

South Indian Cultural Association S.S. School No.2



Session: 2020-21

Topic: Tuition Management

By:

Aniruddh Modi

Tejas Mandloi

Sam Varghese

Certificate

This is to certify that myself **Aniruddh Modi** of class 12th B of Sica Senior Secondary School No.2 has successfully completed CS project on topic “Tuition Management” under the guidance of CS mam Mrs Rajni Uperati in partial fulfillment of curriculum of curriculum of Central Board Of Secondary Education for the academic session 2020-21.

Name:

Roll Number:

Internal Sign

External Sign

School Seal

Principal Sign

INDEX

- 1. ACKNOWLEDGEMENT**
- 2. INTRODUCTION OF PYTHON**
- 3. INTRODUCTION OF MYSQL**
- 4. PROJECT INFORMATION**
- 5. SOURCE CODE**
- 6. OUTPUT (SCREEN SHOTS)**
- 7. BIBLIOGRAPHY**

Acknowledgement

We would like to sincerely and profusely thank my project guide Mrs Rajini Uperati and my project members Tejas Mandloi, Sam Varghese for helping me to successfully complete this project and for their positive advice and constant motivation.

My sincere thanks to Dr Madhukar Pawar ,our principal for his constant support in completion of this project.

I would also like to thank my parents for their motivation at every step of stairs during the project. I wish to thank my classmates for their timely help.

Finally I want to thank all people who helped me directly or indirectly towards the completion of this project.

Introduction To Python

Python is an interpreted, high-level and general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library.

Python was created in the late 1980s, and first released in 1991, by Guido van Rossum as a successor to the ABC programming language.



Logo



Guido Van Rossum
(Creator Of Python)

History Of Python

Python was conceived in the late 1980s by Guido van Rossum at Centrum Wiskunde & Informatica (CWI) in the Netherlands as a successor to ABC programming language, which was inspired by SETL, capable of exception handling and interfacing with the Amoeba operating system. Its implementation began in December 1989. Van Rossum shouldered sole

responsibility for the project, as the lead developer, until 12 July 2018, when he announced his "permanent vacation" from his responsibilities as Python's Benevolent Dictator For Life, a title the Python community bestowed upon him to reflect his long-term commitment as the project's chief decision-maker. He now shares his leadership as a member of a five-person steering council. In January 2019, active Python core developers elected Brett Cannon, Nick Coghlan, Barry Warsaw, Carol Willing and Van Rossum to a five-member "Steering Council" to lead the project. Guido van Rossum has since then withdrawn his nomination for the 2020 Steering council.

Python 2.0 was released on 16 October 2000 with many major new features, including a cycle-detecting garbage collector and support for Unicode.

Python 3.0 was released on 3 December 2008. It was a major revision of the language that is not completely backward-compatible. Many of its major features were backported to Python 2.6.x and 2.7.x version series. Releases of Python 3 include the 2to3 utility, which automates (at least partially) the translation of Python 2 code to Python 3.

Python 2.7's end-of-life date was initially set at 2015 then postponed to 2020 out of concern that a large body

of existing code could not easily be forward-ported to Python 3. No more security patches or other improvements will be released for it. With Python 2's end-of-life, only Python 3.6.x and later are supported.

Python 3.9.2 and 3.8.8 were expedited as all versions of Python (including 2.7) had security issues, leading to possible remote code execution and web cache poisoning.

Design ,Philosophy And Features

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by metaprogramming and metaobjects (magic methods)). Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution (late binding), which binds method and variable names during program execution.

Python's design offers some support for functional programming in the Lisp tradition. It has filter, map, and reduce functions; list comprehensions, dictionaries, sets, and generator expressions. The standard library has two modules (itertools and functools) that implement functional tools borrowed from Haskell and Standard ML.

The language's core philosophy is summarized in the document The Zen of Python (PEP 20), which includes aphorisms such as:

Beautiful is better than ugly.

Explicit is better than implicit.

Simple is better than complex.

Complex is better than complicated.

Readability counts.

Rather than having all of its functionality built into its core, Python was designed to be highly extensible. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach.

Python strives for a simpler, less-cluttered syntax and grammar while giving developers a choice in their

coding methodology. In contrast to Perl's "there is more than one way to do it" motto, Python embraces a "there should be one—and preferably only one—obvious way to do it" design philosophy. Alex Martelli, a Fellow at the Python Software Foundation and Python book author, writes that "To describe something as 'clever' is not considered a compliment in the Python culture."

Python's developers strive to avoid premature optimization, and reject patches to non-critical parts of the CPython reference implementation that would offer marginal increases in speed at the cost of clarity. When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C, or use PyPy, a just-in-time compiler. Cython is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter.

An important goal of Python's developers is keeping it fun to use. This is reflected in the language's name—a tribute to the British comedy group Monty Python—and in occasionally playful approaches to tutorials and reference materials, such as examples that refer to spam and eggs (from a famous Monty Python sketch) instead of the standard foo and bar.

A common neologism in the Python community is *pythonic*, which can have a wide range of meanings related to program style. To say that code is *pythonic* is to say that it uses Python idioms well, that it is natural or shows fluency in the language, that it conforms with Python's minimalist philosophy and emphasis on readability. In contrast, code that is difficult to understand or reads like a rough transcription from another programming language is called *unpythonic*.

Users and admirers of Python, especially those considered knowledgeable or experienced, are often referred to as *Pythonistas*.



python

Uses Of Python

Since 2003, Python has consistently ranked in the top ten most popular programming languages in the TIOBE Programming Community Index where, as of February 2021, it is the third most popular language (behind Java, and C). It was selected Programming Language of the Year (for "the highest rise in ratings in a year") in 2007, 2010, 2018, and 2020 (the only language to do so four times).

An empirical study found that scripting languages, such as Python, are more productive than conventional languages, such as C and Java, for programming problems involving string manipulation and search in a dictionary, and determined that memory consumption was often "better than Java and not much worse than C or C++".

Large organizations that use Python include Wikipedia, Google, Yahoo!, CERN, NASA, Facebook, Amazon, Instagram, Spotify and some smaller entities like ILM and ITA. The social news networking site Reddit was written mostly in Python.

Python can serve as a scripting language for web applications, e.g., via `mod_wsgi` for the Apache web server. With Web Server Gateway Interface, a standard

API has evolved to facilitate these applications. Web frameworks like Django, Pylons, Pyramid, TurboGears, web2py, Tornado, Flask, Bottle and Zope support developers in the design and maintenance of complex applications. Pyjs and IronPython can be used to develop the client-side of Ajax-based applications. SQLAlchemy can be used as a data mapper to a relational database. Twisted is a framework to program communications between computers, and is used (for example) by Dropbox.

Libraries such as NumPy, SciPy and Matplotlib allow the effective use of Python in scientific computing, with specialized libraries such as Biopython and Astropy providing domain-specific functionality. SageMath is a mathematical software with a notebook interface programmable in Python: its library covers many aspects of mathematics, including algebra, combinatorics, numerical mathematics, number theory, and calculus. OpenCV has python bindings with a rich set of features for computer vision and image processing.

Python is commonly used in artificial intelligence projects and machine learning projects with the help of libraries like TensorFlow, Keras, Pytorch and Scikit-learn. As a scripting language with modular architecture, simple syntax and rich text processing

tools, Python is often used for natural language processing.

Python has been successfully embedded in many software products as a scripting language, including in finite element method software such as Abaqus, 3D parametric modeler like FreeCAD, 3D animation packages such as 3ds Max, Blender, Cinema 4D, Lightwave, Houdini, Maya, modo, MotionBuilder, Softimage, the visual effects compositor Nuke, 2D imaging programs like GIMP, Inkscape, Scribus and Paint Shop Pro, and musical notation programs like scorewriter and capella. GNU Debugger uses Python as a pretty printer to show complex structures such as C++ containers. Esri promotes Python as the best choice for writing scripts in ArcGIS. It has also been used in several video games, and has been adopted as first of the three available programming languages in Google App Engine, the other two being Java and Go.

Many operating systems include Python as a standard component. It ships with most Linux distributions, AmigaOS 4 (using Python 2.7), FreeBSD (as a package), NetBSD, OpenBSD (as a package) and macOS and can be used from the command line (terminal). Many Linux distributions use installers written in Python: Ubuntu uses the Ubiquity installer, while Red Hat Linux and Fedora use the Anaconda installer. Gentoo Linux uses Python in its package management system, Portage.

Python is used extensively in the information security industry, including in exploit development.

Most of the Sugar software for the One Laptop per Child XO, now developed at Sugar Labs, is written in Python. The Raspberry Pi single-board computer project has adopted Python as its main user-programming language.

LibreOffice includes Python, and intends to replace Java with Python. Its Python Scripting Provider is a core feature since Version 4.0 from 7 February 2013.



Introduction Of SQL

SQL, Structured Query Language) is a domain-specific language used in programming and designed for managing data held in a relational database management system (RDBMS), or for stream processing in a relational data stream management system (RDSMS). It is particularly useful in handling structured data, i.e. data incorporating relations among entities and variables.

SQL offers two main advantages over older read–write APIs such as ISAM or VSAM. Firstly, it introduced the concept of accessing many records with one single command. Secondly, it eliminates the need to specify how to reach a record, e.g. with or without an index.

Originally based upon relational algebra and tuple relational calculus, SQL consists of many types of statements, which may be informally classed as sublanguages, commonly: a data query language (DQL),^[a] a data definition language (DDL), a data control language (DCL), and a data manipulation language (DML). The scope of SQL includes data query, data manipulation (insert, update and delete), data definition (schema creation and modification), and data access control. Although SQL is essentially a declarative language (4GL), it also includes procedural elements.

SQL was one of the first commercial languages to utilize Edgar F. Codd's relational model. The model was described in his influential 1970 paper, "A Relational Model of Data for Large Shared Data Banks". Despite not entirely adhering to the relational model as described by Codd, it became the most widely used database language.

SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987. Since then the standard has been revised to include a larger set of features. Despite the existence of standards, most SQL code requires at least some changes before being ported to different database systems.



History

SQL was initially developed at IBM by Donald D. Chamberlin and Raymond F. Boyce after learning about the relational model from Edgar F. Codd in the early 1970s. This version, initially called SEQUEL (Structured English Query Language), was designed to manipulate and retrieve data stored in IBM's original quasi-relational database management system, System R, which a group at IBM San Jose Research Laboratory had developed during the 1970s.

Chamberlin and Boyce's first attempt at a relational database language was Square, but it was difficult to use due to subscript notation. After moving to the San Jose Research Laboratory in 1973, they began work on SEQUEL. The acronym SEQUEL was later changed to SQL because "SEQUEL" was a trademark of the UK-based Hawker Siddeley Dynamics Engineering Limited company.

After testing SQL at customer test sites to determine the usefulness and practicality of the system, IBM began developing commercial products based on their System R prototype including System/38, SQL/DS, and DB2, which were commercially available in 1979, 1981, and 1983, respectively.

In the late 1970s, Relational Software, Inc. (now Oracle Corporation) saw the potential of the concepts described by Codd, Chamberlin, and Boyce, and developed their own SQL-based RDBMS with aspirations of selling it to the U.S. Navy, Central Intelligence Agency, and other U.S. government agencies. In June 1979, Relational Software, Inc. introduced the first commercially available implementation of SQL, Oracle V2 (Version2) for VAX computers.

By 1986, ANSI and ISO standard groups officially adopted the standard "Database Language SQL" language definition. New versions of the standard were published in 1989, 1992, 1996, 1999, 2003, 2006, 2008, 2011 and, most recently, 2016.



Donald Chamberlin
(Creator of SQL)

Project Introduction

I along with my project partners wanted to work on a project which can be implemented in real life .So during our search for such a project topic ,we noticed many coaching centers in our surroundings which were still performing their administrative works manually. We could not see any better opportunity than making a project to automate their works .So like this we selected our project topic ***Tuition Management***.

For this project we used Python and the concept of My SQL Connectivity to achieve the required functionality.



Source Code

```
#Tuition Management
Program

#By:-
#Sam Varghese,Tejas
Mandloi,Anirudh Modi

import mysql.connector
import datetime
import string

#MYSQL CONNECTIVITY:-

con=mysql.connector.connect(host='localhost',user='root',password='root')

if con.is_connected():
    print('Your Connection Is Successful :) ')
else:
    print('Sorry Sir But An Unexpected Error Has Occurred ,
Please Retry.')

cur=con.cursor()

#CREATING SQL DATABASE AND TABLE:-

cur.execute('create database if not exists tution_software')
cur.execute('use tution_software')
cur.execute('create table if not exists tuition (ROLL_NUMBER
int NOT NULL PRIMARY KEY,NAME varchar(15) NOT NULL ,CLASS int
NOT NULL,SCHOOL varchar(15),PHONE_NUMBER
bigint,DATE_OF_ADMISSION varchar(10))')
#CREATING MENUS:-

exit_loop=False
```

```

while not exit_loop:
    print('\n')
    print("*"*80)
    print('\t\t\t\t\t * MENU *')
    print("*"*80)
    print("\nPlease select the task which you want me to
perform:-\n\n\
    1)Registration\n\
    2)Student's information of a specific class\n\
    3)Get records of a specific student\n\
    4>Delete records of a specific student\n\
    5)Updating records of a specific student\n\
    6)Exit")

    try:
        choice=int(input('\nYour choice please:- '))
        if choice not in range(1,7):
            choice=int(input('\nIncorrect choice , please
enter a valid choice:- '))
    except Exception:
        choice=int(input('\nIncorrect choice , please enter a
valid choice:- '))

    if choice==1:#Registration Section

        #Getting the name of student:
        stu_name=string.capwords(input('\nPlease enter the
name of the student who want to join tuition: '))

        #Getting the class of student:
        try:
            stu_class=int(input('Please enter the class of
'+stu_name+' : '))

```

```

        if stu_class>12 or stu_class<=0:#Raising custom
error if class>12 or <=0
            raise ValueError('Invalid class number')

    except Exception as e:
        print('ERROR:-',e)
        print('\nSorry to say ,but an invalid class has
been entered so kindly rectify this data')
        stu_class=int(input('Please enter the class of
'+stu_name+' : '))
        print('\n')

    #Getting the roll number of the student:
    try:
        stu_roll_no=int(input('Please enter the roll
number of '+stu_name+' : '))
    except Exception as e:
        print('ERROR:-',e)
        print('\nSorry to say but an invalid roll number
has been entered so kindly rectify this data.')
        stu_roll_no=int(input('Please enter the roll
number of '+stu_name+' : '))
        print('\n')

    #Getting the school name of the student
    stu_school=string.capwords(input('Please enter the
school name of '+stu_name+' : '))

    try:
        stu_ph_no=int(input('Please enter the contact
number of '+stu_name+' : '))
    except Exception as e:
        print('ERROR:-',e)
        print('\nSorry but an invalid contact number has
been entered so kindly rectify this data.')

```



```

        stu_ph_no=int(input('Please enter the contact
number of '+stu_name+': '))
        print('\n')

        #Generating current date

stu_admission_date=datetime.datetime.now().strftime('%d-%m-%Y'
)

        #Making SQL Query

        query="insert into tution
values({},'{}',{},{},'{}',{},{})".format(stu_roll_no,stu_name,s
tu_class,stu_school,stu_ph_no,stu_admission_date)

        #Inserting data to MySQL Database
        try:
            cur.execute(query)
            con.commit()
            except mysql.connector.IntegrityError:#Handling
repeated roll number exception

            cur.execute('select * from tution where
ROLL_NUMBER=%s'%stu_roll_no)#Showing about the student with
that roll no

            data=cur.fetchall()

            print('\nSorry but this roll number has been used
for '+data[0][1]+' also.')

            print('\nHence please re-enter a valid roll
number.')

            stu_roll_no=int(input('Please re-enter the roll
number of '+stu_name+': '))

```

```

        query="insert into tution
values({},'{}',{},{},'{}',{},{})".format(stu_roll_no,stu_name,s
tu_class,stu_school,stu_ph_no,stu_admission_date)
        cur.execute(query)
        con.commit()

    print('\nName successfully registered')

elif choice==2:#Getting record of a specific class

    try:#Getting class from the user
        classs=int(input('\nPlease enter the class whose
records you want to display: '))
    except Exception as e:
        print('ERROR:-',e)
        print('\nSorry but an invalid class has been
entered so kindly rectify this data')
        classs=int(input('\nPlease enter the class whose
records you want to display: '))
        print('\n')

    #Creating SQL Query
        query='select * from tution where
CLASS={} '.format(classs)

    #Getting data from SQL Database
    cur.execute(query)
    data=cur.fetchall()

    #Showing data
    print('\nPrinting data of class ',classs,':-')
    print('\n')
    count=1
    for i in data:

```

```

        print(count,')Roll   Number:-',i[0],',
,\nName:-',i[1],',   ,\nClass:-',i[2],',   ,\nSchool   Name:-',i[3],',
,\nContact Number:-',i[4],',   ,\nDate of Admission:-',i[5])
        print('\n')
        count+=1

elif choice==3:#Getting records of a specific student

    try:

        rn=input("\nPlease enter the roll number of the
student. : ")
        query="SELECT * from tution WHERE ROLL_NUMBER="+rn
        cur.execute(query)
        data=cur.fetchall()
        print(data)

        c=cur.rowcount

        if c==0:
            rn=int(input("\nSorry ,but there is no student
in this tution by this roll number so please re-enter the
correct roll\
number:-"))

            cur.execute(query)
            data=cur.fetchall()

        print("\nShowing records of the student:-\n")
        print('Name:-   ',data[0][1],'\nClass:-
',data[0][2],'\nRoll   Number:-   ',data[0][0],'\nSchool   Name:-
',data[0][3],\

```

```

                                '\nContact information:-
',data[0][4],'\nDate of Admission:- ',data[0][5])

    except Exception as e:
        print('\nERROR:-',e)
        print("\nPLEASE TRY AGAIN SOMETHING WENT
WRONG...")

elif choice==4:#Deleting records of a student

    try:
        dr=input("\nPlease enter the roll number of the
student whose records you want to delete:- ")

        query="DELETE from tution WHERE ROLL_NUMBER="+dr
        cur.execute(query)
        con.commit()
        c=cur.rowcount
        if c>0:
            print("\nData successfully deleted.")
        else:
            dr=input('Roll number. '+dr+' not found
,please enter a valid roll number:- ')
            query="DELETE from tution WHERE
ROLL_NUMBER="+dr
            cur.execute(query)
            con.commit()
            print("\nData successfully deleted.")
    except Exception as e:
        print('\nERROR: ',e)
        print("PLEASE TRY AGAIN SOMETHING WENT WRONG...")

```

```

elif choice==5:##Modifying Data
    try:
        roll_no = input('\nEnter roll number whose data
you want to update:- ')
        query = 'SELECT * FROM tution WHERE
ROLL_NUMBER='+roll_no
        cur.execute(query)
        record=cur.fetchall()
        c=cur.rowcount

        if c == 0:
            print('Student with ' ,roll_no,'does not
exist! Please enter valid roll number.')
        else:
            sname=record[0][1]
            sclass=record[0][2]
            sschool=record[0][3]
            scontact=record[0][4]
            print('Roll Number:- ',record[0][0],'\nName:-
',sname,'\nClass:- ',sclass,'\nRoll          Number:-
',record[0][0],'\nSchool Name:- ',sschool,\
            '\nContact information:- ',scontact,'\nDate
of Admission:- ',record[0][5])
            print('\nEnter respective values to modify.
Press enter for no change\n')

            change=input('\nEnter new name of the student:
')

            if len(change)>0:
                sname=change

            change=input('Enter new class of the student:
')

            if len(change)>0:

```

```

        change=int(change)
        if change>12:
            raise ValueError("Invalid class
number")

        sclass=change

        change=input('Enter name of new school of the
student: ')

        if len(change)>0:
            sschool=change

        change=input('Enter new contact of the
student: ')

        if len(change)>0:
            change=int(change)
            scontact=change

        query="UPDATE tution set NAME='"+sname+"'+"
,CLASS="+str(sclass)+"          ,SCHOOL="+str(sschool)+"+"
,PHONE_NUMBER="+str(scontact)+" WHERE ROLL_NUMBER="+roll_no
        cur.execute(query)

        print('\nData successfully updated\n')

    except Exception as e:
        print('\nERROR: ',e)
        print("PLEASE TRY AGAIN SOMETHING WENT WRONG...")

elif choice==6:##Exit
    print("\nThank you. HAVE A NICE DAY!!")
    con.close()
    exit_loop=True

```

SQL Connectivity And Setup

The following code is that part of our project which sets up SQL Connectivity with Python along the code of menus.

```
#Tuition Management Program

#By:-
#Sam Varghese,Tejas
Mandloi,Anirudh Modi

import mysql.connector
import datetime
import string

#MYSQL CONNECTIVITY:-

con=mysql.connector.connect(host='localhost',user='root',password='root')

if con.is_connected():
    print('Your Connection Is Successful :) ')
else:
    print('Sorry Sir But An Unexpected Error Has Occurred ,
Please Retry.')

cur=con.cursor()

#CREATING SQL DATABASE AND TABLE:-

cur.execute('create database if not exists tution_software')
cur.execute('use tution_software')
cur.execute('create table if not exists tution (ROLL_NUMBER
int NOT NULL PRIMARY KEY,NAME varchar(15) NOT NULL ,CLASS int
```



```
Python 3.8.6 Shell
File Edit Shell Debug Options Window Help
Python 3.8.6 (tags/v3.8.6:db45529, Sep 23 2020, 15:52:53) [MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Python 3.8 Files\fun.py =====
Your Connection Is Successfull :)

*****
== TUTION MANAGEMENT ==
*****

Please select the task which you want me to perform:-

1)Registration
2)Student's information of a specific class
3)Get records of a specific student
4>Delete records of a specific student
5)Updating records of a specific student
6)Exit

Your choice please:-
```

Registration Section

The following is the code of registering the names of students.

```
if choice==1:#Registration Section

    #Getting the name of student:
    stu_name=string.capwords(input('\nPlease enter the
name of the student who want to join tuition: '))

    #Getting the class of student:
```

```

        try:
            stu_class=int(input('Please enter the class of
'+stu_name+' : '))

            if stu_class>12 or stu_class<=0:#Raising custom
error if class>12 or <=0
                raise ValueError('Invalid class number')

        except Exception as e:
            print('ERROR:-',e)
            print('\nSorry to say ,but an invalid class has
been entered so kindly rectify this data')
            stu_class=int(input('Please enter the class of
'+stu_name+' : '))
            print('\n')

        #Getting the roll number of the student:
        try:
            stu_roll_no=int(input('Please enter the roll
number of '+stu_name+' : '))
        except Exception as e:
            print('ERROR:-',e)
            print('\nSorry to say but an invalid roll number
has been entered so kindly rectify this data.')
            stu_roll_no=int(input('Please enter the roll
number of '+stu_name+' : '))
            print('\n')

        #Getting the school name of the student
        stu_school=string.capwords(input('Please enter the
school name of '+stu_name+' : '))

        try:
            stu_ph_no=int(input('Please enter the contact
number of '+stu_name+' : '))
        except Exception as e:
            print('ERROR:-',e)

```

```

        print('\nSorry but an invalid contact number has
been entered so kindly rectify this data.')
        stu_ph_no=int(input('Please enter the contact
number of '+stu_name+: '))
        print('\n')

        #Generating current date

stu_admission_date=datetime.datetime.now().strftime('%d-%m-%Y'
)

        #Making SQL Query
        query="insert into tution
values({},'{}',{},{},{},{})".format(stu_roll_no,stu_name,s
tu_class,stu_school,stu_ph_no,stu_admission_date)

        #Inserting data to MySQL Database
        try:
            cur.execute(query)
            con.commit()

        except mysql.connector.IntegrityError: #Handling
repeated roll number exception

            cur.execute('select * from tution where
ROLL_NUMBER=%s'%stu_roll_no) #Showing about the student with
that roll no

            data=cur.fetchall()

            print('\nSorry but this roll number has been used
for '+data[0][1]+' also.')

            print('\nHence please re-enter a valid roll
number.')

            stu_roll_no=int(input('Please re-enter the roll
number of '+stu_name+: '))

```

```

        query="insert into tution
values({},'{}',{},{},'{}',{},{},'{}')".format(stu_roll_no,stu_name,s
tu_class,stu_school,stu_ph_no,stu_admission_date)
        cur.execute(query)
        con.commit()

    print('\nName successfully registered')

```

The following screenshot shows the working of Registration section:

```

Python 3.8.6 Shell
File Edit Shell Debug Options Window Help
Python 3.8.6 (tags/v3.8.6:db45529, Sep 23 2020, 15:52:53) [MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Python 3.8 Files\fun.py =====
Your Connection Is Successfull :)

*****
== TUTION MANAGEMENT ==
*****

Please select the task which you want me to perform:-

1)Registration
2)Student's information of a specific class
3)Get records of a specific student
4>Delete records of a specific student
5)Updating records of a specific student
6)Exit

Your choice please:- 1

Please enter the name of the student who want to join tution: sam
Please enter the class of Sam: 12
Please enter the roll number of Sam: 12217
Please enter the school name of Sam: Sica
Please enter the contact number of Sam: 7828515205

Name successfully registered

*****
== TUTION MANAGEMENT ==
*****
Ln: 45 Col: 21

```

Accessing Records Of Specific Class

The following is the code for accessing records of a specific class of tuition.

```
try:#Getting class from the user
    classs=int(input('\nPlease enter the class whose
records you want to display: '))
except Exception as e:
    print('ERROR:-',e)
    print('\nSorry but an invalid class has been
entered so kindly rectify this data')
    classs=int(input('\nPlease enter the class whose
records you want to display: '))
    print('\n')

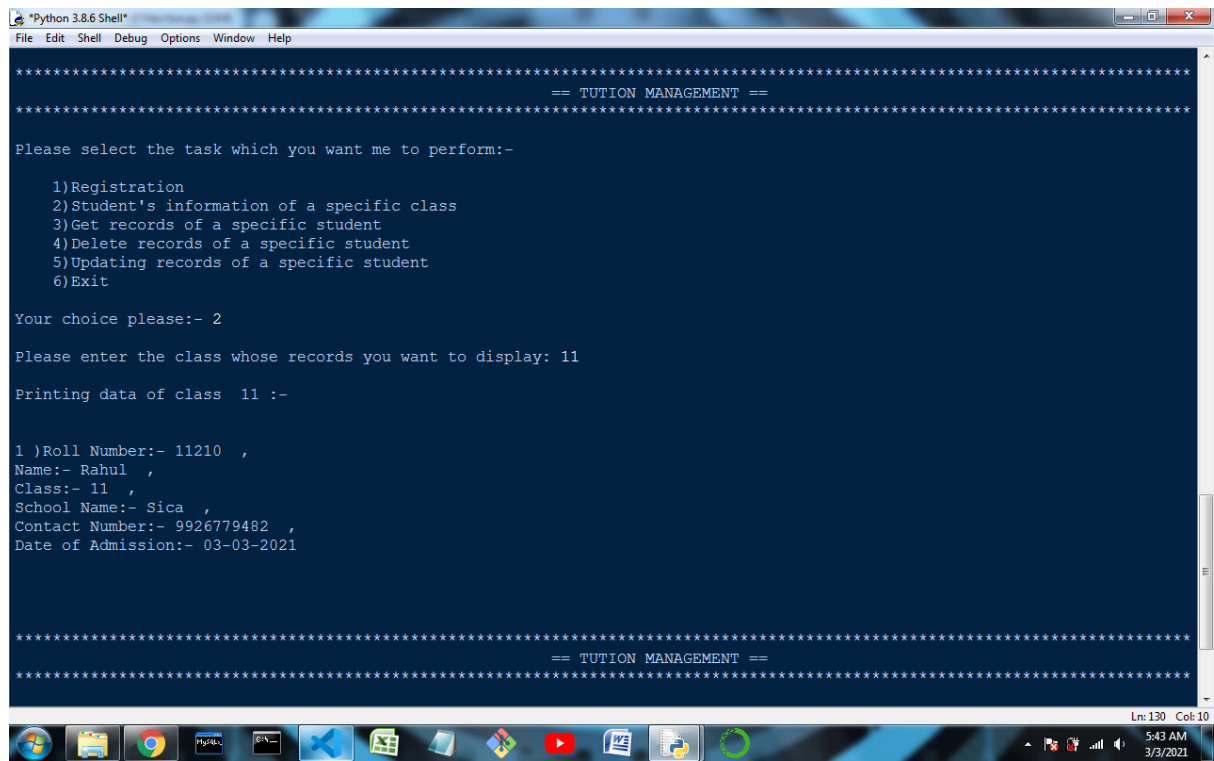
#Creating SQL Query
query='select * from tuition where
CLASS={}'.format(classs)

#Getting data from SQL Database
cur.execute(query)
data=cur.fetchall()

#Showing data
print('\nPrinting data of class ',classs,':-')
print('\n')
count=1
for i in data:
    print(count,') Roll Number:-',i[0],',
,\nName:-',i[1],', ,\nClass:-',i[2],', ,\nSchool Name:-',i[3],',
,\nContact Number:-',i[4],', ,\nDate of Admission:-',i[5])
```

```
print('\n')  
count+=1
```

The following screenshot is the working of this part of program:



```
Python 3.8.6 Shell  
File Edit Shell Debug Options Window Help  
*****  
== TUTION MANAGEMENT ==  
*****  
Please select the task which you want me to perform:-  
  
1)Registration  
2)Student's information of a specific class  
3)Get records of a specific student  
4>Delete records of a specific student  
5)Updating records of a specific student  
6)Exit  
  
Your choice please:- 2  
  
Please enter the class whose records you want to display: 11  
  
Printing data of class 11 :-  
  
1 )Roll Number:- 11210 ,  
Name:- Rahul ,  
Class:- 11 ,  
School Name:- Sica ,  
Contact Number:- 9926779482 ,  
Date of Admission:- 03-03-2021  
  
*****  
== TUTION MANAGEMENT ==  
*****  
Ln: 130 Col: 10  
5:43 AM  
3/3/2021
```

Accessing Records Of Specific Student

The following is the code for accessing of a specific student

```
elif choice==3:#Getting records of a specific student

    try:

        rn=input("\nPlease enter the roll number of the
                student. : ")
        query="SELECT * from tution WHERE ROLL_NUMBER="+rn
        cur.execute(query)
        data=cur.fetchall()
        print(data)

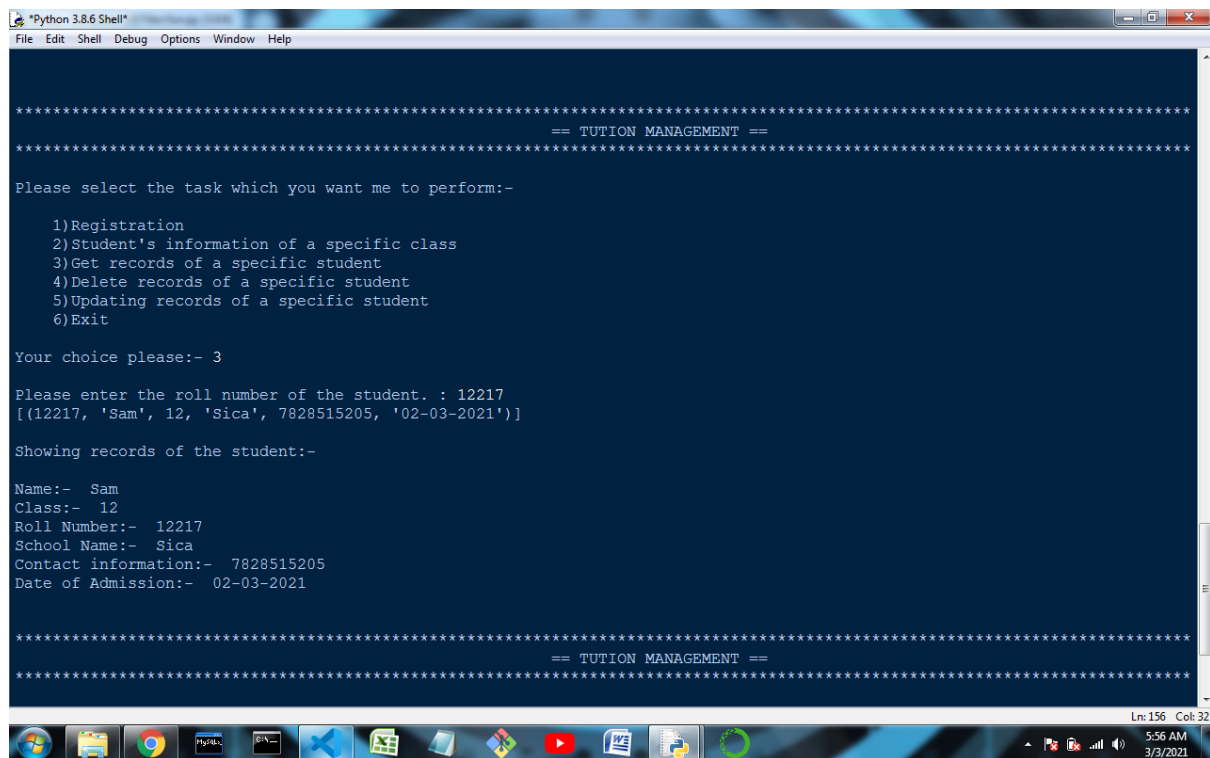
        c=cur.rowcount

        if c==0:
            rn=int(input("\nSorry ,but there is no student
            in this tution by this roll number so please re-enter the
            correct roll\
            number:-"))
            cur.execute(query)
            data=cur.fetchall()

        print("\nShowing records of the student:-\n")
        print('Name:- ',data[0][1],'\nClass:-
        ',data[0][2],'\nRoll Number:- ',data[0][0],'\nSchool Name:-
        ',data[0][3],\
        '\nContact information:-
        ',data[0][4],'\nDate of Admission:- ',data[0][5])
```

```
except Exception as e:
    print('\nERROR:-',e)
print("\nPLEASE TRY AGAIN SOMETHING WENT
      WRONG...")
```

The following is the output of this part of program:



```
*Python 3.8.6 Shell*
File Edit Shell Debug Options Window Help

*****
                        == TUTION MANAGEMENT ==
*****

Please select the task which you want me to perform:-

1)Registration
2)Student's information of a specific class
3)Get records of a specific student
4>Delete records of a specific student
5)Updating records of a specific student
6)Exit

Your choice please:- 3

Please enter the roll number of the student. : 12217
[(12217, 'Sam', 12, 'Sica', 7828515205, '02-03-2021')]

Showing records of the student:-

Name:- Sam
Class:- 12
Roll Number:- 12217
School Name:- Sica
Contact information:- 7828515205
Date of Admission:- 02-03-2021

*****
                        == TUTION MANAGEMENT ==
*****

Ln: 156 Col: 32
```


Deleting Records

The following is the code in order to delete the records of a specific student

```
elif choice==4:#Deleting records of a student

    try:
        dr=input("\nPlease enter the roll number of the
student whose records you want to delete:- ")

        query="DELETE from tution WHERE ROLL_NUMBER="+dr
        cur.execute(query)
        con.commit()
        c=cur.rowcount
        if c>0:
            print("\nData successfully deleted.")
        else:
            dr=input('Roll number. '+dr+' not found
,please enter a valid roll number:- ')
            query="DELETE from tution WHERE
ROLL_NUMBER="+dr
            cur.execute(query)
            con.commit()
            print("\nData successfully deleted.")
    except Exception as e:
        print('\nERROR: ',e)
        print("PLEASE TRY AGAIN SOMETHING WENT WRONG...")
```

The following is the output of this part of program:

```
Python 3.8.6 Shell
File Edit Shell Debug Options Window Help
6)Exit

Your choice please:- 5

Enter roll number whose data you want to update:- 11217
Student with 11217 does not exist! Please enter valid roll number.

ERROR: name 'sname' is not defined
PLEASE TRY AGAIN SOMETHING WENT WRONG...

=====
== TUTION MANAGEMENT ==
=====

Please select the task which you want me to perform:-

1)Registration
2)Student's information of a specific class
3)Get records of a specific student
4>Delete records of a specific student
5)Updating records of a specific student
6)Exit

Your choice please:- 4

Please enter the roll number of the student whose records you want to delete:- 11210

Data successfully deleted.

=====
== TUTION MANAGEMENT ==
=====

Ln: 229 Col: 31
5:58 AM
3/3/2021
```

Modifying Records

The following is the code in order to modify records of a specific student

```
elif choice==5:##Modifying Data
    try:
        roll_no = input('\nEnter roll number whose data
you want to update:- ')
        query = 'SELECT * FROM tution WHERE
ROLL_NUMBER='+roll_no
        cur.execute(query)
```

```

        record=cur.fetchall()
        c=cur.rowcount

        if c == 0:
            print('Student with ',roll_no,'does not
exist! Please enter valid roll number.')
        else:
            sname=record[0][1]
            sclass=record[0][2]
            sschool=record[0][3]
            scontact=record[0][4]
            print('Roll Number:- ',record[0][0],'\nName:-
',sname,'\nClass:- ',sclass,'\nRoll Number:-
',record[0][0],'\nSchool Name:- ',sschool,\
            '\nContact information:- ',scontact,'\nDate
of Admission:- ',record[0][5])
            print('\nEnter respective values to modify.
Press enter for no change\n')

            change=input('\nEnter new name of the student:
')

            if len(change)>0:
                sname=change

            change=input('Enter new class of the student:
')

            if len(change)>0:
                change=int(change)
                if change>12:
                    raise ValueError("Invalid class
number")

                sclass=change

            change=input('Enter name of new school of the
student: ')

            if len(change)>0:
                sschool=change

```

```

        change=input('Enter new contact of the
student: ')

        if len(change)>0:
            change=int(change)
            scontact=change

        query="UPDATE tution set NAME='"+sname+"','"+
,CLASS="+str(sclass)+" ,SCHOOL='"+sschool+"','"+
,PHONE_NUMBER="+str(scontact)+" WHERE ROLL_NUMBER="+roll_no
        cur.execute(query)

        print('\nData successfully updated\n')

    except Exception as e:
        print('\nERROR: ',e)
        print("PLEASE TRY AGAIN SOMETHING WENT WRONG...")

```

The following is the output for this part of program:

```

Python 3.8.6 Shell
File Edit Shell Debug Options Window Help
===== TUTION MANAGEMENT =====
Please select the task which you want me to perform:-

1)Registration
2)Student's information of a specific class
3)Get records of a specific student
4)Delete records of a specific student
5)Updating records of a specific student
6)Exit

Your choice please:- 5

Enter roll number whose data you want to update:- 12217
Roll Number:- 12217
Name:- Sam
Class:- 12
Roll Number:- 12217
School Name:- Sica
Contact information:- 24234234234
Date of Admission:- 03-03-2021

Enter respective values to modify. Press enter for no change

Enter new name of the student: Sam varghese
Enter new class of the student: 12
Enter name of new school of the student: sica ss school
Enter new contact of the student: 7828414205

Data successfully updated
Ln: 299 Col: 10
5:59 AM
3/3/2021

```

Exit

The following is the code in order to end program

```
elif choice==6:##Exit
    print("\nThank you. HAVE A NICE DAY!!")
    con.close()
    exit_loop=True
```

The following is the working for this part of program:

```
*****
                        == TUTION MANAGEMENT ==
*****
Please select the task which you want me to perform:-

1)Registration
2)Student's information of a specific class
3)Get records of a specific student
4>Delete records of a specific student
5)Updating records of a specific student
6)Exit

Your choice please:- 6

Thank you. HAVE A NICE DAY!!
>>>
```

Bibliography

- Sumita Arora
- Google
- Stack Overflow
- Geeksforgeeks
- Python.org

THANK YOU