In the Command Window, type main and press Enter to run the script.

The main script initializes the StudentDatabase and demonstrates the primary functionalities available to the user. Below are details on each function in the code and how a user can interact with it.

---

## 2. Adding New Students to the Database

The main script includes an example of adding new students.

Open main.m and locate the lines where sample students are added, such as:
matlab

To add a new student, use the same line format:
matlab

- **StudentID**: A string,
- **Name**: The student's name
- **Age**: The student's age as an integer
- **GPA**: The student's GPA as a decimal
- **Major**: The student's major as a string

Save and re-run main.m. The student will now be added to the database.

---

## 3. Displaying All Students in the Database

You can view a list of all students in the database using the following steps:

1. Locate the DisplayDatabase function call in the main script.

Add the following line anywhere in main.m after adding students:

Copy code
database.displayDatabase();

2. Save and run main.m. The command window will display the **ID, Name, Age, GPA, and Major** of each student in the database.

---

## 4. Finding a Student by ID

To find and display a specific student's details by their ID:

Locate the findStudentByID function in the main script, the following is an example:

```
student = database.findStudentByID('003');
```

Replace '003' with the ID of the student you wish to find. For example, if the ID is 006, write:

```
student = database.findStudentByID('006');
```

This code displays the student info

```
if ~isempty(student)
    student.displayInfo();
end
```

1. Save and run main.m. If the student with the specified ID exists, their information will be printed in the Command Window.

---

## 5. Filtering Students by Major

To retrieve and display all students from a specific major:

Use the getStudentsByMajor function in main.m:

```
engineeringStudents = database.getStudentsByMajor('Engineering');
```

1. Replace 'Engineering' with the desired major (e.g., 'Mathematics').

This loop will print information for all students in this major:

```
disp('Engineering Students:');
for i = 1:length(engineeringStudents)
    engineeringStudents(i).displayInfo();
end
```

2. Save and run main.m. The information of all students in the specified major will be displayed in the Command Window.

---

## 6. Saving the Database to a File

To save the current database of students:

The saveDatabase function is already demonstrated in main.m as:

```
database.saveDatabase('studentDatabase.mat');
```

1. This command saves the StudentDatabase object in a .mat file named studentDatabase.mat.

If you want to change the filename, replace 'studentDatabase.mat' with a new name, like:

```
database.saveDatabase('myDatabase.mat');
```

2. Run main.m to save the database. Check the Current Folder panel in MATLAB to ensure the .mat file has been created.

---

## 7. Loading the Database from a File

To load a previously saved database:

Use the loadDatabase function, which might look like:

```
database = database.loadDatabase('studentDatabase.mat');
```

Replace 'studentDatabase.mat' with the filename you want to load, like:

Copy code
```
database = database.loadDatabase('myDatabase.mat');
```

1. Run main.m to load the database. If the file exists, MATLAB will load it into the database object.

---

## 8. Updating a Student's GPA

To update a student's GPA:

Use the updateGPA method for an existing student object. For example:

```
student = database.findStudentByID('001');
if ~isempty(student)
    student = student.updateGPA(3.9);
end
```

1. Replace '001' with the desired student's ID and 3.9 with the new GPA.
2. Run main.m. The Command Window will display the updated GPA if the student exists.

---

## 9. Generating Visualizations

The DataVisualizer class provides static methods for generating three types of plots. To use each, run the respective commands in main.m

**a) GPA Distribution Histogram**

The following code to main.m generates a histogram of GPAs:

figure;

DataVisualizer.plotGPADistribution(database);

Run main.m and a histogram will open in a Figure Window, showing the distribution of student GPAs.

b) Average GPA by Major

This code allows you to average GPA of students grouped by major:

figure;

DataVisualizer.plotAverageGPAByMajor(database);

Run main.m. A bar plot will appear, showing the average GPA for each major in the database.

c) Age Distribution Histogram

This code displays a histogram of the age distribution among students:

figure;

DataVisualizer.plotAgeDistribution(database);

Run main.m. A histogram will open in a Figure Window, displaying the age distribution of students.

---

## 10. Error Handling and Validation

This code provides built-in error handling for file operations (e.g., when saving or loading the database). If a file operation fails, MATLAB will display an error message in the Command

Window. Similarly, input validation can be added for user-specific needs, such as ensuring IDs are unique or verifying GPA and age.

---