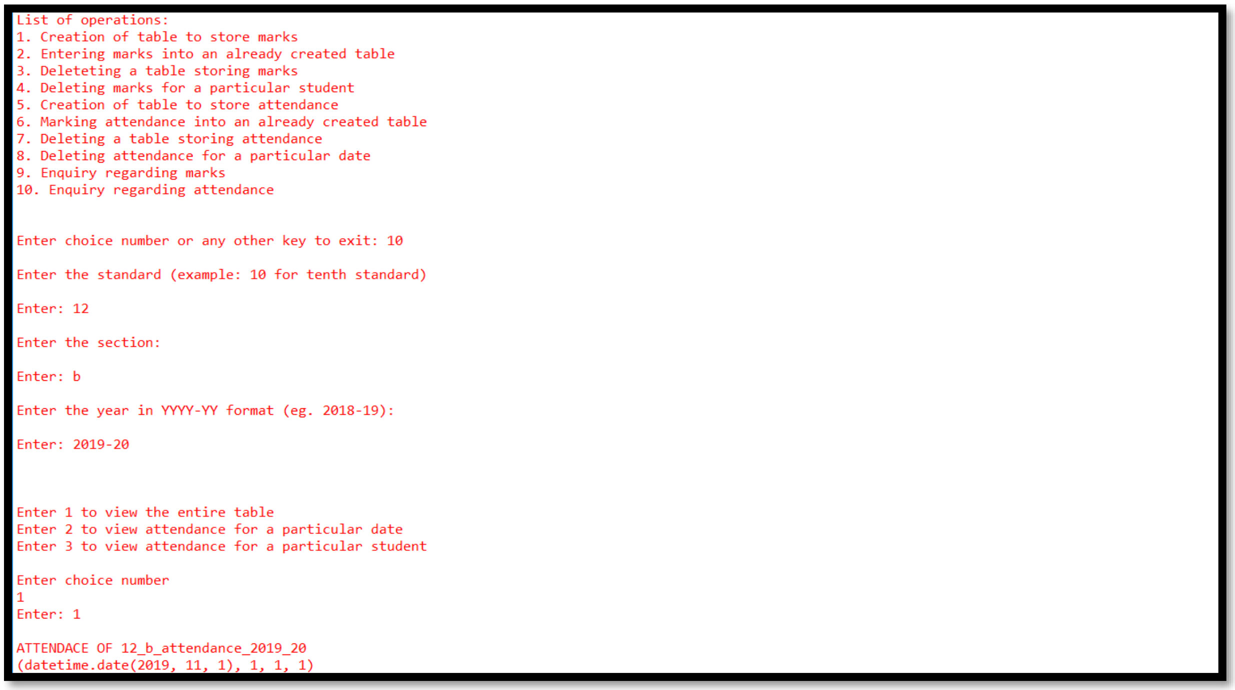


```
Enter 1 to view the entire table
Enter 2 to view marks for a particular student
Enter 3 to view marks for a particular subject

Enter choice number: 2

Enter the Roll Number of the student

Enter: 1
[(1, 95, 95)]
MARKS OF ROLL NO.  1

95 marks have been secured in the subject CS

95 marks have been secured in the subject English
The average marks of Roll No. 1 is 95.0
```

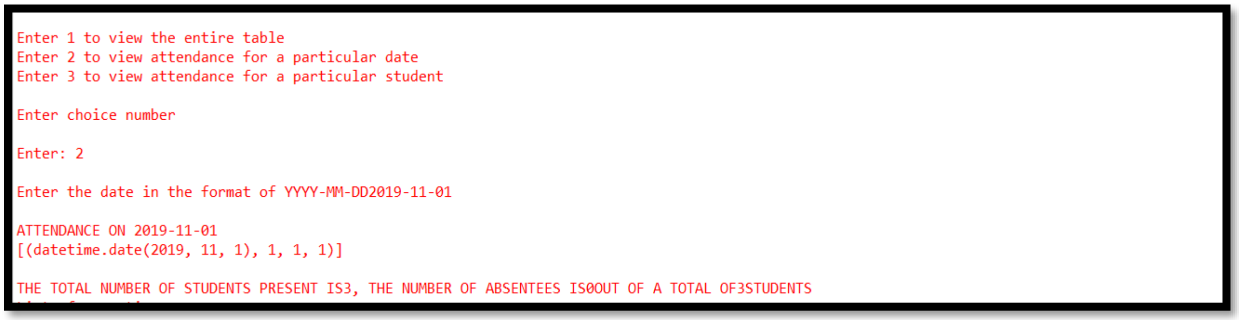


```
Enter 1 to view the entire table
Enter 2 to view marks for a particular student
Enter 3 to view marks for a particular subject

Enter choice number: 3

Enter the subject whose marks is to be displayed

Enter: CS
[1]
MARKS OF THE SUBJECT  CS

ROLLNO.  1 has secured 95 marks
THE AVERAGE MARKS IS 95.0
```

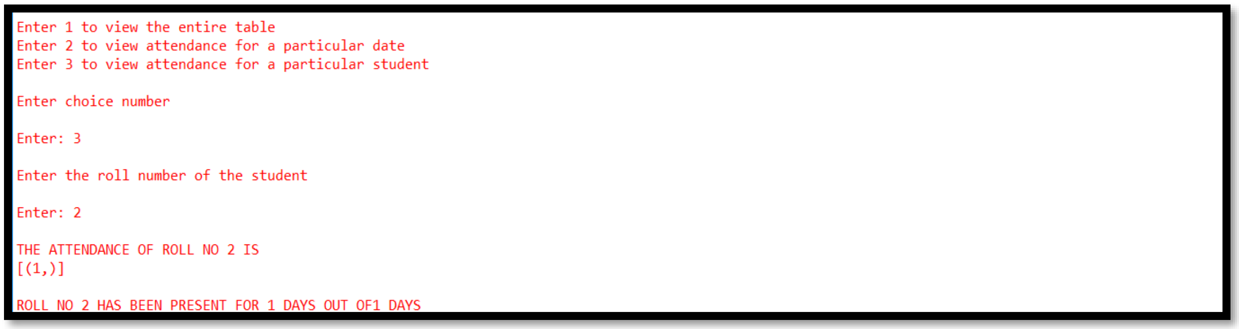
Options 9.1,9.2 and 9.3 respectively



```
List of operations:
1. Creation of table to store marks
2. Entering marks into an already created table
3. Deleteting a table storing marks
4. Deleting marks for a particular student
5. Creation of table to store attendance
6. Marking attendance into an already created table
7. Deleting a table storing attendance
8. Deleting attendance for a particular date
9. Enquiry regarding marks
10. Enquiry regarding attendance


Enter choice number or any other key to exit: 10

Enter the standard (example: 10 for tenth standard)

Enter: 12

Enter the section:

Enter: b

Enter the year in YYYY-YY format (eg. 2018-19):

Enter: 2019-20



Enter 1 to view the entire table
Enter 2 to view attendance for a particular date
Enter 3 to view attendance for a particular student

Enter choice number
1
Enter: 1

ATTENDACE OF 12_b_attendance_2019_20
(datetime.date(2019, 11, 1), 1, 1, 1)
```

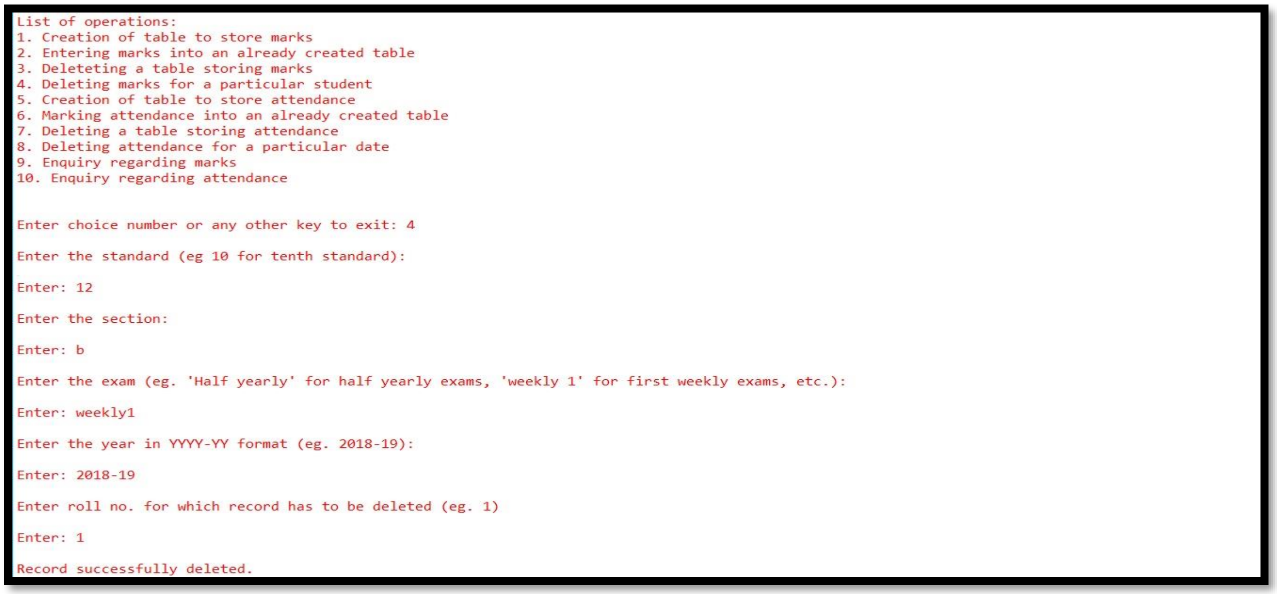


```
Enter 1 to view the entire table
Enter 2 to view attendance for a particular date
Enter 3 to view attendance for a particular student

Enter choice number

Enter: 2

Enter the date in the format of YYYY-MM-DD2019-11-01

ATTENDANCE ON 2019-11-01
[(datetime.date(2019, 11, 1), 1, 1, 1)]

THE TOTAL NUMBER OF STUDENTS PRESENT IS3, THE NUMBER OF ABSENTEES IS0OUT OF A TOTAL OF3STUDENTS
```



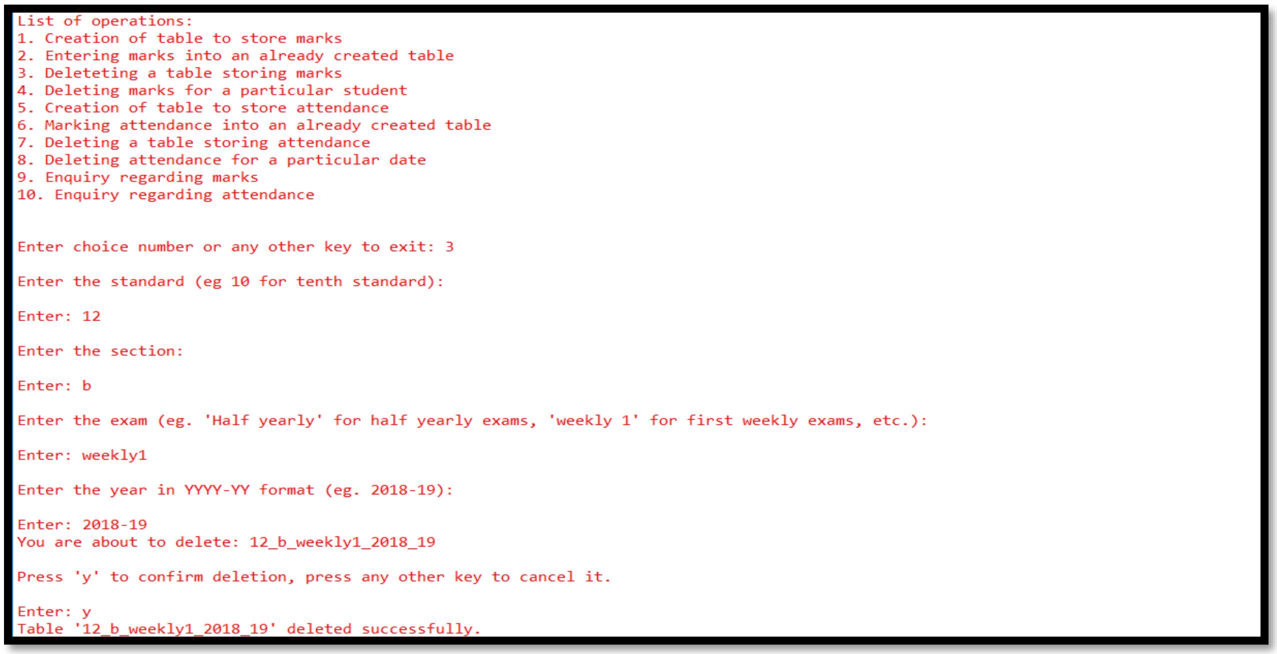
```
Enter 1 to view the entire table
Enter 2 to view attendance for a particular date
Enter 3 to view attendance for a particular student

Enter choice number

Enter: 3

Enter the roll number of the student

Enter: 2

THE ATTENDANCE OF ROLL NO 2 IS
[(1,)]

ROLL NO 2 HAS BEEN PRESENT FOR 1 DAYS OUT OF1 DAYS
```

Options 10.1,10.2 and 10.3 respectively



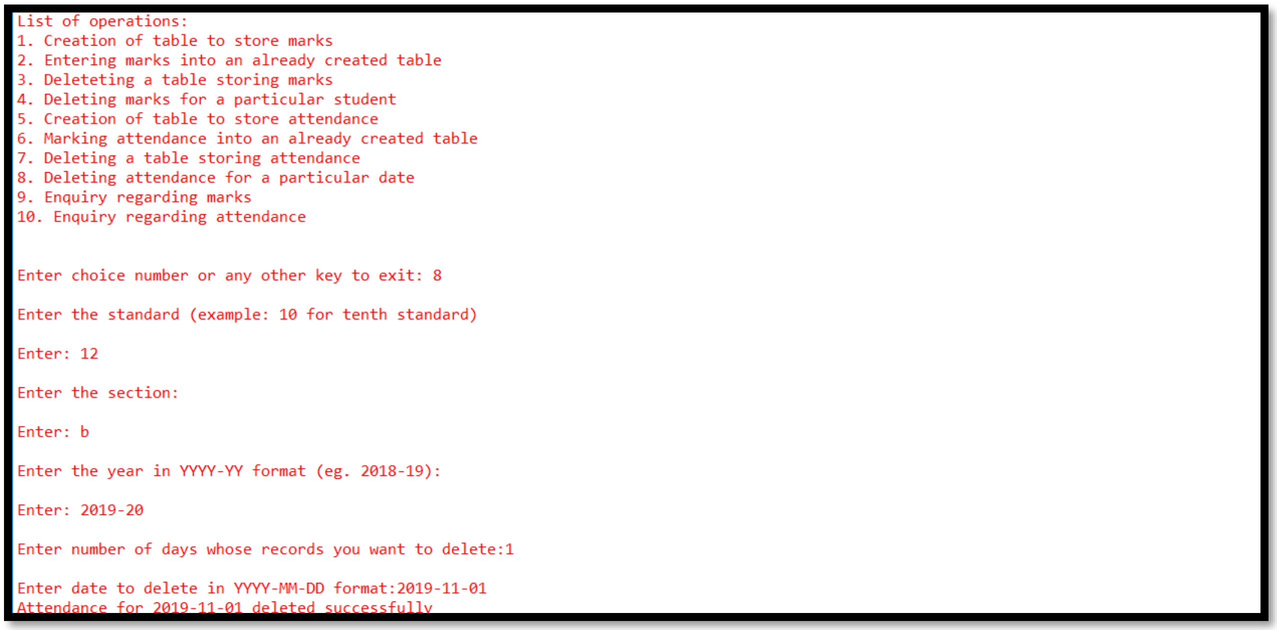
```
List of operations:
1. Creation of table to store marks
2. Entering marks into an already created table
3. Deleteting a table storing marks
4. Deleting marks for a particular student
5. Creation of table to store attendance
6. Marking attendance into an already created table
7. Deleting a table storing attendance
8. Deleting attendance for a particular date
9. Enquiry regarding marks
10. Enquiry regarding attendance


Enter choice number or any other key to exit: 4

Enter the standard (eg 10 for tenth standard):

Enter: 12

Enter the section:

Enter: b

Enter the exam (eg. 'Half yearly' for half yearly exams, 'weekly 1' for first weekly exams, etc.):

Enter: weekly1

Enter the year in YYYY-YY format (eg. 2018-19):

Enter: 2018-19

Enter roll no. for which record has to be deleted (eg. 1)

Enter: 1

Record successfully deleted.
```

Option 4



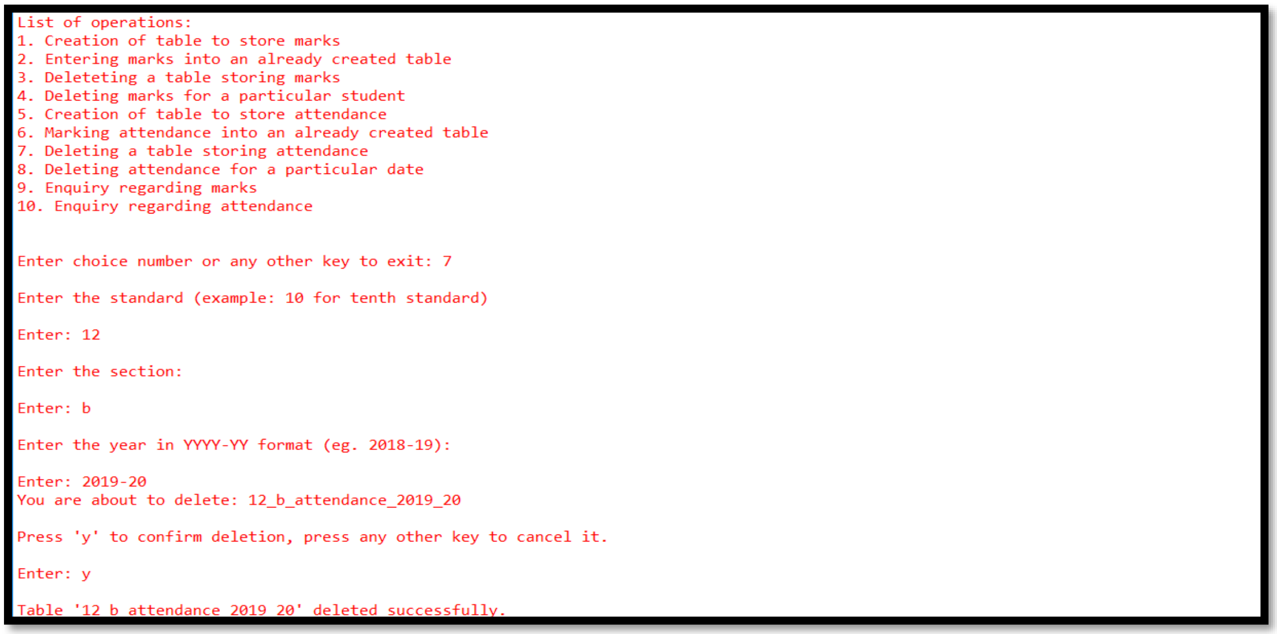
```
List of operations:
1. Creation of table to store marks
2. Entering marks into an already created table
3. Deleteting a table storing marks
4. Deleting marks for a particular student
5. Creation of table to store attendance
6. Marking attendance into an already created table
7. Deleting a table storing attendance
8. Deleting attendance for a particular date
9. Enquiry regarding marks
10. Enquiry regarding attendance


Enter choice number or any other key to exit: 3

Enter the standard (eg 10 for tenth standard):

Enter: 12

Enter the section:

Enter: b

Enter the exam (eg. 'Half yearly' for half yearly exams, 'weekly 1' for first weekly exams, etc.):

Enter: weekly1

Enter the year in YYYY-YY format (eg. 2018-19):

Enter: 2018-19
You are about to delete: 12_b_weekly1_2018_19

Press 'y' to confirm deletion, press any other key to cancel it.

Enter: y
Table '12_b_weekly1_2018_19' deleted successfully.
```

Option 3



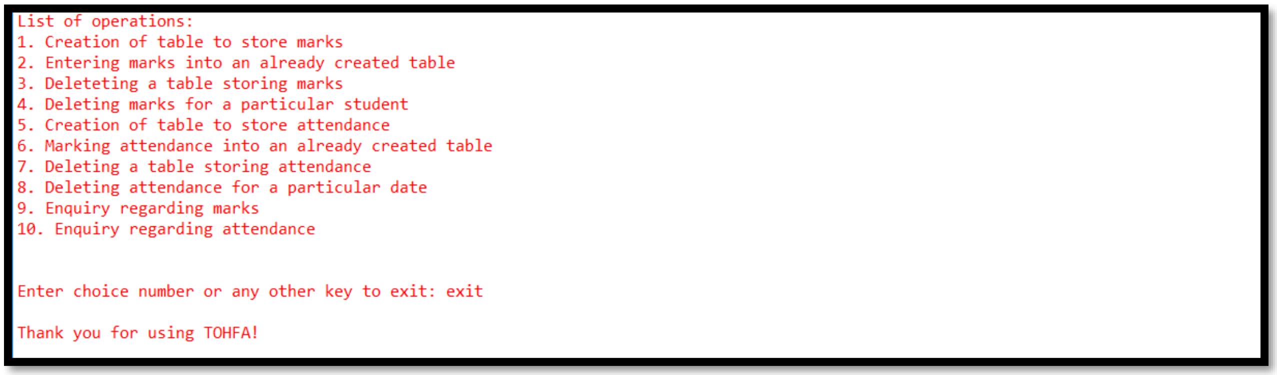
```
List of operations:
1. Creation of table to store marks
2. Entering marks into an already created table
3. Deleteting a table storing marks
4. Deleting marks for a particular student
5. Creation of table to store attendance
6. Marking attendance into an already created table
7. Deleting a table storing attendance
8. Deleting attendance for a particular date
9. Enquiry regarding marks
10. Enquiry regarding attendance


Enter choice number or any other key to exit: 8

Enter the standard (example: 10 for tenth standard)

Enter: 12

Enter the section:

Enter: b

Enter the year in YYYY-YY format (eg. 2018-19):

Enter: 2019-20

Enter number of days whose records you want to delete:1

Enter date to delete in YYYY-MM-DD format:2019-11-01
Attendance for 2019-11-01 deleted successfully
```

Option 8



```
List of operations:
1. Creation of table to store marks
2. Entering marks into an already created table
3. Deleteting a table storing marks
4. Deleting marks for a particular student
5. Creation of table to store attendance
6. Marking attendance into an already created table
7. Deleting a table storing attendance
8. Deleting attendance for a particular date
9. Enquiry regarding marks
10. Enquiry regarding attendance


Enter choice number or any other key to exit: 7

Enter the standard (example: 10 for tenth standard)

Enter: 12

Enter the section:

Enter: b

Enter the year in YYYY-YY format (eg. 2018-19):

Enter: 2019-20
You are about to delete: 12_b_attendance_2019_20

Press 'y' to confirm deletion, press any other key to cancel it.

Enter: y

Table '12_b_attendance_2019_20' deleted successfully.
```

Option 7



```
List of operations:
1. Creation of table to store marks
2. Entering marks into an already created table
3. Deleteting a table storing marks
4. Deleting marks for a particular student
5. Creation of table to store attendance
6. Marking attendance into an already created table
7. Deleting a table storing attendance
8. Deleting attendance for a particular date
9. Enquiry regarding marks
10. Enquiry regarding attendance


Enter choice number or any other key to exit: exit

Thank you for using TOHFA!
```

Thank you for using T.O.H.F.A.

DECLARATION

I, Aayush Rajesh, do hereby declare that this Computer Science project, entitled 'T.O.H.F.A.' has been majorly created by Aryan Tiwari, Satyam Rath and myself, along with assistance from our Computer Science teacher, Mrs. Radhika Sridhar and some of our classmates.

This project, made through the Spyder IDE and following the guidelines of CBSE for the Computer Science project of AISSCE 2019-20 is an original work of my teammates and me, and any sort of resemblance to any other project is just a mere coincidence.