ASSIGNMENT 2 FRONT SHEET

Qualification	**BTEC Level 5 HND Diploma in Computing**		
Unit number and title	**PROG102: Procedural Programming**		
Submission date		**Date Received 1st submission**	
Re-submission Date		**Date Received 2nd submission**	
Student Name	**Nguyen Van Truong**	**Student ID**	**GCD201597**
Class	**GCD0905**	**Assessor name**	**Phan Thanh Tra**
Student declaration			

I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice.		
	Student's signature	**Truong NV**

Grading grid

P4	**P5**	**M3**	**M4**	**D2**

❐ Summative Feedback: Feedback:		❐ Resubmission
Grade:	Assessor Signature:	Date:
Lecturer Signature:		

Table of Contents

TABLE OF FIGURE

I.INTRODUCTION:

In the previous report, we learned the basic steps to create a point management program such as drawing a flowchart or a work breakdown structure. They are quite specific, created step by step. So, in this report, we will implement a specific program that is to write a complete score management program in C language. I use camelCase writing for all the cases where there are programs such as functions, variable names, ...

II.IMPLEMENTATION:

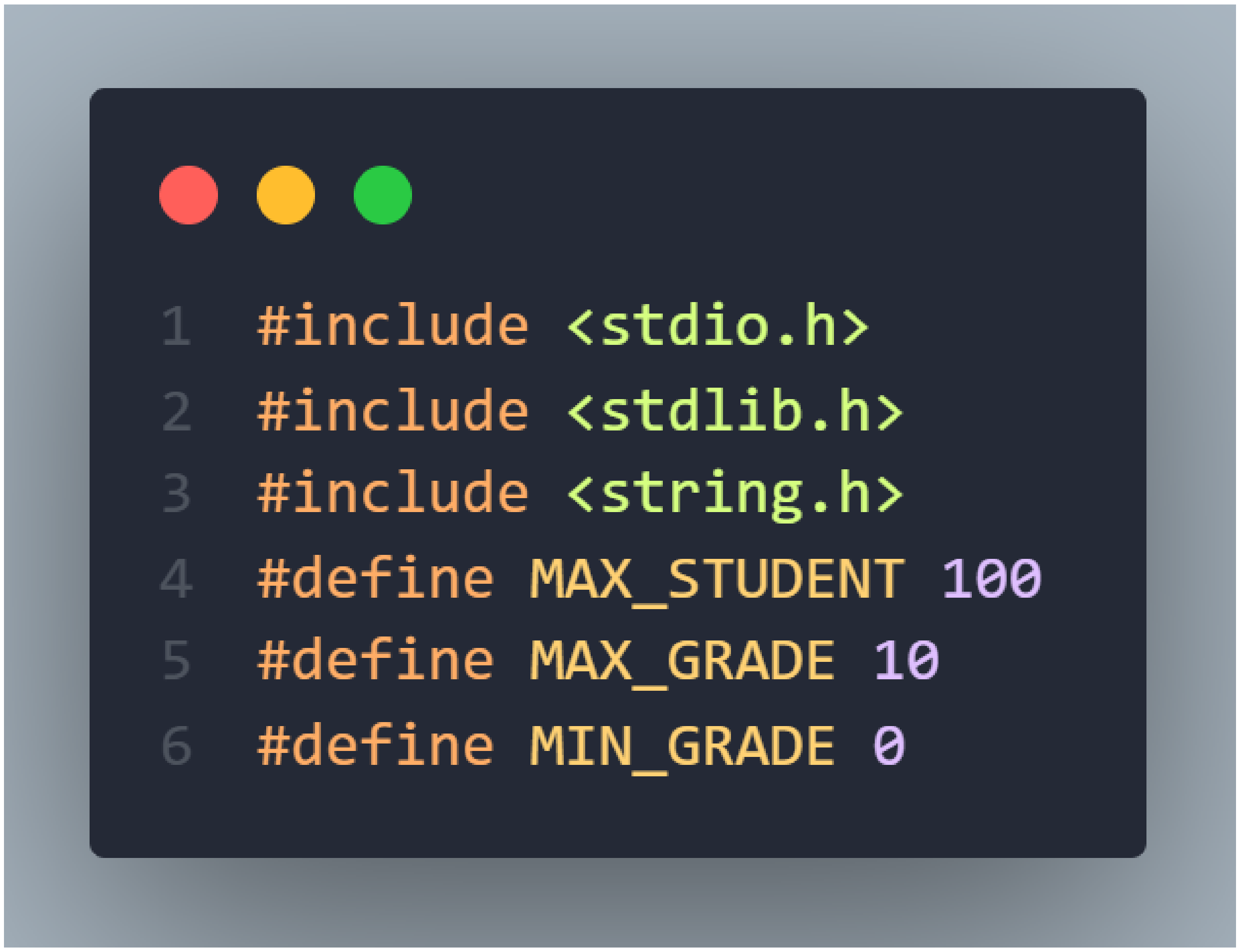
1.Structure of program:

1.1. Declaration:

- Library function:

In the program, I will use 3 types of C standard library functions, which are:

  - #include <stdio.h> : stands for Standard Input-Output. It contains information on input and output functions.
  - #include <string.h>: Contains one variable type, one macro, and a number of functions for manipulating character arrays.
  - #include <stdlib.h>: stands for Standard Library. It contains data on memory allocation and freeing functions.
  - #define is a function that allows naming an integer or a constant real number. When compiling will replace the constants you are using with their key values. This replacement is called pre-compile.



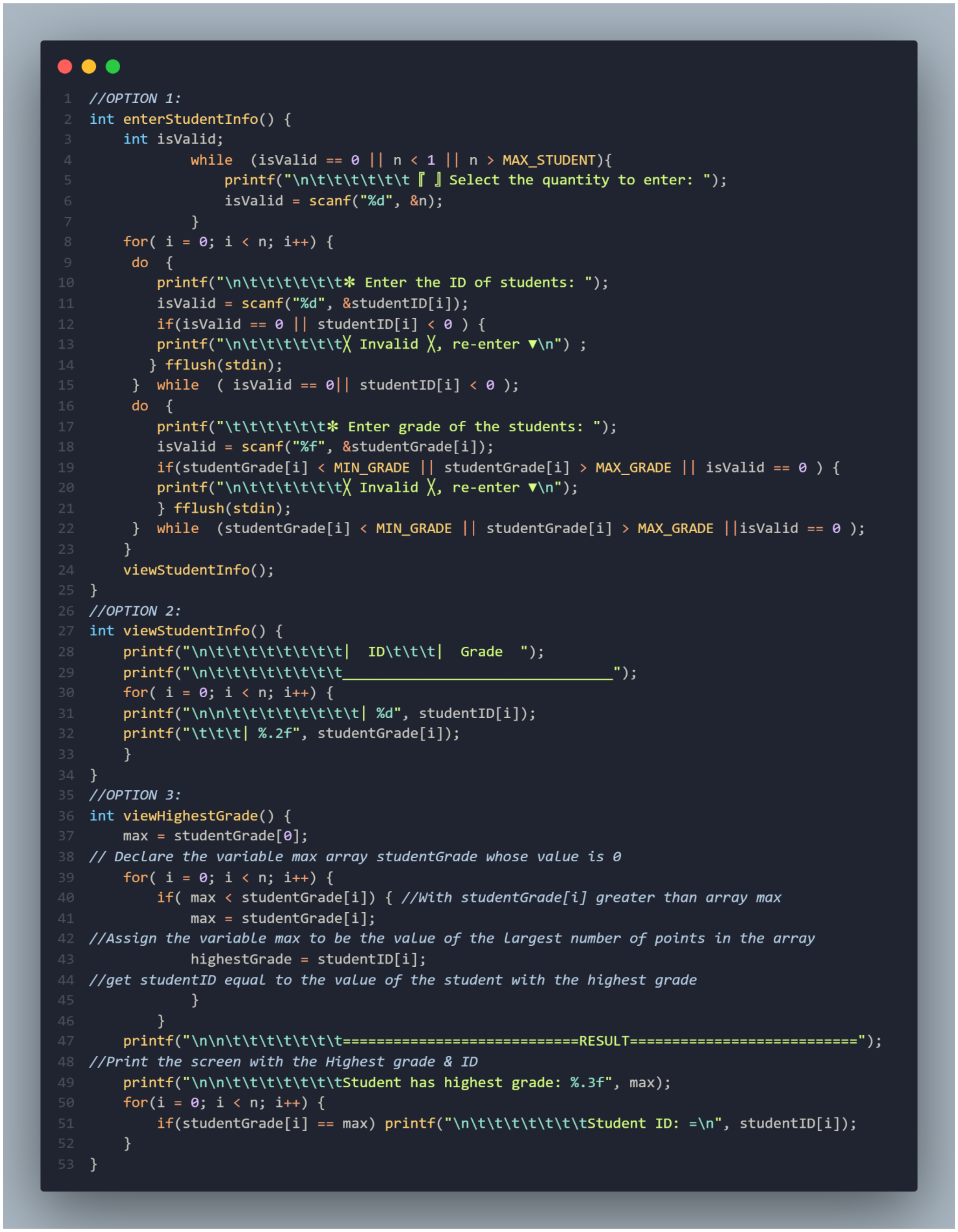
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAX_STUDENT 100
#define MAX_GRADE 10
#define MIN_GRADE 0
```

Figure 1: Standards library function [1]

and to explain it further, #include is interpreted as a preprocessor directory and is wrapped with curly braces around it.

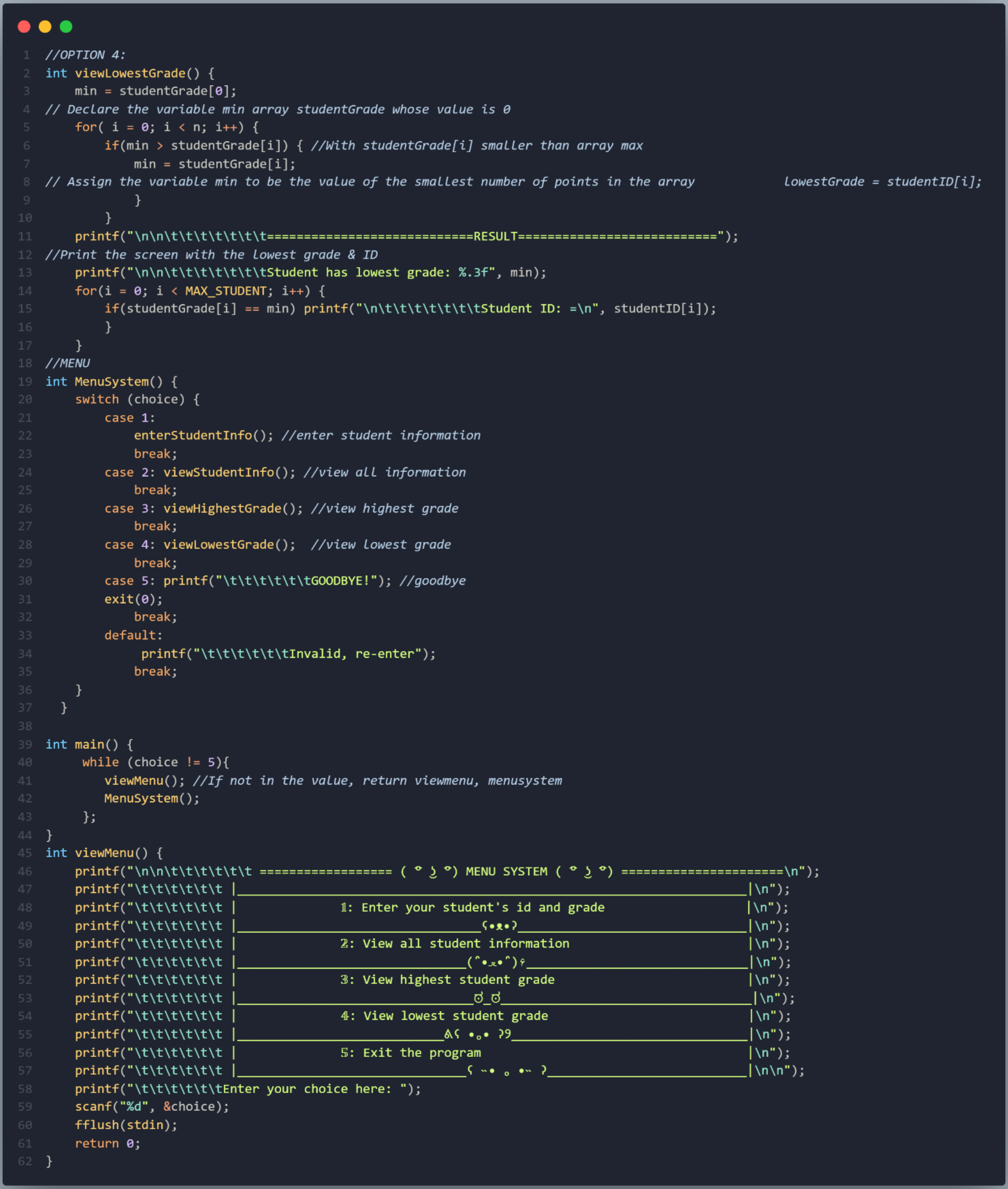
2.Function:

A function is a set of statements that take input, perform a specific calculation, and return a result. In the program, I use the function in any option in the program to optimize a particular function and make the program read the code faster. Below are the functions used in this lesson as well as all the complete functions of a score management program.



```c
//OPTION 1:
int enterStudentInfo() {
    int isValid;
            while  (isValid == 0 || n < 1 || n > MAX_STUDENT){
                printf("\n\t\t\t\t\t\t〖 〗Select the quantity to enter: ");
                isValid = scanf("%d", &n);
            }
    for( i = 0; i < n; i++) {
     do  {
        printf("\n\t\t\t\t\t\t\t✻ Enter the ID of students: ");
        isValid = scanf("%d", &studentID[i]);
        if(isValid == 0 || studentID[i] < 0 ) {
        printf("\n\t\t\t\t\t\t\t╳ Invalid ╳, re-enter ▼\n") ;
       } fflush(stdin);
     }  while  ( isValid == 0|| studentID[i] < 0 );
     do  {
        printf("\t\t\t\t\t\t\t✻ Enter grade of the students: ");
        isValid = scanf("%f", &studentGrade[i]);
        if(studentGrade[i] < MIN_GRADE || studentGrade[i] > MAX_GRADE || isValid == 0 ) {
        printf("\n\t\t\t\t\t\t\t╳ Invalid ╳, re-enter ▼\n");
        } fflush(stdin);
     }  while  (studentGrade[i] < MIN_GRADE || studentGrade[i] > MAX_GRADE ||isValid == 0 );
    }
    viewStudentInfo();
}
//OPTION 2:
int viewStudentInfo() {
    printf("\n\t\t\t\t\t\t\t\t|  ID\t\t\t|  Grade  ");
    printf("\n\t\t\t\t\t\t\t\t________________________________");
    for( i = 0; i < n; i++) {
    printf("\n\n\t\t\t\t\t\t\t\t| %d", studentID[i]);
    printf("\t\t\t| %.2f", studentGrade[i]);
    }
}
//OPTION 3:
int viewHighestGrade() {
    max = studentGrade[0];
// Declare the variable max array studentGrade whose value is 0
    for( i = 0; i < n; i++) {
        if( max < studentGrade[i]) { //With studentGrade[i] greater than array max
            max = studentGrade[i];
//Assign the variable max to be the value of the largest number of points in the array
            highestGrade = studentID[i];
//get studentID equal to the value of the student with the highest grade
            }
        }
    printf("\n\n\t\t\t\t\t\t\t\t===========================RESULT===========================");
//Print the screen with the Highest grade & ID
    printf("\n\n\t\t\t\t\t\t\t\tStudent has highest grade: %.3f", max);
    for(i = 0; i < n; i++) {
        if(studentGrade[i] == max) printf("\n\t\t\t\t\t\t\tStudent ID: =\n", studentID[i]);
    }
}
```

Figure 2: Function [2]

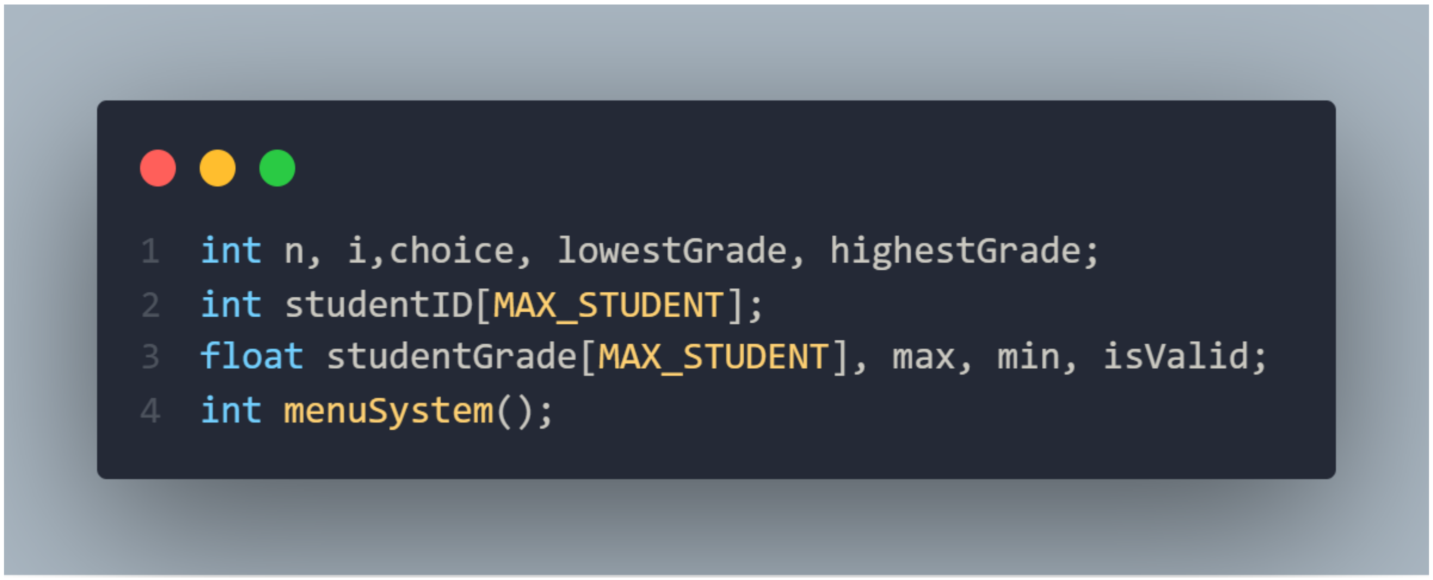


```c
//OPTION 4:
int viewLowestGrade() {
    min = studentGrade[0];
// Declare the variable min array studentGrade whose value is 0
    for( i = 0; i < n; i++) {
        if(min > studentGrade[i]) { //With studentGrade[i] smaller than array max
            min = studentGrade[i];
// Assign the variable min to be the value of the smallest number of points in the array            lowestGrade = studentID[i];
            }
        }
    printf("\n\n\t\t\t\t\t\t\t============================RESULT===========================");
//Print the screen with the lowest grade & ID
    printf("\n\n\t\t\t\t\t\t\tStudent has lowest grade: %.3f", min);
    for(i = 0; i < MAX_STUDENT; i++) {
        if(studentGrade[i] == min) printf("\n\t\t\t\t\t\t\tStudent ID: =\n", studentID[i]);
        }
    }
//MENU
int MenuSystem() {
    switch (choice) {
        case 1:
            enterStudentInfo(); //enter student information
            break;
        case 2: viewStudentInfo(); //view all information
            break;
        case 3: viewHighestGrade(); //view highest grade
            break;
        case 4: viewLowestGrade();  //view lowest grade
            break;
        case 5: printf("\t\t\t\t\t\tGOODBYE!"); //goodbye
        exit(0);
            break;
        default:
             printf("\t\t\t\t\t\tInvalid, re-enter");
            break;
    }
  }

int main() {
     while (choice != 5){
        viewMenu(); //If not in the value, return viewmenu, menusystem
        MenuSystem();
     };
}
int viewMenu() {
    printf("\n\n\t\t\t\t\t\t ================== ( ͡° ͜ʖ ͡°) MENU SYSTEM ( ͡° ͜ʖ ͡°) ======================\n");
    printf("\t\t\t\t\t\t |_____________________________________________________________________|\n");
    printf("\t\t\t\t\t\t |              1: Enter your student's id and grade                   |\n");
    printf("\t\t\t\t\t\t |_________________________________ʕ•ᴥ•ʔ_______________________________|\n");
    printf("\t\t\t\t\t\t |              2: View all student information                        |\n");
    printf("\t\t\t\t\t\t |_______________________________(ᵔ•ᴥ•ᵔ)ﻭ______________________________|\n");
    printf("\t\t\t\t\t\t |              3: View highest student grade                          |\n");
    printf("\t\t\t\t\t\t |________________________________ಠ_ಠ_________________________________|\n");
    printf("\t\t\t\t\t\t |              4: View lowest student grade                           |\n");
    printf("\t\t\t\t\t\t |____________________________ᕕʕ •ₒ• ʔ୨_______________________________|\n");
    printf("\t\t\t\t\t\t |              5: Exit the program                                    |\n");
    printf("\t\t\t\t\t\t |_______________________________ʕ ˵• ₒ •˵ ʔ___________________________|\n\n");
    printf("\t\t\t\t\t\tEnter your choice here: ");
    scanf("%d", &choice);
    fflush(stdin);
    return 0;
}
```

Figure 3: Function [3]

- Variable:

Many of you ask me, why don't use void main () function but use int main () function, the reason int main () function is because they are ANSI C standard. The function will return the value within the int variable. Also, in the case of integers, we use int in C programming. But when our program is simple and it won't finish before reaching the last line of code or error-free code, we can use void main ().



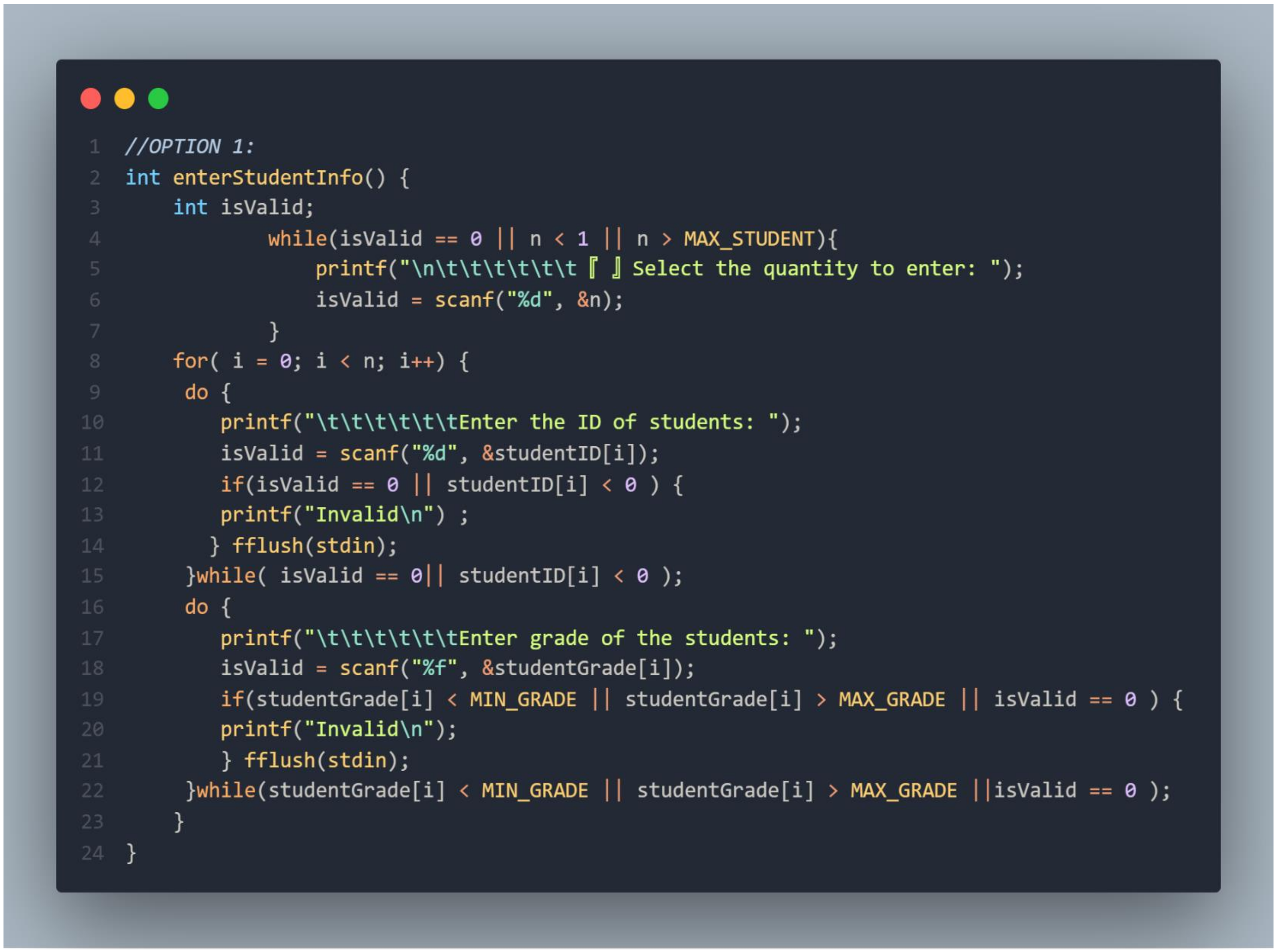
```
int n, i,choice, lowestGrade, highestGrade;
int studentID[MAX_STUDENT];
float studentGrade[MAX_STUDENT], max, min, isValid;
int menuSystem();
```

Figure 4: Variable [6]

2.Function:

1.1. “Enter student information” function:

- Purpose: Used to perform the input of student information including grades and IDs.
- Steps to take:
    - To execute the program, we initialize the isValid variable whose task is to check the existence condition contained in the function.
    - Since the range of user need to enter the number of all students to enter is in the range from 0 to 100 we need to use a while loop for the variable levels that will perform that action. The advantage of the while loop is that it checks the condition before it will repeat the code.
    - We will use a for loop to iterate over the correct number of students to enter. Inside, is the do-while function to check the allowed exact range.
    - In the ID input section, if the user enters a value of 0 and student[ID] is less than 0, the computer will return "invalid".
    - In order to meet the criteria that the provided score is within the range of 0 to 10, the user must input the student's score accurately. When a user submits an incorrect grade, the system returns "invalid" and returns to the previous re-entry of the grade.
    - When all conditions are satisfied, the program will print the score and ID

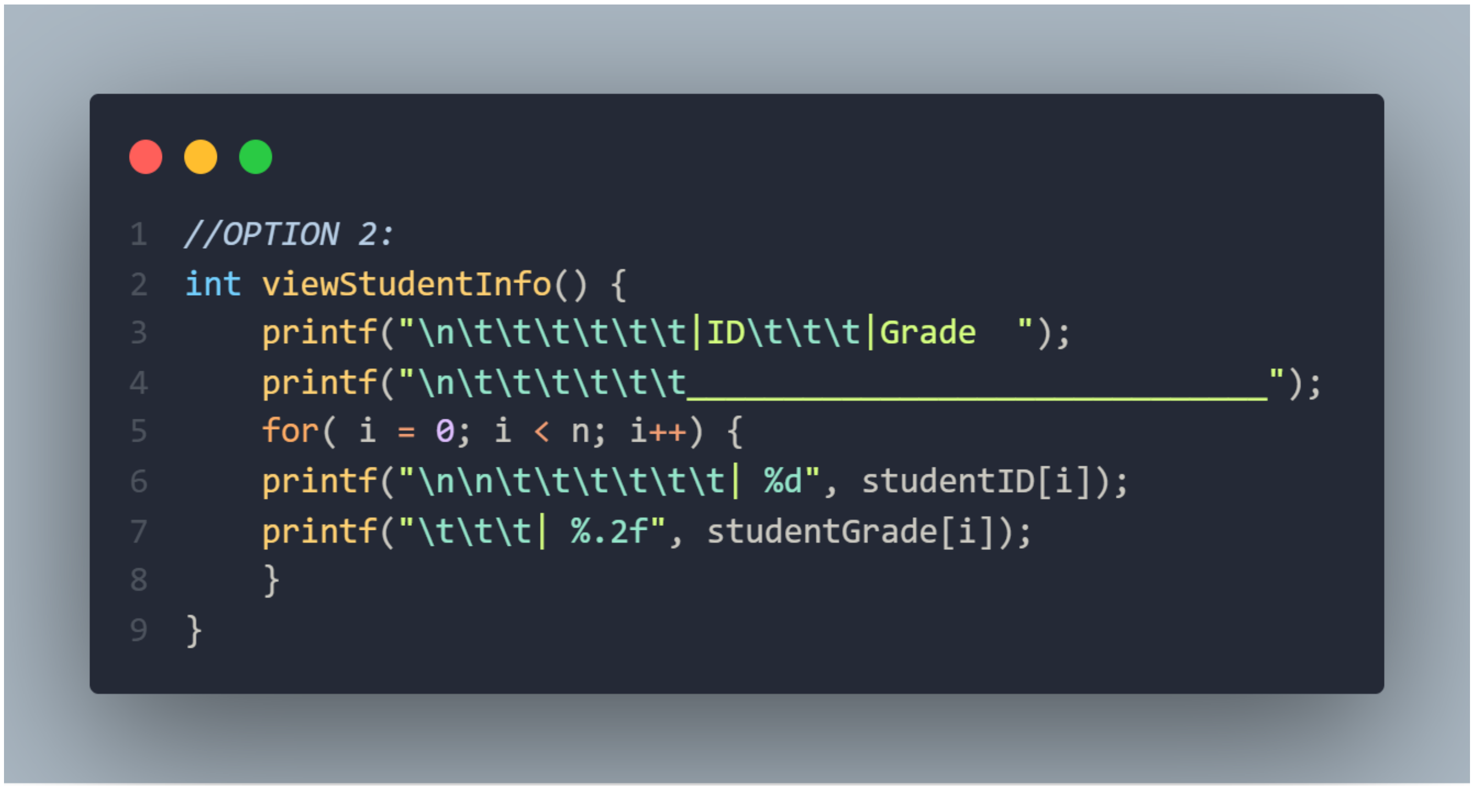


```c
//OPTION 1:
int enterStudentInfo() {
    int isValid;
            while(isValid == 0 || n < 1 || n > MAX_STUDENT){
                printf("\n\t\t\t\t\t\t 〚 〛Select the quantity to enter: ");
                isValid = scanf("%d", &n);
            }
    for( i = 0; i < n; i++) {
     do {
        printf("\t\t\t\t\t\tEnter the ID of students: ");
        isValid = scanf("%d", &studentID[i]);
        if(isValid == 0 || studentID[i] < 0 ) {
        printf("Invalid\n") ;
       } fflush(stdin);
     }while( isValid == 0|| studentID[i] < 0 );
     do {
        printf("\t\t\t\t\t\tEnter grade of the students: ");
        isValid = scanf("%f", &studentGrade[i]);
        if(studentGrade[i] < MIN_GRADE || studentGrade[i] > MAX_GRADE || isValid == 0 ) {
        printf("Invalid\n");
        } fflush(stdin);
     }while(studentGrade[i] < MIN_GRADE || studentGrade[i] > MAX_GRADE ||isValid == 0 );
    }
}
```

Figure 5: “Enter student information” function [8]

1.2. “View student information” function:

- Purpose: Used to view student grades and information.
- Steps to take:
  - In this feature, we use a for loop to help the program display full student information. As well as using the printf() command to output the score and ID, we use %d and %f to perform printing in the command. Where %d is used to print integers and %d is used to print real numbers.



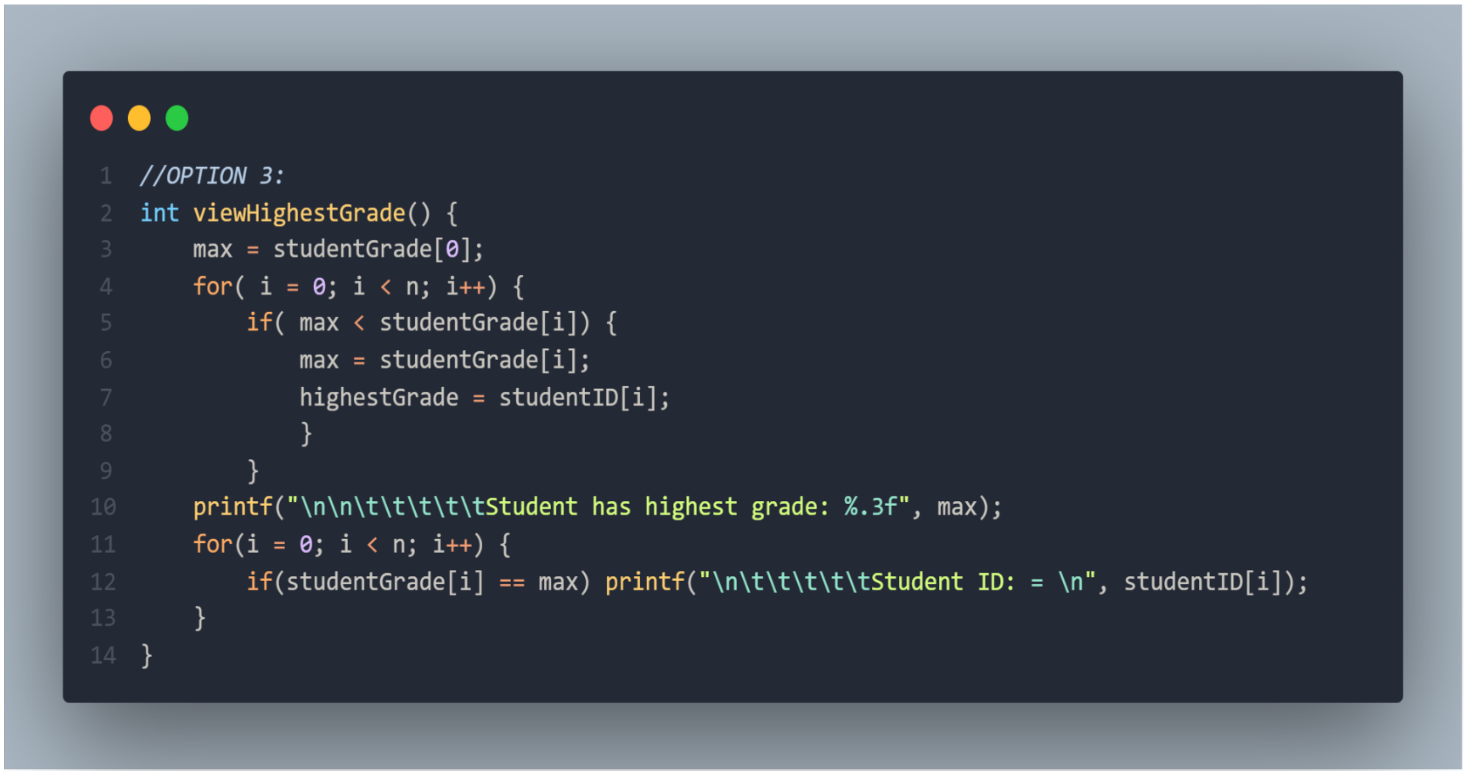
```c
//OPTION 2:
int viewStudentInfo() {
    printf("\n\t\t\t\t\t\t|ID\t\t\t|Grade  ");
    printf("\n\t\t\t\t\t\t______________________________");
    for( i = 0; i < n; i++) {
    printf("\n\n\t\t\t\t\t\t| %d", studentID[i]);
    printf("\t\t\t| %.2f", studentGrade[i]);
    }
}
```

Figure 6: "View student information” function [9]

1.3. “View highest student grade” function:

- Purpose: Used to view the highest grade in the list
- Steps to take:
  - We make a declaration of the max sea, there is a studentGrade array with 0 as an integer element. Then we make the loop to repeat the condition below.

- ○ In the condition, when the max array is less than the studentGrade array, it means we reassign the i element. After completing the assignment and comparison task, we will print the maximum point and ID

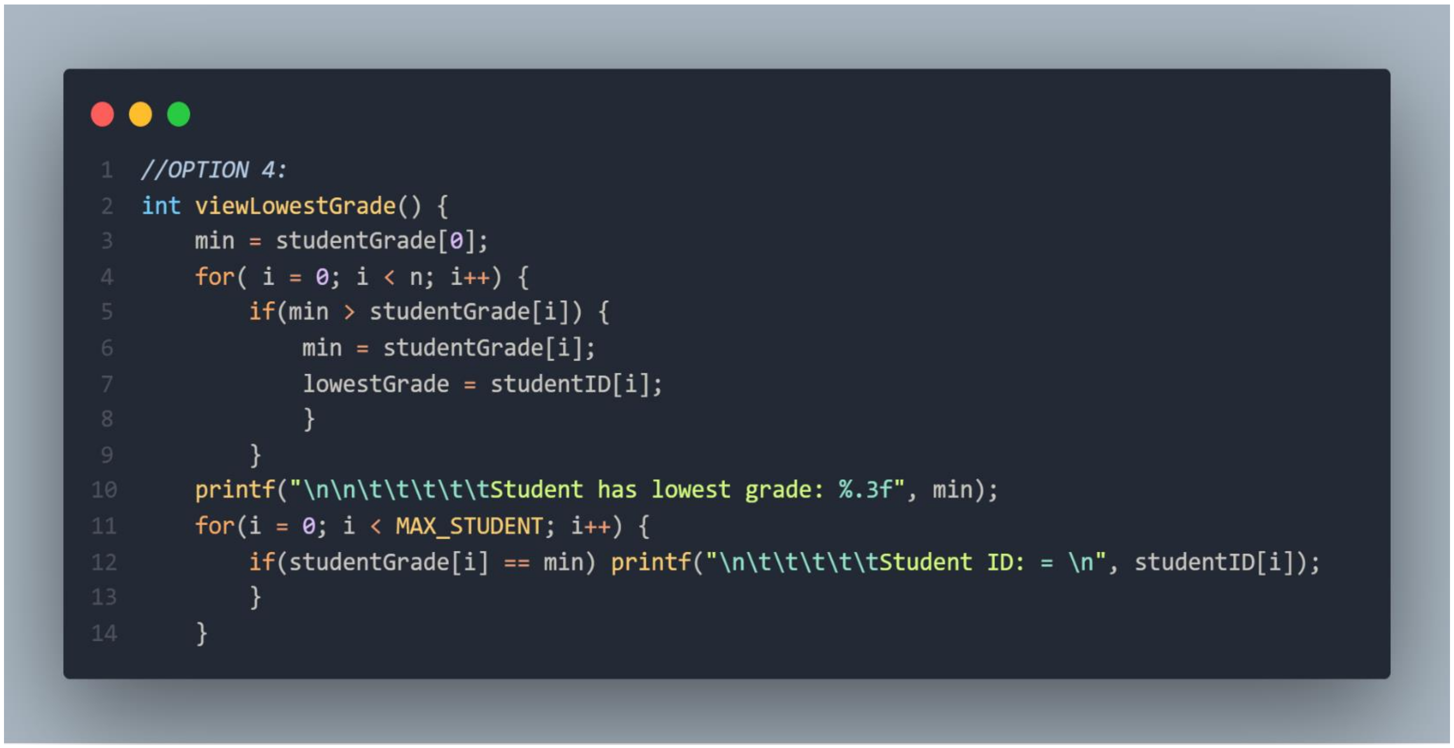


```c
//OPTION 3:
int viewHighestGrade() {
    max = studentGrade[0];
    for( i = 0; i < n; i++) {
        if( max < studentGrade[i]) {
            max = studentGrade[i];
            highestGrade = studentID[i];
            }
        }
    printf("\n\n\t\t\t\t\tStudent has highest grade: %.3f", max);
    for(i = 0; i < n; i++) {
        if(studentGrade[i] == max) printf("\n\t\t\t\t\tStudent ID: = \n", studentID[i]);
    }
}
```

Figure 7: "View highest student grade" function [10]

1.4. "View lowest student grade" function:

- Purpose: Used to view the lowest grade in the list
- Steps to take:
  - ○ Similar to finding the highest grade, we create the studentGrade array with 0 as an integer element. Then we create a loop to repeat the condition below.
  - ○ When min array is larger than studentGrade array, that means we reassign element i. After completing the assignment and comparison task, we will print the highest student and ID.

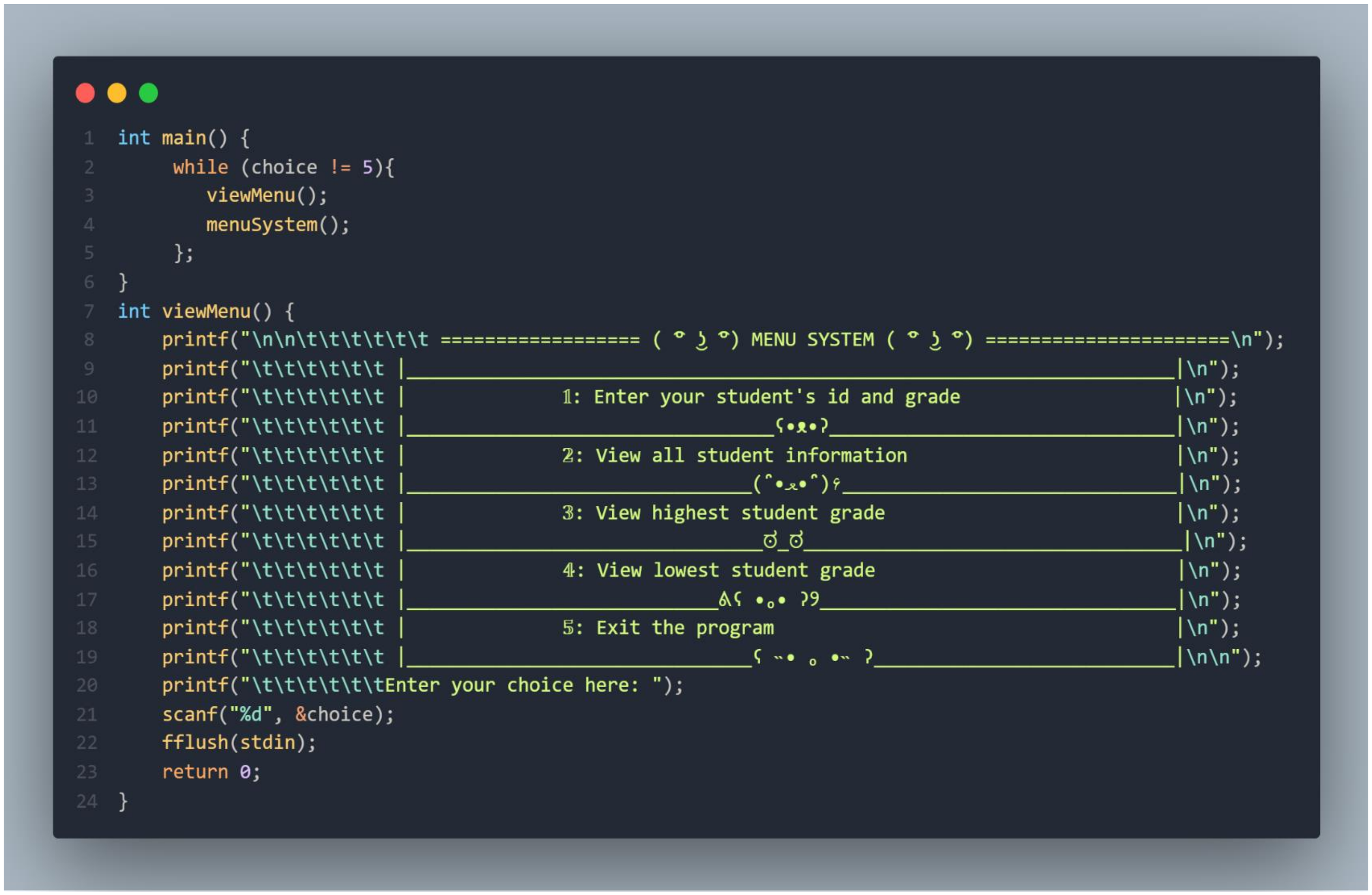


```c
//OPTION 4:
int viewLowestGrade() {
    min = studentGrade[0];
    for( i = 0; i < n; i++) {
        if(min > studentGrade[i]) {
            min = studentGrade[i];
            lowestGrade = studentID[i];
            }
        }
    printf("\n\n\t\t\t\t\tStudent has lowest grade: %.3f", min);
    for(i = 0; i < MAX_STUDENT; i++) {
        if(studentGrade[i] == min) printf("\n\t\t\t\t\tStudent ID: = \n", studentID[i]);
        }
    }
```

Figure 8:"View lowest student grade" [11]

1.5. "View menu" function:

- Purpose: Used to view the lowest grade in the list.
- Steps to take:
  - The printf statement in this function is used to print all the information the user needs in the article. We use the scanf command to receive input from the user into the program, finally fflush is used to clear the program's temporary memory so that there are no garbage values like above and we have can use cin.ignore() as another option.



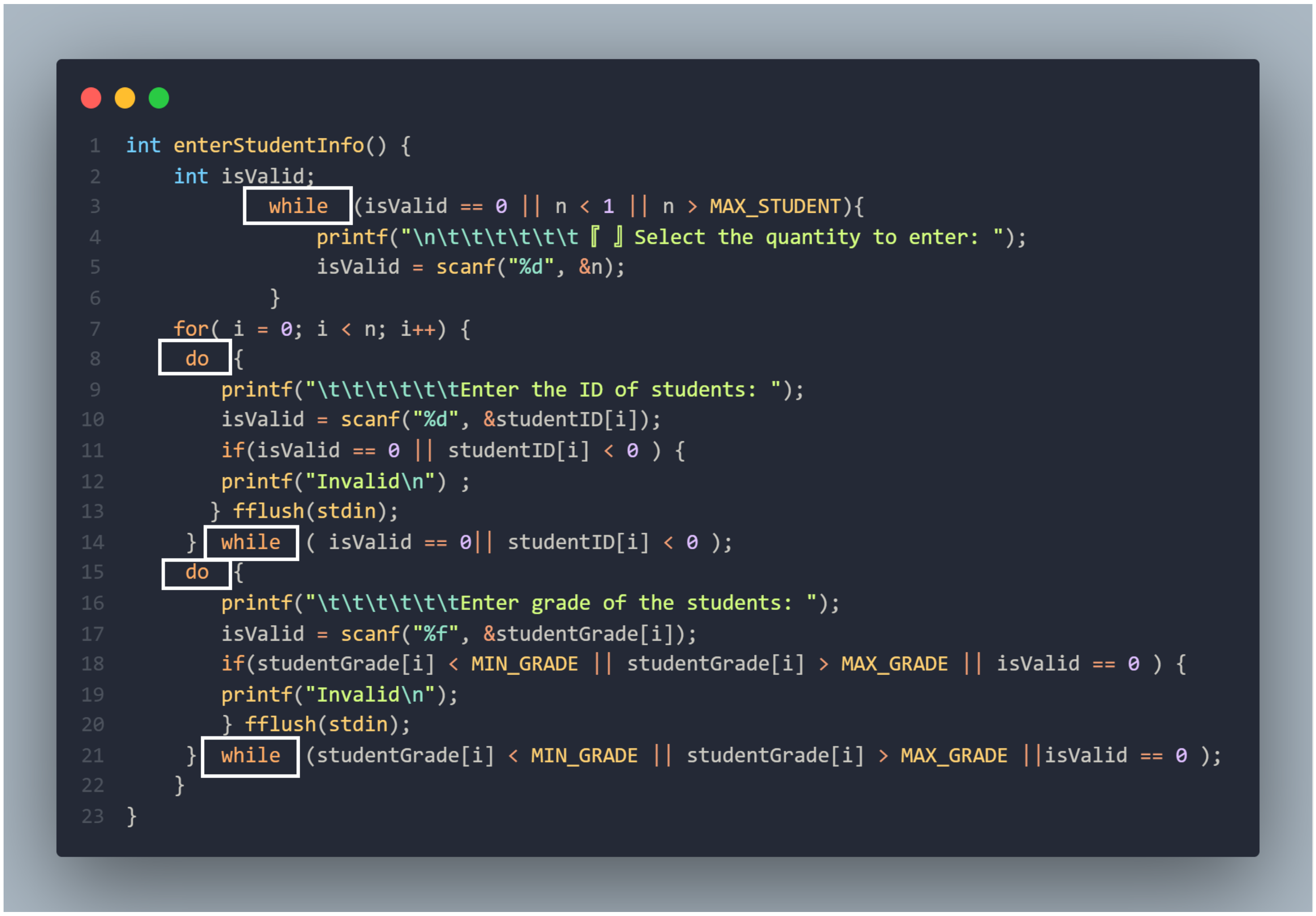
```c
int main() {
     while (choice != 5){
        viewMenu();
        menuSystem();
     };
}
int viewMenu() {
    printf("\n\n\t\t\t\t\t\t ================== ( ͡° ͜ʖ ͡°) MENU SYSTEM ( ͡° ͜ʖ ͡°) =====================\n");
    printf("\t\t\t\t\t\t |______________________________________________________________________|\n");
    printf("\t\t\t\t\t\t |              1: Enter your student's id and grade                    |\n");
    printf("\t\t\t\t\t\t |_________________________________ʕ•ᴥ•ʔ________________________________|\n");
    printf("\t\t\t\t\t\t |              2: View all student information                         |\n");
    printf("\t\t\t\t\t\t |________________________________(ˆ•ﻌ•ˆ)ϩ______________________________|\n");
    printf("\t\t\t\t\t\t |              3: View highest student grade                           |\n");
    printf("\t\t\t\t\t\t |________________________________ಠ_ಠ__________________________________|\n");
    printf("\t\t\t\t\t\t |              4: View lowest student grade                            |\n");
    printf("\t\t\t\t\t\t |____________________________ᕦʕ •ₒ• ʔ9________________________________|\n");
    printf("\t\t\t\t\t\t |              5: Exit the program                                     |\n");
    printf("\t\t\t\t\t\t |________________________________ʕ ˵• ₒ •˵ ʔ__________________________|\n\n");
    printf("\t\t\t\t\t\tEnter your choice here: ");
    scanf("%d", &choice);
    fflush(stdin);
    return 0;
}
```

Figure 9: "View menu" function [12]

3. Algorithms:

Algorithm characteristics in C language are quite simple and clear, but the implementation of an algorithm needs to have two stages, namely "A prior analysis" and "A Prior Analysis", the program code must be logically and properly arranged for the program to execute.

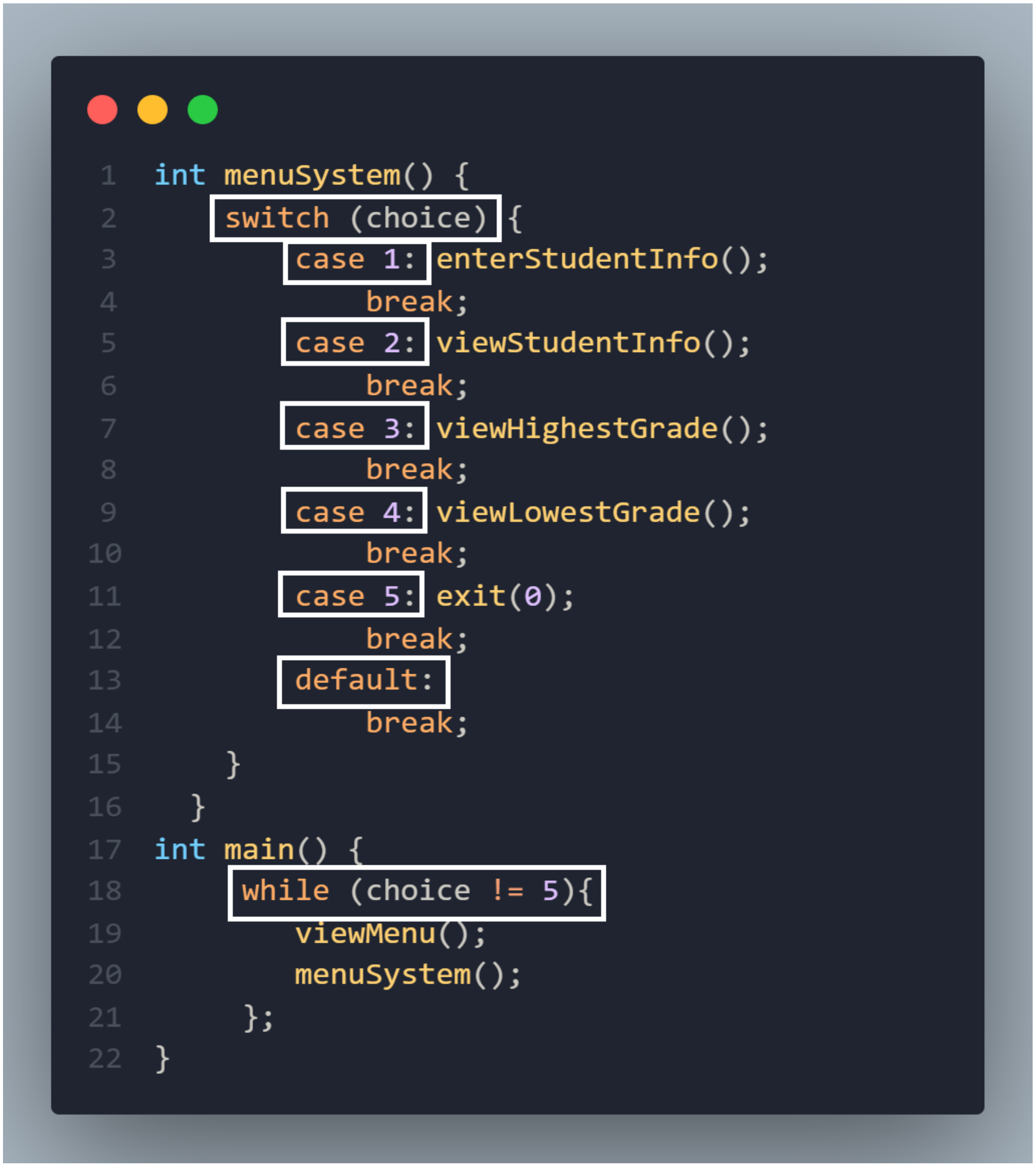
In the program, we use while and do-while loop to execute options in the program. Their advantage is that the code can be reused without rewriting. In the example they are placed before commands like entering score and ID, to ensure that the input will be reasonable and correct to the user request, when the condition is false, the program will stop the loop and return to the next part. before so that the user can use re-enter.



```c
int enterStudentInfo() {
    int isValid;
            while  (isValid == 0 || n < 1 || n > MAX_STUDENT){
                printf("\n\t\t\t\t\t\t 『 』Select the quantity to enter: ");
                isValid = scanf("%d", &n);
            }
    for( i = 0; i < n; i++) {
     do  {
        printf("\t\t\t\t\t\tEnter the ID of students: ");
        isValid = scanf("%d", &studentID[i]);
        if(isValid == 0 || studentID[i] < 0 ) {
        printf("Invalid\n") ;
       } fflush(stdin);
     } while  ( isValid == 0|| studentID[i] < 0 );
     do  {
        printf("\t\t\t\t\t\tEnter grade of the students: ");
        isValid = scanf("%f", &studentGrade[i]);
        if(studentGrade[i] < MIN_GRADE || studentGrade[i] > MAX_GRADE || isValid == 0 ) {
        printf("Invalid\n");
        } fflush(stdin);
     } while  (studentGrade[i] < MIN_GRADE || studentGrade[i] > MAX_GRADE ||isValid == 0 );
    }
}
```

Figure 10: Algorithm [13]

The switch case statement is a branching conditional statement that replaces the if else structure in the case of multiple cases. Using a switch case will make our code easier to write and read. In the program, there will be many options from 1 to 5 in the menu. For switch cases, they are special that there is a default case that will be executed if no case matches the value of the condition and returns an error stating that the user entered it incorrectly.



```
int menuSystem() {
    switch (choice) {
        case 1: enterStudentInfo();
            break;
        case 2: viewStudentInfo();
            break;
        case 3: viewHighestGrade();
            break;
        case 4: viewLowestGrade();
            break;
        case 5: exit(0);
            break;
        default:
            break;
    }
  }
int main() {
     while (choice != 5){
        viewMenu();
        menuSystem();
      };
}
```

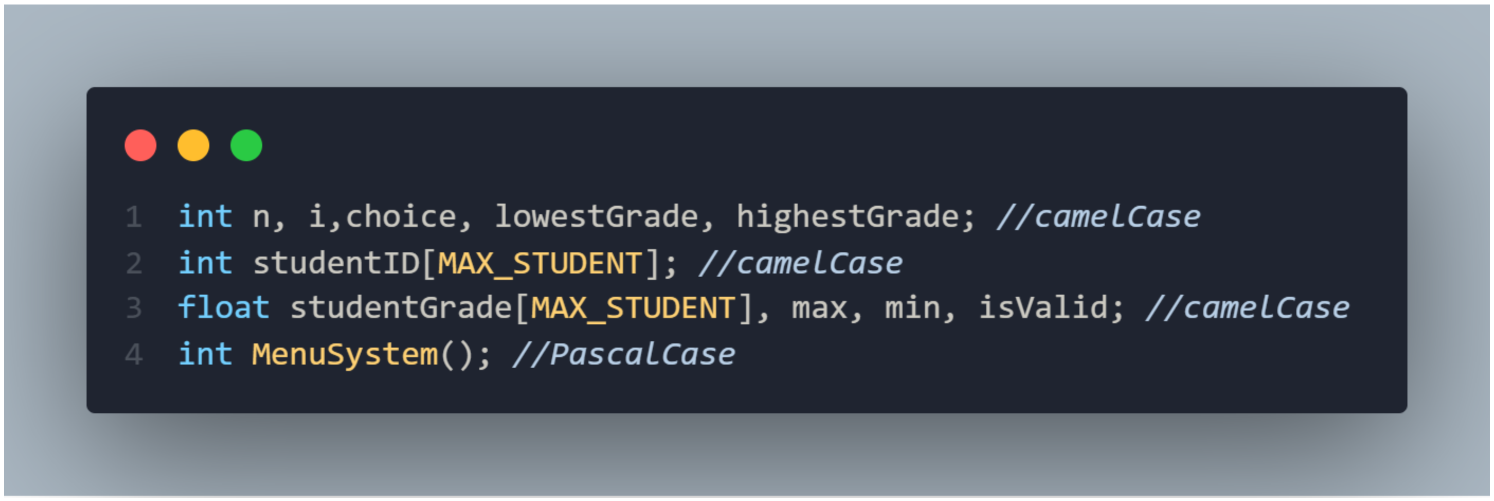
Figure 11: Switch case [14]

3. Coding standards:

Coding standards is a set of rules that regulate how to write the code of a program that programmers must follow when participating in a project to develop that program. There are different standards for each project. A set of rules has many types and the types we often encounter are the rules for naming variables, constants, functions, classes, ... or declare and use variables. Standards compliance helps us know what we're doing, and what's happening with lines of code. It should be easy to find and correct mistakes.

- Naming conventions for etc.:

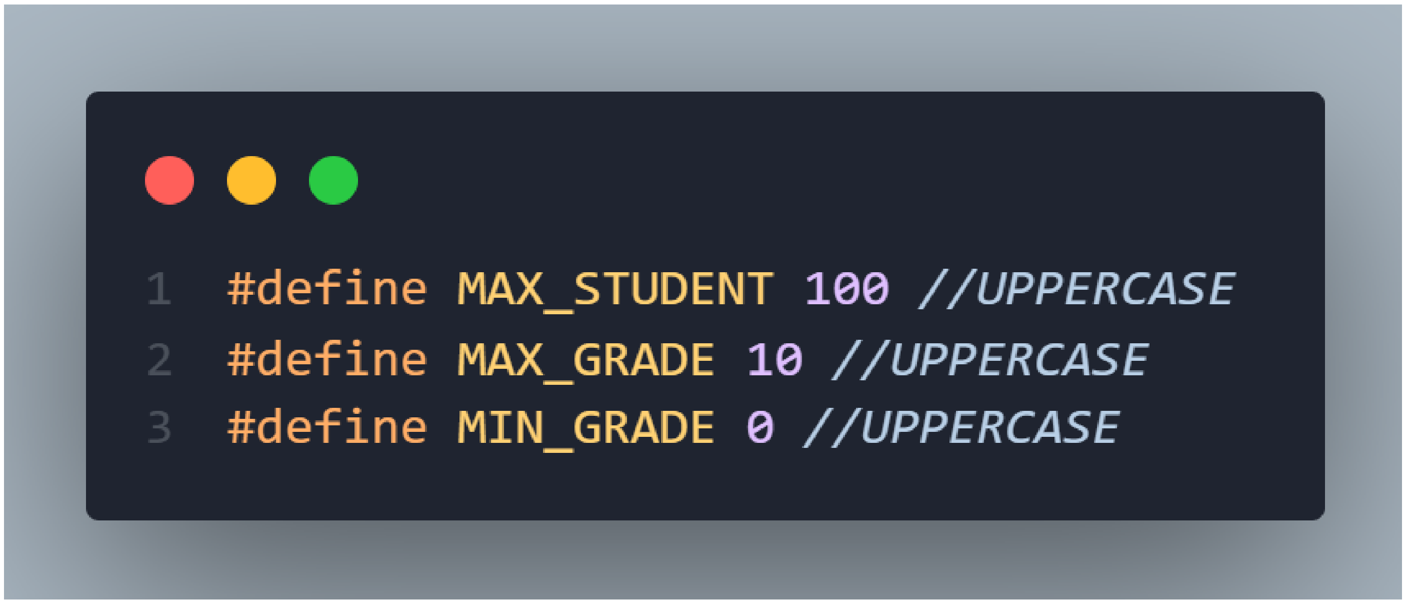
In the program, I use standard Camelcase writing. It is a naming convention for writing file or object names that contain at least one compound word or concatenation that begins with a capital letter. It is often used to name files and functions without violating the naming rules of the underlying language. It is useful in programming because element names cannot contain spaces. In the program, I also use PascalCase writing style for components in C, for example, menus,...



```c
int n, i,choice, lowestGrade, highestGrade; //camelCase
int studentID[MAX_STUDENT]; //camelCase
float studentGrade[MAX_STUDENT], max, min, isValid; //camelCase
int MenuSystem(); //PascalCase
```

Figure 12: camelCase naming style in C [15]

As for the UPPER_CASE type, we often use them in the case of constants, which need to be used for many purposes in the code.

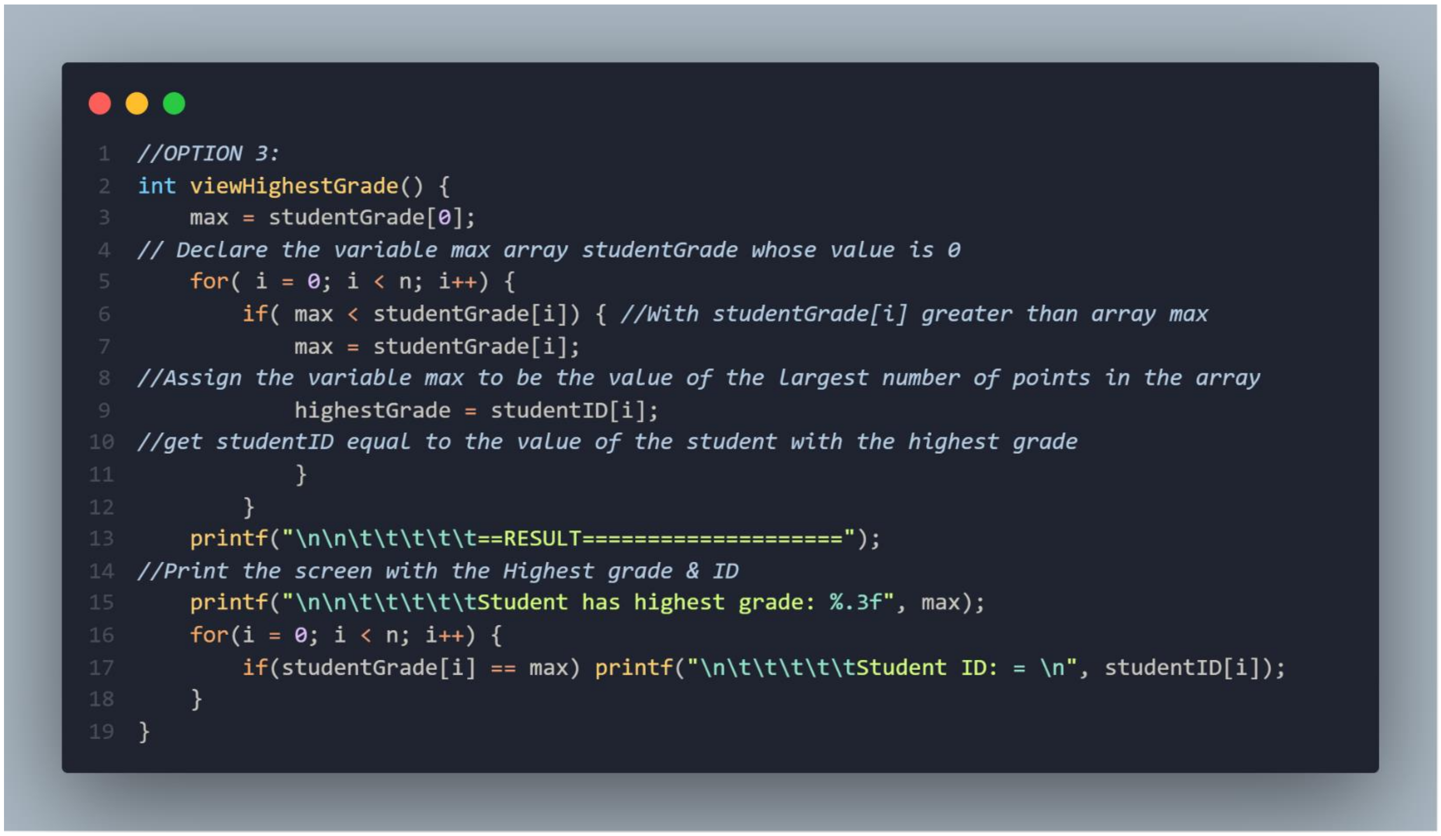


```c
#define MAX_STUDENT 100 //UPPERCASE
#define MAX_GRADE 10 //UPPERCASE
#define MIN_GRADE 0 //UPPERCASE
```

Figure 13: UPPER_CASE naming style in C [16]

- Comment:

  In computer programming, a comment is an explanation or comment that can be read by the programmer in the source code of a computer program. Their purpose is to help the reader understand what the code will do. All characters available inside any comment are ignored by the C compiler. In the program I only use one-line comments with syntax starting with //, extending to the end of the line.

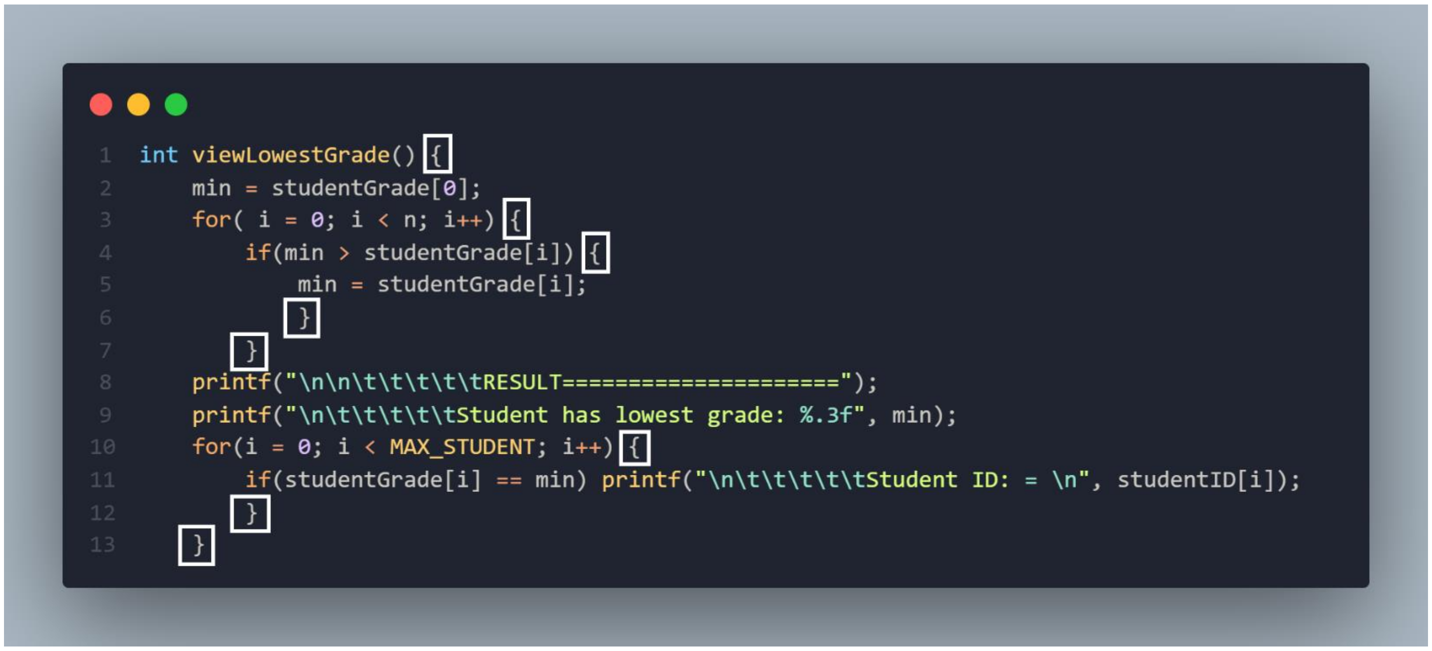


```c
//OPTION 3:
int viewHighestGrade() {
    max = studentGrade[0];
// Declare the variable max array studentGrade whose value is 0
    for( i = 0; i < n; i++) {
        if( max < studentGrade[i]) { //With studentGrade[i] greater than array max
            max = studentGrade[i];
//Assign the variable max to be the value of the largest number of points in the array
            highestGrade = studentID[i];
//get studentID equal to the value of the student with the highest grade
            }
        }
    printf("\n\n\t\t\t\t\t==RESULT====================");
//Print the screen with the Highest grade & ID
    printf("\n\n\t\t\t\t\tStudent has highest grade: %.3f", max);
    for(i = 0; i < n; i++) {
        if(studentGrade[i] == max) printf("\n\t\t\t\t\tStudent ID: = \n", studentID[i]);
    }
}
```

Figure 14: Comment [17]

- Bracket conversions:

  In C programs, we often use {} in if-else conditional statements to surround a piece of code, called a code block. If after for, if or while you have multiple statements, we use {} and there is only one statement then we don't need to use

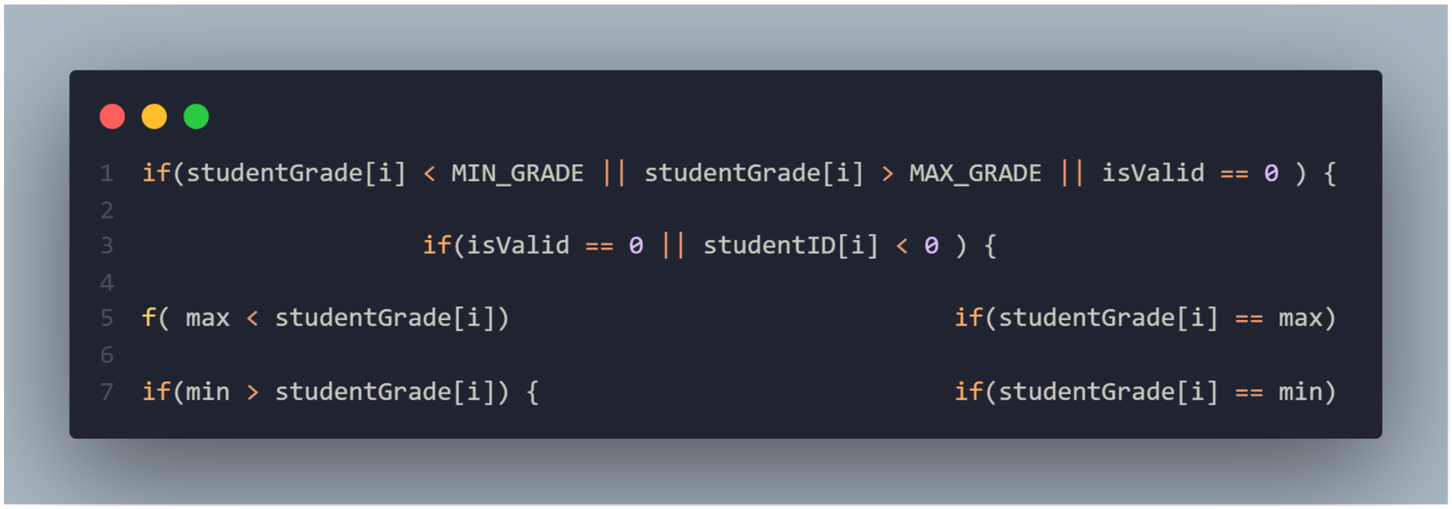


```c
int viewLowestGrade() {
    min = studentGrade[0];
    for( i = 0; i < n; i++) {
        if(min > studentGrade[i]) {
            min = studentGrade[i];
        }
    }
    printf("\n\n\t\t\t\t\tRESULT====================");
    printf("\n\t\t\t\t\tStudent has lowest grade: %.3f", min);
    for(i = 0; i < MAX_STUDENT; i++) {
        if(studentGrade[i] == min) printf("\n\t\t\t\t\tStudent ID: = \n", studentID[i]);
    }
}
```

Figure 15: Bracket conversion [18]

- Conditional statement:

  A conditional statement is a statement that is used to verify any condition based on the user's request. These conditional expressions in C allow us to make a program judgment in any case. In the program, we always use conditional statements to remove invalid results from user input.



```c
if(studentGrade[i] < MIN_GRADE || studentGrade[i] > MAX_GRADE || isValid == 0 ) {

                    if(isValid == 0 || studentID[i] < 0 ) {

f( max < studentGrade[i])                               if(studentGrade[i] == max)

if(min > studentGrade[i]) {                             if(studentGrade[i] == min)
```

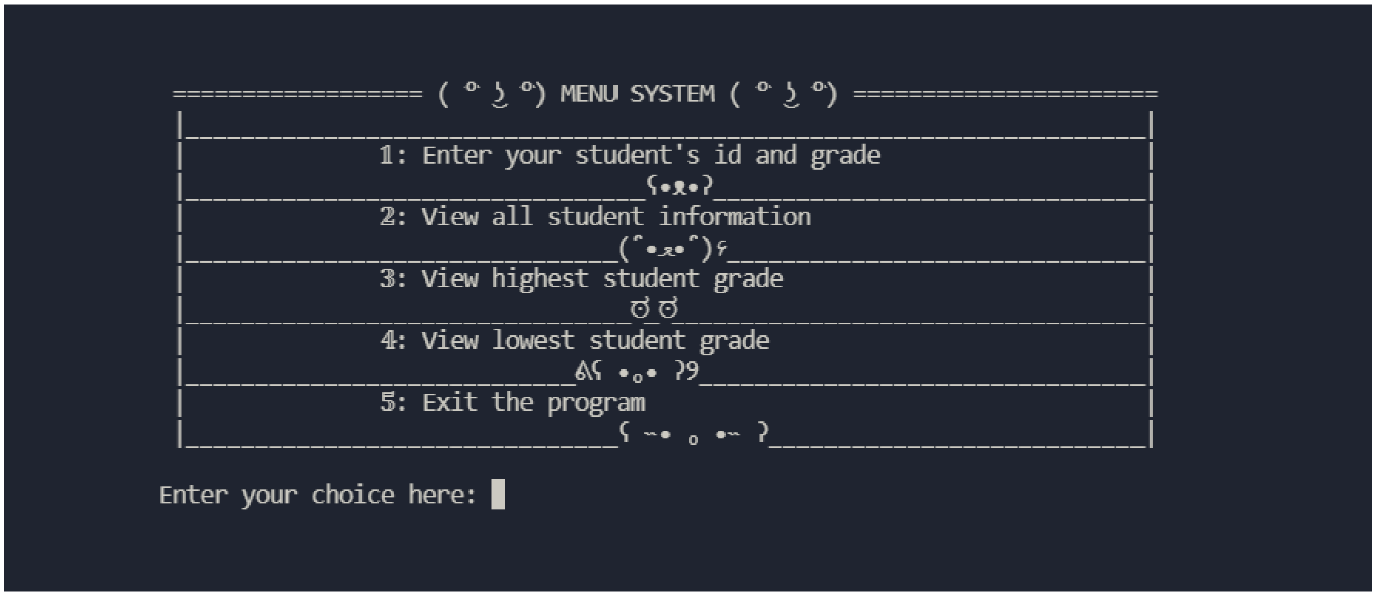
Figure 16: Conditional statement [19]

II.PROGRAM RESULT:

1. Result of the option 1:

This is the menu of the program, designed with a cute and easy-to-use interface, suitable for all basic users. The user will perform the appropriate data entry. To start, the system will show you all 5 function options and we will use one of these to start the function.

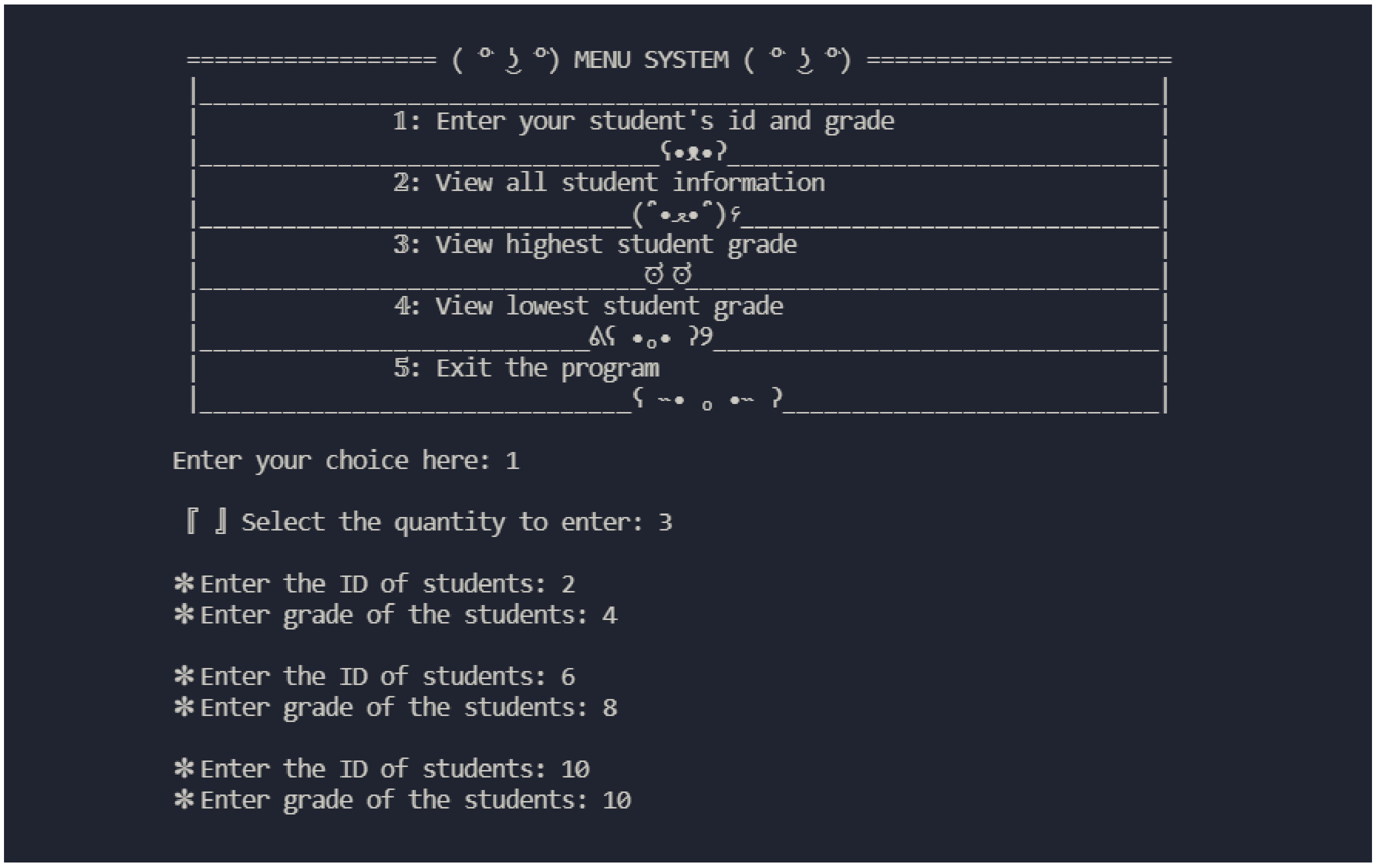
Example: When you select option 2, you will get a list of all student IDs and grades.



```
================== ( ͡° ͜ʖ ͡°) MENU SYSTEM ( ͡° ͜ʖ ͡°) =====================
 |                                                                       |
 |_______________________________________________________________________|
 |                1: Enter your student's id and grade                   |
 |______________________________ʕ•ᴥ•ʔ____________________________________|
 |                2: View all student information                        |
 |____________________________(ᵔ•ᴥ•ᵔ)ϛ__________________________________|
 |                3: View highest student grade                          |
 |_____________________________ಠ ಠ______________________________________|
 |                4: View lowest student grade                           |
 |__________________________ᕕʕ •ₒ• ʔ9___________________________________|
 |                5: Exit the program                                    |
 |____________________________ʕ ᵔ• ₒ •ᵔ ʔ_______________________________|

Enter your choice here: █
```

Figure 17: Result of menu function [20]

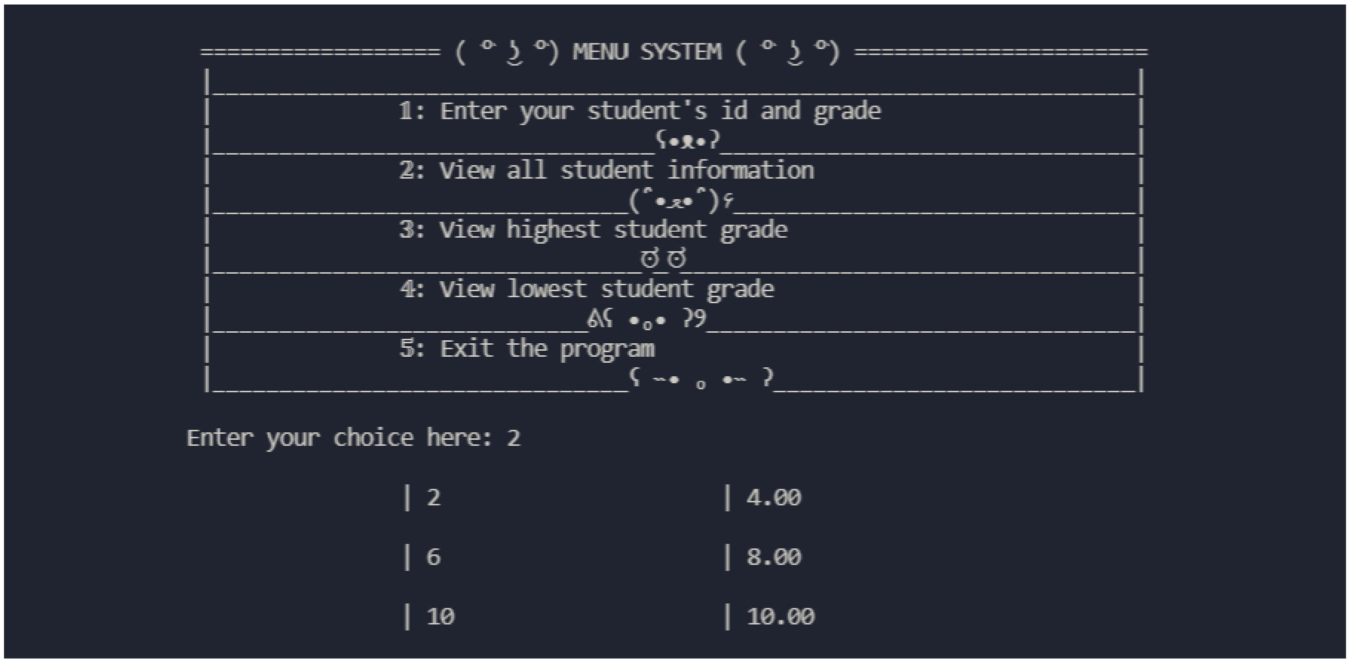
When the user selects option 1, a message prompting the user to input a few students appears on the screen. Display user-selected options and statements for entering student information.



```
 ================== ( ͡° ͜ʖ ͡°) MENU SYSTEM ( ͡° ͜ʖ ͡°) ======================
 |_____________________________________________________________________|
 |               1: Enter your student's id and grade                  |
 |_________________________________ʕ•ᴥ•ʔ_______________________________|
 |               2: View all student information                       |
 |_______________________________(ᵔ•ﻌ•ᵔ)ʖ______________________________|
 |               3: View highest student grade                         |
 |________________________________ಠ_ಠ__________________________________|
 |               4: View lowest student grade                          |
 |____________________________ᕦʕ •ₒ• ʔ９________________________________|
 |               5: Exit the program                                   |
 |_______________________________ʕ ~• ₒ •~ ʔ___________________________|

Enter your choice here: 1

 『 』Select the quantity to enter: 3

✻Enter the ID of students: 2
✻Enter grade of the students: 4

✻Enter the ID of students: 6
✻Enter grade of the students: 8

✻Enter the ID of students: 10
✻Enter grade of the students: 10
```

Figure 18: Result of option 1 function [21]

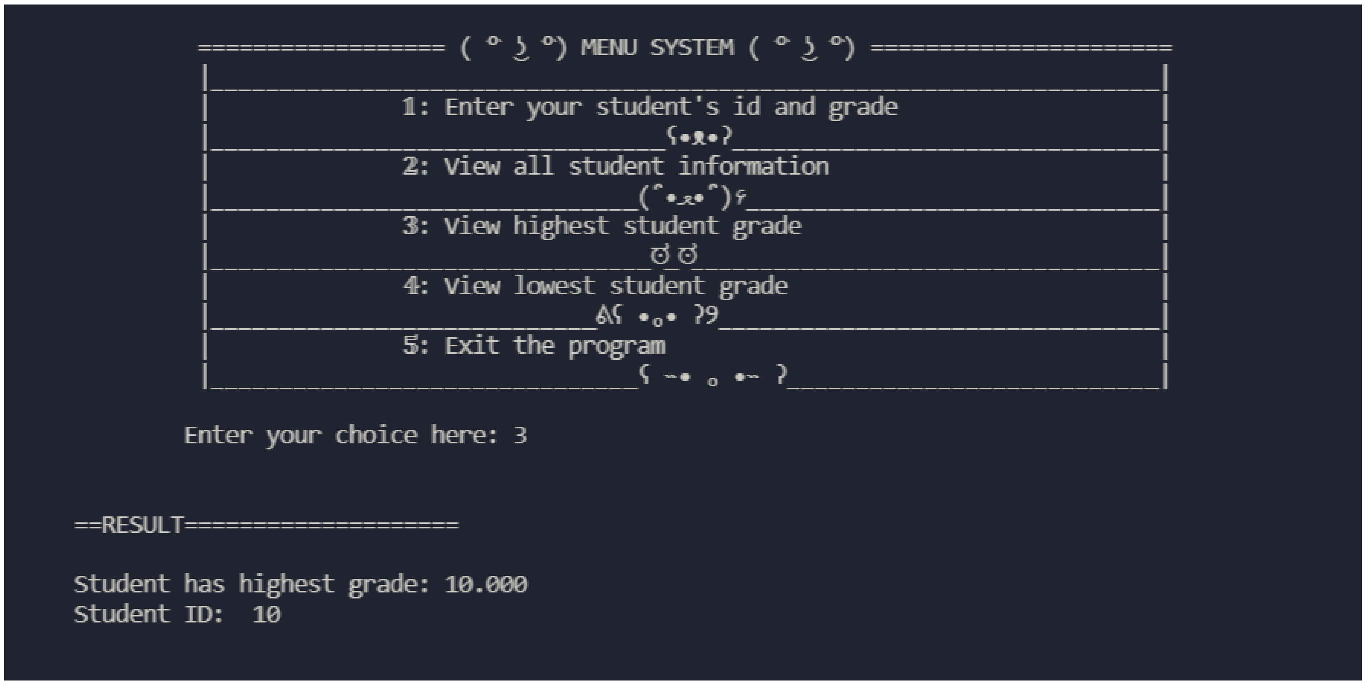
When the user chooses option 2, the program will display on the screen all the student's information including the student's grade and ID.



```
                  ================= ( ͡° ͜ʖ ͡°) MENU SYSTEM ( ͡° ͜ʖ ͡°) =====================
                  |__________________________________________________________________|
                  |              1: Enter your student's id and grade                |
                  |________________________________ʕ•ᴥ•ʔ_____________________________|
                  |              2: View all student information                    |
                  |______________________________(ˆ•ﻌ•ˆ)ﻭ___________________________|
                  |              3: View highest student grade                       |
                  |________________________________ಠ ಠ_______________________________|
                  |              4: View lowest student grade                        |
                  |_____________________________6ʕ •ₒ• ʔ9____________________________|
                  |              5: Exit the program                                 |
                  |______________________________ʕ ᵔ• ₒ •ᵔ ʔ_________________________|

        Enter your choice here: 2

                        | 2                    | 4.00

                        | 6                    | 8.00

                        | 10                   | 10.00
```

Figure 19: Result of option 2 function [22]

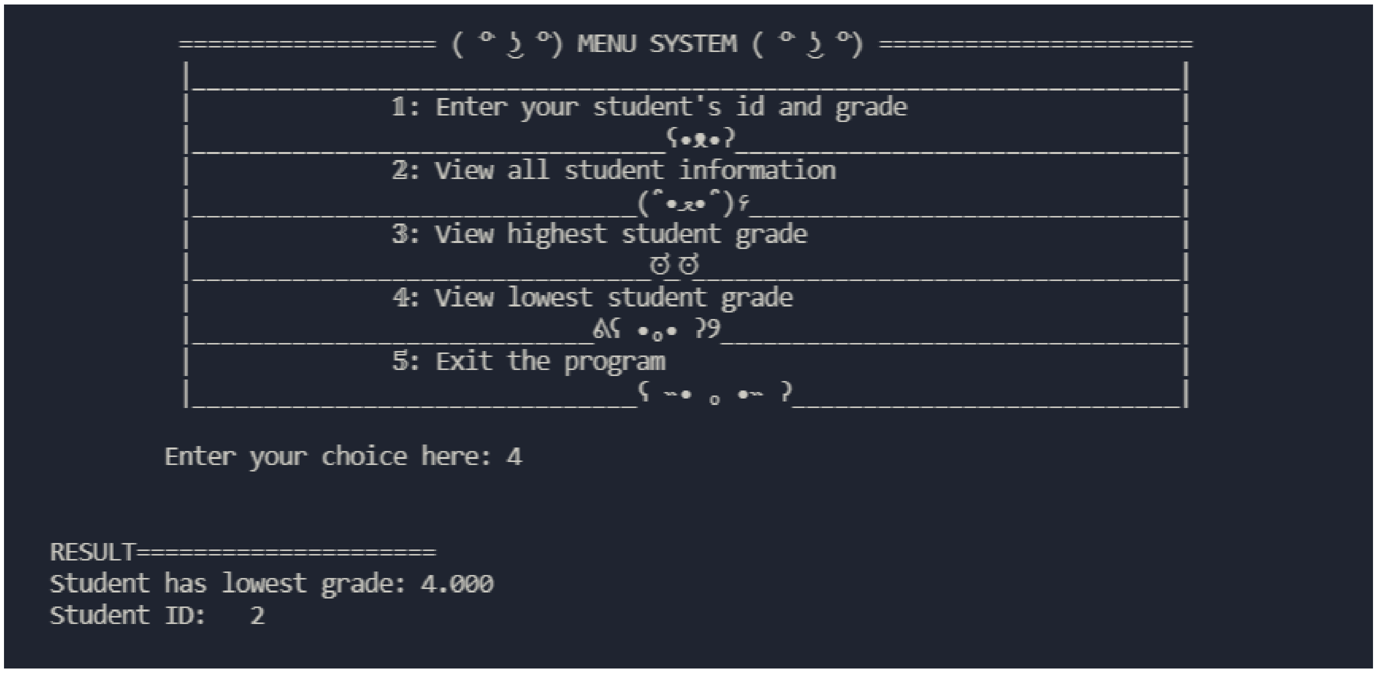
When the user does option 3, the program will display on the screen the student with the highest grade.



```
                  ================= ( ͡° ͜ʖ ͡°) MENU SYSTEM ( ͡° ͜ʖ ͡°) =====================
                  |__________________________________________________________________|
                  |              1: Enter your student's id and grade                |
                  |________________________________ʕ•ᴥ•ʔ_____________________________|
                  |              2: View all student information                    |
                  |______________________________(ˆ•ﻌ•ˆ)ﻭ___________________________|
                  |              3: View highest student grade                       |
                  |________________________________ಠ ಠ_______________________________|
                  |              4: View lowest student grade                        |
                  |_____________________________6ʕ •ₒ• ʔ9____________________________|
                  |              5: Exit the program                                 |
                  |______________________________ʕ ᵔ• ₒ •ᵔ ʔ_________________________|

        Enter your choice here: 3


    ==RESULT===================

    Student has highest grade: 10.000
    Student ID:  10
```

Figure 20: Result of option 3 [23]

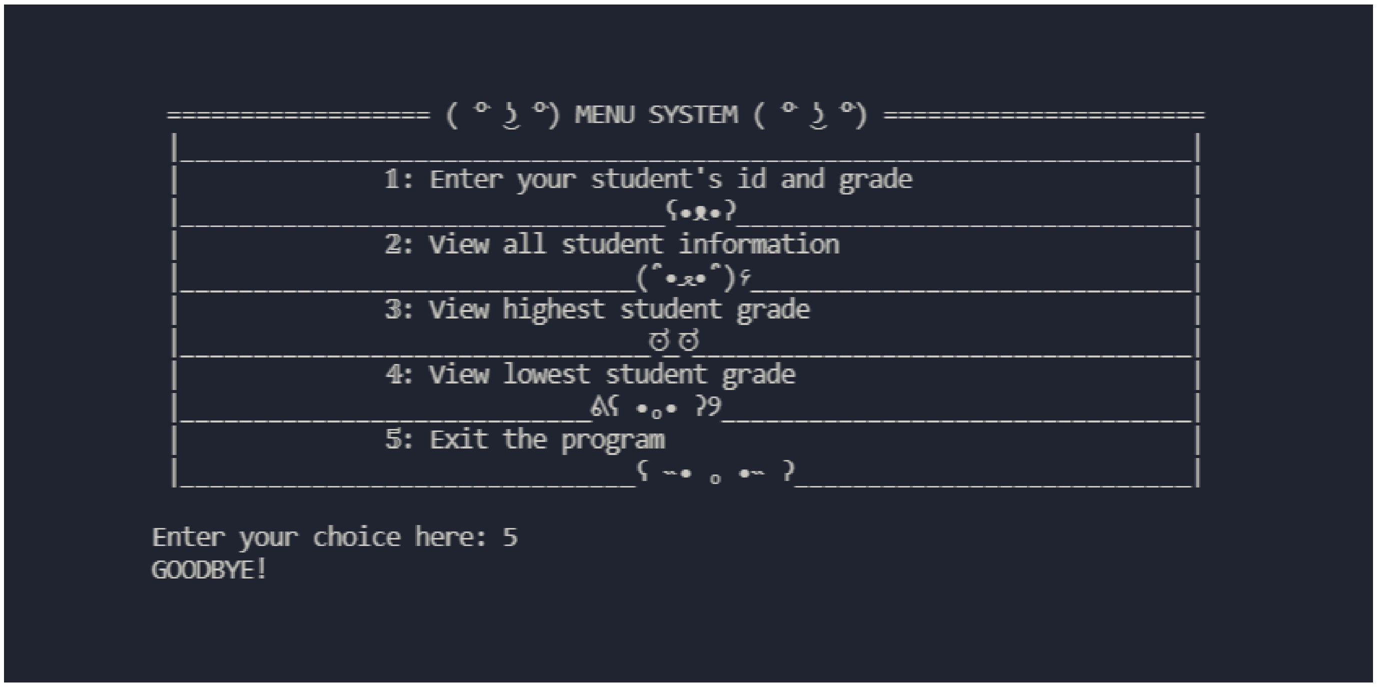
User chooses option 4, the program will return the lowest grade to the user



```
        ================= ( ͡° ͜ʖ ͡°) MENU SYSTEM ( ͡° ͜ʖ ͡°) =====================
        |______________________________________________________________________|
        |              1: Enter your student's id and grade                    |
        |_________________________________ʕ•ᴥ•ʔ________________________________|
        |              2: View all student information                         |
        |______________________________(ᵔᴥᵔ)ʃ_________________________________|
        |              3: View highest student grade                           |
        |_______________________________ಠಠ_____________________________________|
        |              4: View lowest student grade                            |
        |_____________________________ᕦʕ •ₒ• ʔ୨________________________________|
        |              5: Exit the program                                     |
        |_______________________________ʕ ˵• ₒ •˵ ʔ____________________________|

       Enter your choice here: 4


   RESULT=====================
   Student has lowest grade: 4.000
   Student ID:    2
```

Figure 21: Result of option 4 [24]

When the user selects option 5, the program will be terminated. The goodbye will be displayed on the screen.

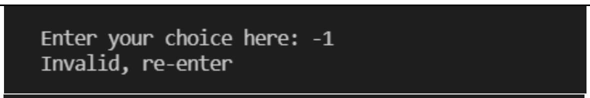
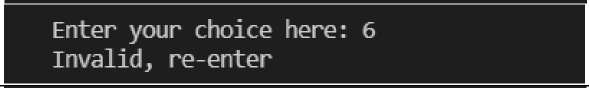
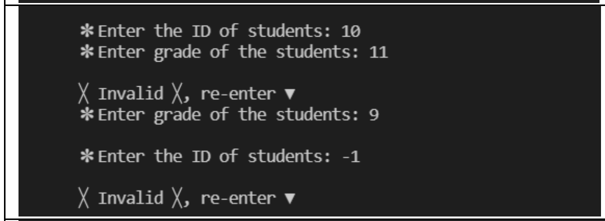
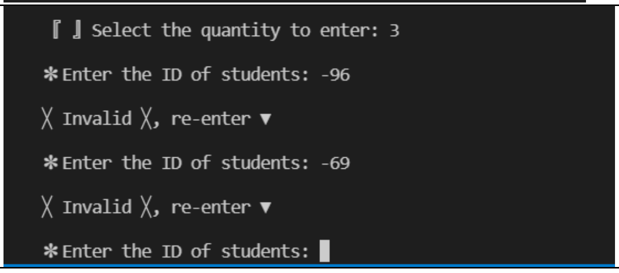
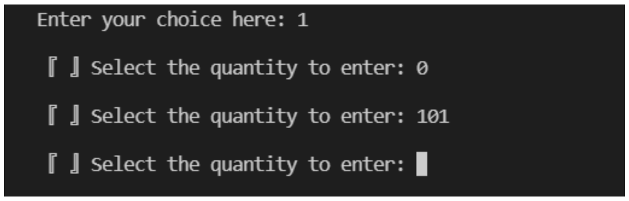
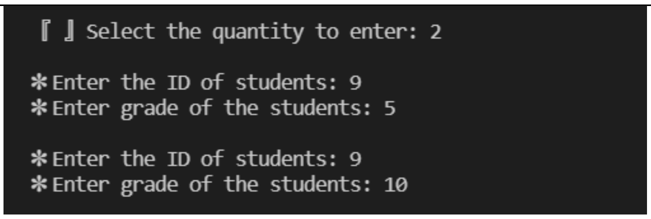
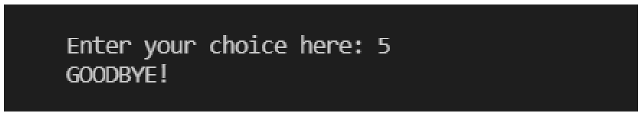


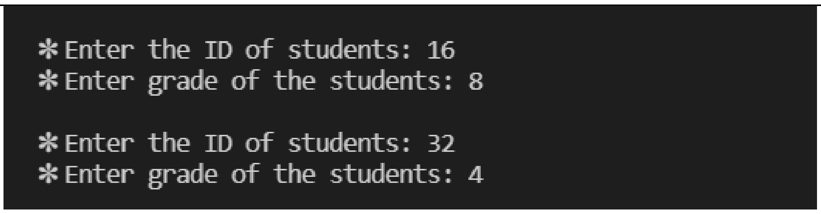
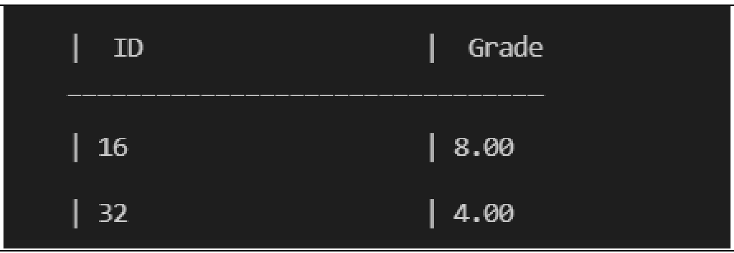
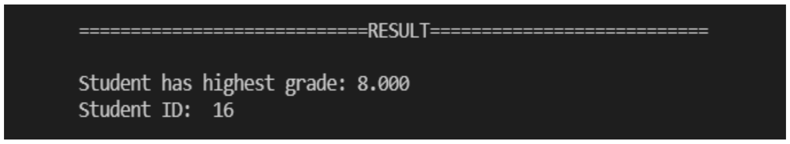
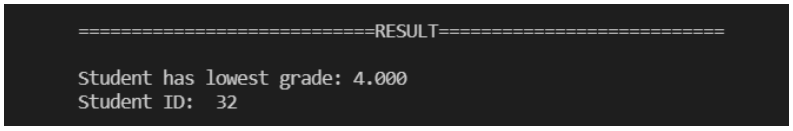
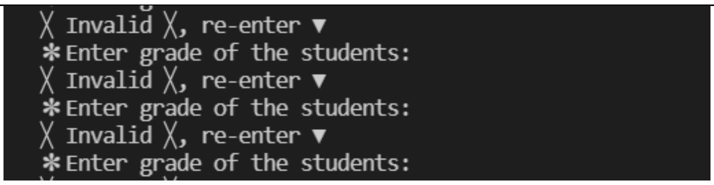
```
        ================= ( ͡° ͜ʖ ͡°) MENU SYSTEM ( ͡° ͜ʖ ͡°) =====================
        |______________________________________________________________________|
        |              1: Enter your student's id and grade                    |
        |_________________________________ʕ•ᴥ•ʔ________________________________|
        |              2: View all student information                         |
        |______________________________(ᵔᴥᵔ)ʃ_________________________________|
        |              3: View highest student grade                           |
        |_______________________________ಠಠ_____________________________________|
        |              4: View lowest student grade                            |
        |_____________________________ᕦʕ •ₒ• ʔ୨________________________________|
        |              5: Exit the program                                     |
        |_______________________________ʕ ˵• ₒ •˵ ʔ____________________________|

       Enter your choice here: 5
       GOODBYE!
```

Figure 22: Result of option 5 [25]

III.TESTING:

1.Test:

Test case	Test content	Test data	Expected	Actual result	Pass /Fail
1	Enter the wrong choice (<=0 or >5)	Enter option numbers -1 and 6		 Enter your choice here: -1 Invalid, re-enter  Enter your choice here: 6 Invalid, re-enter	Pass
2	Enter the wrong grade (<=0 or > 10)	Enter score <0 or >10	Returns "Invalid, re-enter"	 ✻Enter the ID of students: 10 ✻Enter grade of the students: 11 ╳ Invalid ╳, re-enter ▼ ✻Enter grade of the students: 9 ✻Enter the ID of students: -1 ╳ Invalid ╳, re-enter ▼	Pass
3	Enter the wrong range of valid ID values	Enter ID values: -96 and -69		 〖 〗 Select the quantity to enter: 3 ✻Enter the ID of students: -96 ╳ Invalid ╳, re-enter ▼ ✻Enter the ID of students: -69 ╳ Invalid ╳, re-enter ▼ ✻Enter the ID of students: █	Pass
4	Enter the wrong range of conditions for the number of students to enter in option 1	Enter quantity <=0 or >100	The program returns to the input of the number of students	 Enter your choice here: 1 〖 〗 Select the quantity to enter: 0 〖 〗 Select the quantity to enter: 101 〖 〗 Select the quantity to enter: █	Pass
5	Enter a duplicate ID that already exists	Enter the ID that matches the previous ID with a value of 9	The program must report an error	 〖 〗 Select the quantity to enter: 2 ✻Enter the ID of students: 9 ✻Enter grade of the students: 5 ✻Enter the ID of students: 9 ✻Enter grade of the students: 10	Fail
6	Test exit	Enter option 5	The program prints goodbye and exits	 Enter your choice here: 5 GOODBYE!	Pass
7	Test ID	Enter student's	Input is valid and print the		Pass

		ID: 16 and 32	correct result		
8	Test grade	Enter student's grade: 8 and 4	Input is valid and print the correct result	 ✻Enter the ID of students: 16 ✻Enter grade of the students: 8 ✻Enter the ID of students: 32 ✻Enter grade of the students: 4	Pass
9	Test view all Student Info	Enter Student Information in testcase 8 and then choose option 2	Full display of all entered ID: 16, Grade: 8 ID: 32, Grade: 4	 | ID | Grade -- | 16 | 8.00 | 32 | 4.00	Pass
10	Test view ID and Grade of student has highest grade	Check if the highest person score is correct from the previous entry	Check that the results appear correctly on the screen.	 ===========================RESULT=========================== Student has highest grade: 8.000 Student ID: 16	Pass
11	Test view ID and Grade of student has lowest grade	Check if the highest person score is correct from the previous entry	Check that the results appear correctly on the screen.	 ===========================RESULT=========================== Student has lowest grade: 4.000 Student ID: 32	Pass
12	Enter letters in option 1	Try entering any character like: "dump".	The program must stop and report an error to the user	 ╳ Invalid ╳, re-enter ▼ ✻Enter grade of the students: ╳ Invalid ╳, re-enter ▼ ✻Enter grade of the students: ╳ Invalid ╳, re-enter ▼ ✻Enter grade of the students:	Fail

2.Analysis result:

During the design and testing process, we discovered a few problems that will need to be corrected in the future. specifically, when the user enters IDs, they must match the field's ID standard as the first 3 digits are GCD. When the other options have not yet received the grade entry data, there must be a message and display the student's consent option whether to return to option 1 to enter the grade or not.

The application has been unit tested by me for syntax issues and simple logic errors. Basically, the program has very few errors, but their point of existence is that there are still many more carefully handled cases such as:

- Error when entering the same ID that already exists
- Error about hanging console tab when entering letters in score entry
- Error about hanging console tab when entering letters in menu option input

Regarding the basic needs of users, the program still meets all the criteria such as:

- Users can manipulate all tasks through a simple application.
- Beautiful interface, easy to see, easy to use and cute
- There is an option to exit the program when done.
- The program displays the full list of students and their ratings.

III.EVALUATION:

1.Advantage:

- In the program, a sort of function algorithm (sort by ID or sort by class) is used: “Insertion Sort”, a comparison algorithm that checks and ensures that all input data is correct.
- The names of variables and functions are named for their purpose.
- The structure of the program is designed based on procedural programming.
- The program displays search results in a statistical table for easy handling.
- The UI/UX feature of the program is on top.

2. Disadvantage:

- The disadvantage of not being able to reuse code across numerous applications. Rewriting the same code several times throughout a program might raise the project's cost and completion time.
- Because the program uses arrays, the number of items it can hold is limited; if the instructor wants to increase the number of elements, he must return to the code and make changes.

3. Expected improvement:

- Create some extra functionality, for example a general statistics table that includes the average score of a subject or subjects with high or low grade.

- Use a form-filling table so users can enter more things without having to meet all the options' conditions.
- It is not necessary to set the default limit on the number of students.
- The program needs to significantly update the UI for users.

In summary, using procedural programming to solve the problem is a possible alternative. The software uses predefined functions, parameter passing, procedures and programming libraries. However, using OOP, JavaScript and MySQL makes the program can be executed better and easier, the interface is more eye-catching.