# Mastering Embedded System Online Diploma

www.learn-in-depth.com


## First Term Project 2

## Student Database System

BY :
Eng. Mohamed Kamel Aly

My Profile : No progess page created,
Email: mohamed.kamel.m73@gmail.com

# Contents

## Problem Statement

Write a program to build a simple Software for Student Information Management System which can perform the following operations:

- Store the First name of the student.
- Store the Last name of the student.
- Store the unique Roll number for every student.
- Store the GPA of every student.
- Store the courses registered by the student.

## Approach

The idea is to form an individual functions for every operation. All the functions are unified together to form software.

- Add Student Details From File
- Add Student Details Manually
- Find the student by the given roll number
- Find the student by the given first name
- Find the students registered in a course
- Count of students
- Delete a student
- Update Student

## Configuration

In students.h, there are some config options to se

```
// ////////////////////////////////////////
// Configuration
// ////////////////////////////////////////
#define BUFFER_LENGTH 50
#define MAX_NUM_OF_COURSES 10
#define MAX_NAME_CHARACTERS 50
#define FILENAME "studentsDB.txt"
```

BUFFER_LENGTH:

Sets the maximum number of students to save in the DB

MAX_NUM_OF_COURSES:

Sets the maximum No. of courses each single student can take

MAX_NAME_CHARACTERS:

Sets the maximum first/last name No. of characters

## Code APIs And Helper Methods

Github Code:

### APIs

These are the main APIs used to provide functionality

#### addManually

```
void addManually(FIFO_Buf_t *S_FIFO);
```

This API adds a student manually, it uses helper methods of **FIFO_validator, enterStudentInfo, countStudents**

#### addFromFile

```
void addFromFile(FIFO_Buf_t *S_FIFO);
```

This API adds students from an existing text file, it uses helper methods of **FIFO_validator ,isUniqueID**

#### printStudentByID

```
void printStudentByID(FIFO_Buf_t *S_FIFO);
```

This API prints a student by ID, it uses helper methods of **FIFO_validator, getStudentByID, printStudentInfo**

#### printByFirstName

```
void printByFirstName(FIFO_Buf_t *S_FIFO);
```

This API prints a student by ID, it uses helper methods of **FIFO_validator, printStudentInfo**

#### printCourseStudents

```
void printCourseStudents(FIFO_Buf_t *S_FIFO);
```

This API prints students taking a certain course ID, it uses helper methods of **FIFO_validator, printStudentInfo**

#### printAll

```
void printAll(FIFO_Buf_t *S_FIFO);
```

This API prints All students, it uses helper methods of **FIFO_validator, printStudentInfo**

#### deleteByID

```
void deleteByID(FIFO_Buf_t *S_FIFO);
```

This API deletes a student by ID, it uses helper methods of **FIFO_validator, printStudentInfo, getStudentByID**

#### updateByID

```
void updateByID(FIFO_Buf_t *S_FIFO);
```

This API updates a student by ID, it uses helper methods of **FIFO_validato, getStudentByID, enterStudentInfo**

### countStudents

```
void countStudents(FIFO_Buf_t *S_FIFO);
```

This API counts all students, it uses the helper method **FIFO_validator**

## HelperMethods

These are the helper methods used by the main APIs to enhance code readability/maintainability

### isUniqueID

```
bool isUniqueID(FIFO_Buf_t *S_FIFO, uint32 ID);
```

This method is called to check if a given ID input is unique or not, it uses the helper method **getStudentByID**

### getStudentByID

```
Student *getStudentByID(FIFO_Buf_t *S_FIFO, uint32 ID, int
*studentListingNumP);
```

This method iterates on student IDs to return a pointer to the student with the ID, if not found, it returns NULL, it uses the helper method **FIFO_validator**

### printStudentInfo

```
void printStudentInfo(Student *s);
```

This method prints a given student's information

### FIFO_validator

```
bool FIFO_validator(FIFO_Buf_t *S_FIFO, bool isEmptyValidate, bool
isFullValidate);
```

This method is used to validate the FIFO buffer and its' data to check if it's invalid, empty, or full

### enterStudentInfo

```
void enterStudentInfo(FIFO_Buf_t *S_FIFO, Student *targetStudent, uint8
choice);
```

This method is used to enter student data and save it the student pointer

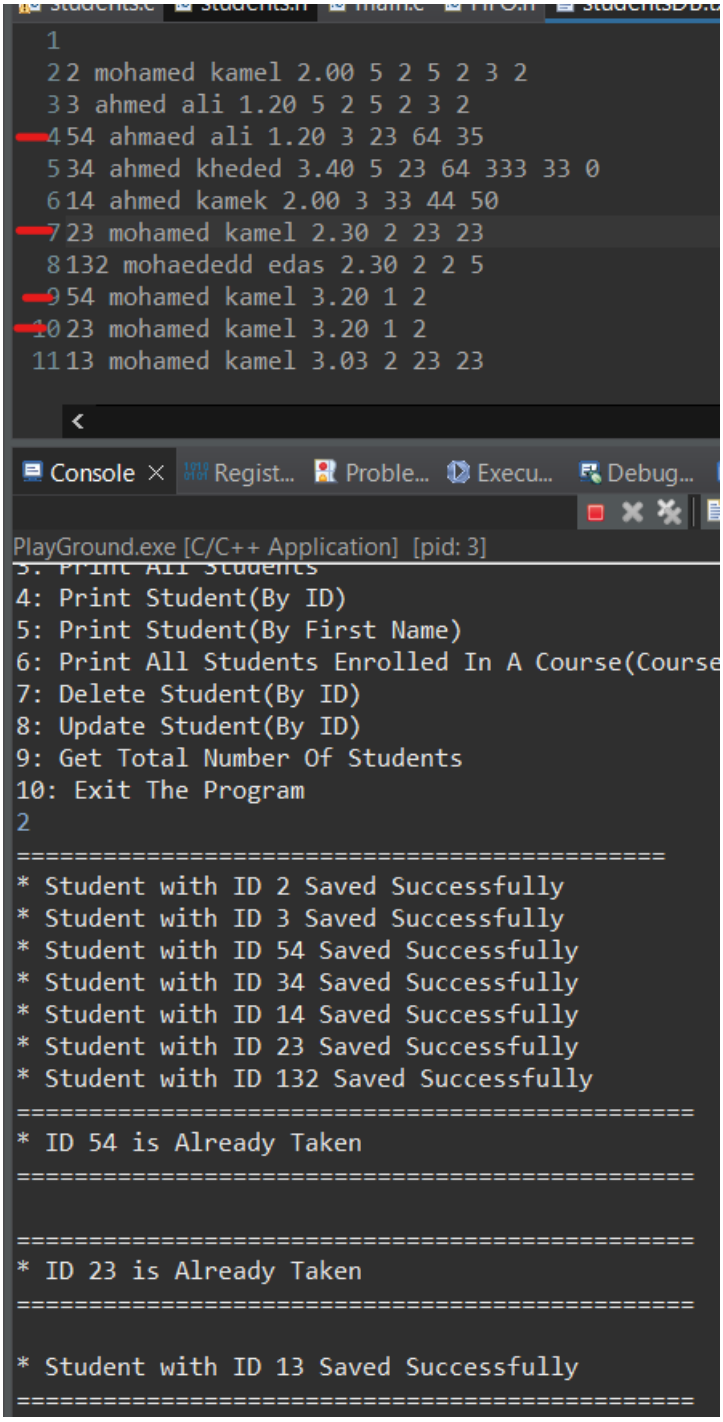
# Tests

These are the options which are presented to the user upon starting the program

```
Choose One Of The Following Options
1: Add Student(Manually)
2: Add Student(Text File)
3: Print All Students
4: Print Student(By ID)
5: Print Student(By First Name)
6: Print All Students Enrolled In A Course(Course ID)
7: Delete Student(By ID)
8: Update Student(By ID)
9: Get Total Number Of Students
10: Exit The Program
3
```

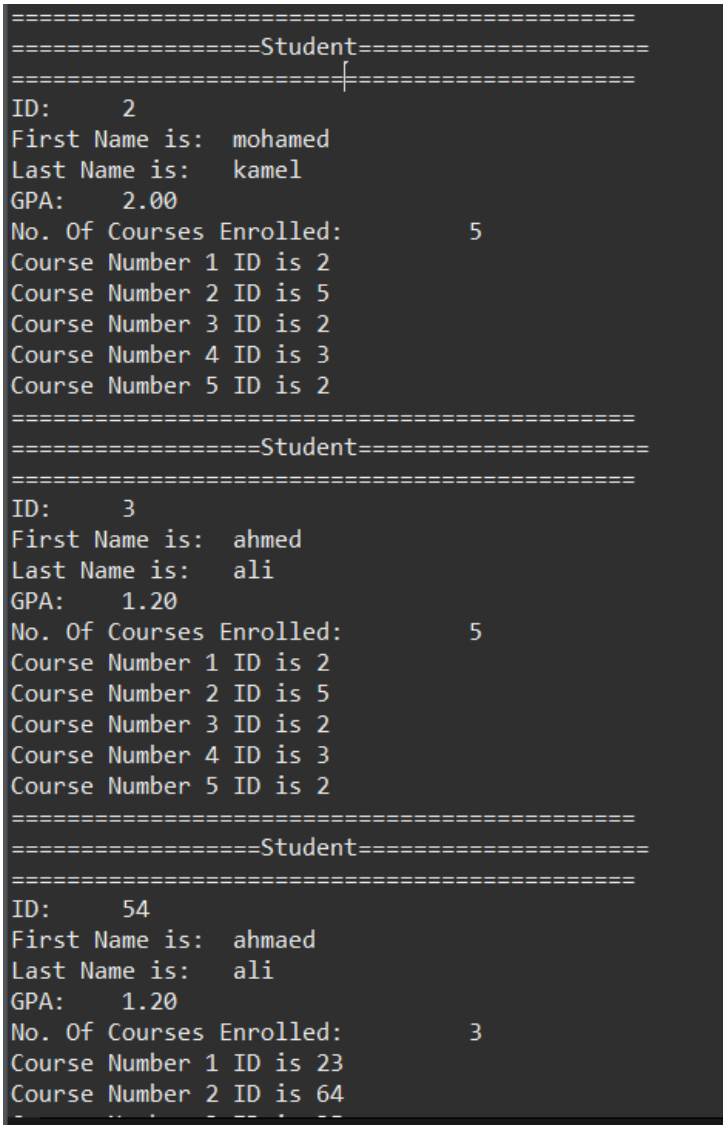
## Add students from file:

While reading each line, any non-unique ID entry is skipped and the program continues saving from the next line



## Print All Students

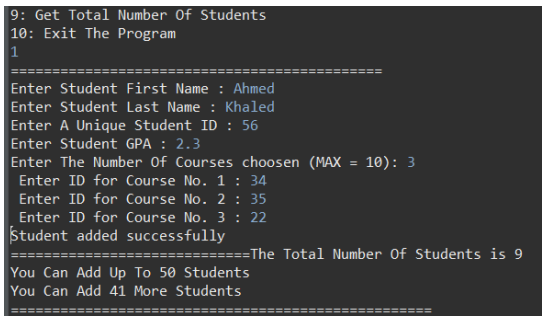
The Students added from the previous file are present when choosing option 3 (PrintAll)

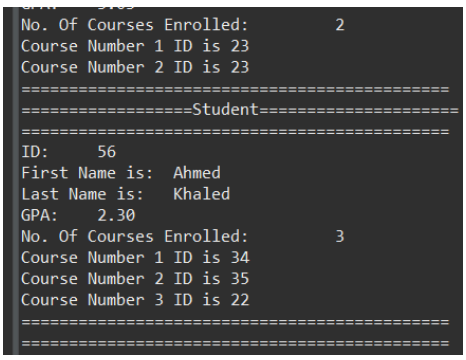



## Add Student Manually

The left screenshot show the process of adding the student

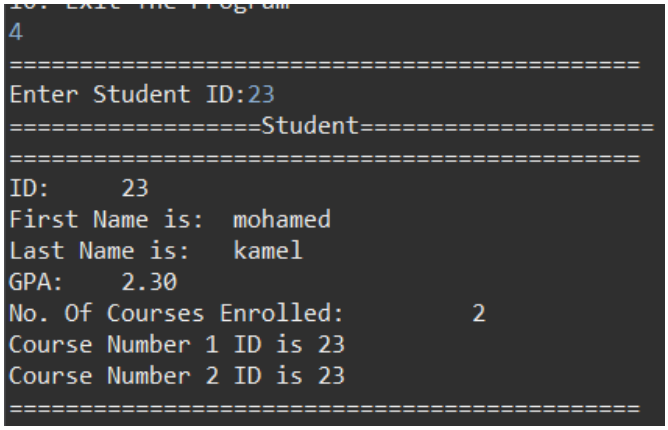
, and the right one shows it's been added by printing all students



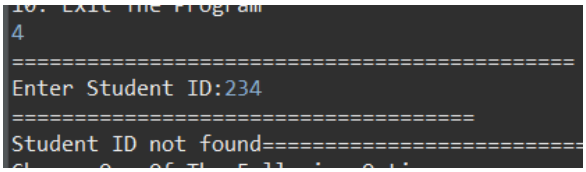



## Print Student by ID

From the above added students, searching for a student with an ID prints the target correctly

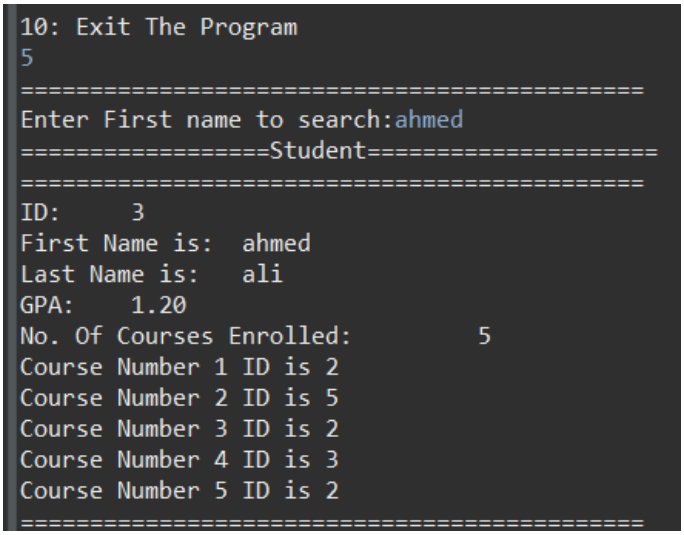


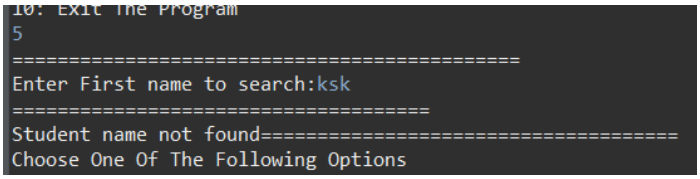

Non-existing IDs print the following




## Print Student By First Name

This also works by entering the name, then the found student is printed, entering a non-existing name is shown in the right screenshot



## Print Students In Course

The students taking course No 33 are all printed as intended

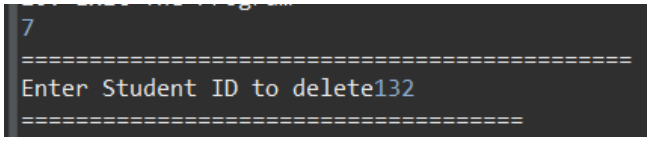
```
6
==========================================
Enter Course ID to search:33
==========================================
ID:     34
First Name is:  ahmed
Last Name is:   kheded
GPA:    3.40
No. Of Courses Enrolled:        5
Course Number 1 ID is 23
Course Number 2 ID is 64
Course Number 3 ID is 333
Course Number 4 ID is 33
Course Number 5 ID is 0
==========================================
==========Student==========
==========================================
ID:     14
First Name is:  ahmed
Last Name is:   kamek
GPA:    2.00
No. Of Courses Enrolled:        3
Course Number 1 ID is 33
Course Number 2 ID is 44
Course Number 3 ID is 50
==========================================
```

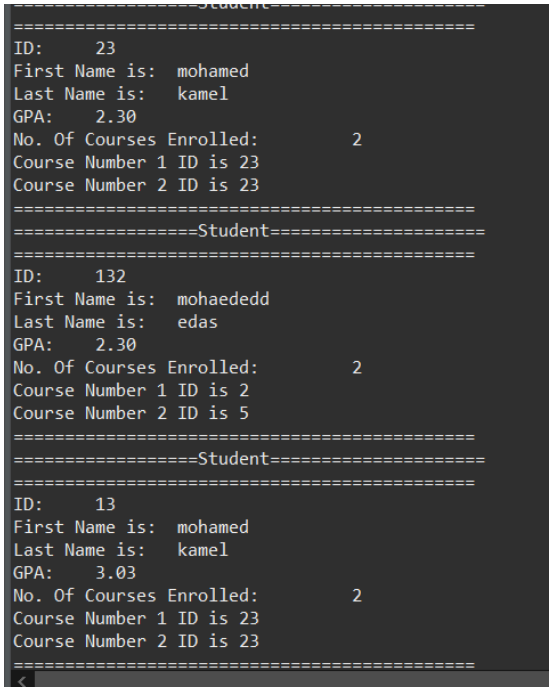
Entering a non-existing course will print the following message:

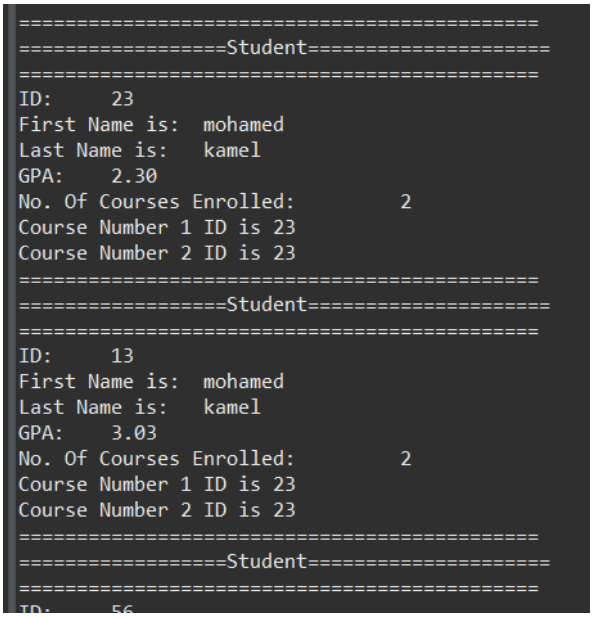
```
10: Exit The Program
6
==========================================
Enter Course ID to search:343
==================================
No Students found taking such course==========
```

## Delete Student By ID

In this test, we delete the student with the ID 132, the lower screenshots show the DB before and after the deletion

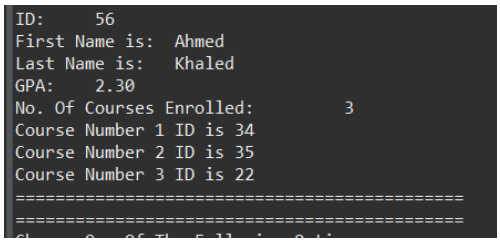


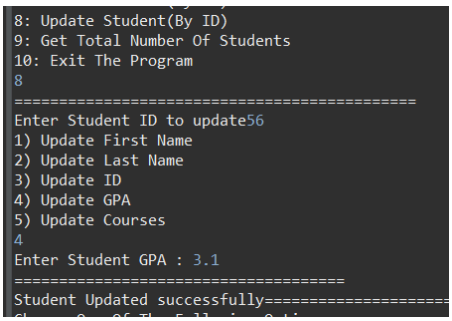



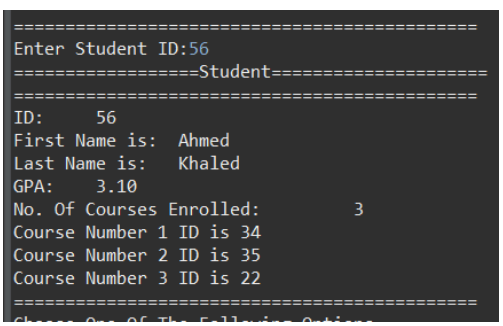




## Update Student By ID

The left-most Screen shows a student we will update, upon entering student ID, we choose what we want to update, the right-most screen shows that the student has been updated successfully



## Get Total Number Of Students

This functionality shows the number of students in our database