OLLSCOIL TEICNEOLAÍOCHTA BHAILE ÁTHA CLIATH

DUBLIN

TECHNOLOGICAL UNIVERSITY DUBLIN

Declaration Of Own Work

Student Name: Robert Kacso
Student Number: B00123508
Course Title: Software Development 2
Lecturer: Ann Marie Cosgrave
Title of Work: Ass1B00123508
Set Submission Date: 23/03/2020

Table of Contents

SD2Ass1B00123508

About the assignment

ITB lecturers require a simple grade analysis program for analysing class tests. The program should provide the following functionality:

1. Allow a user to enter a number of results (a value between 2 & 25)

2. Process the required number of results by:

- Allowing a user to enter a student's name
- Allowing a user to enter a student's grade (a value between 1 & 100)

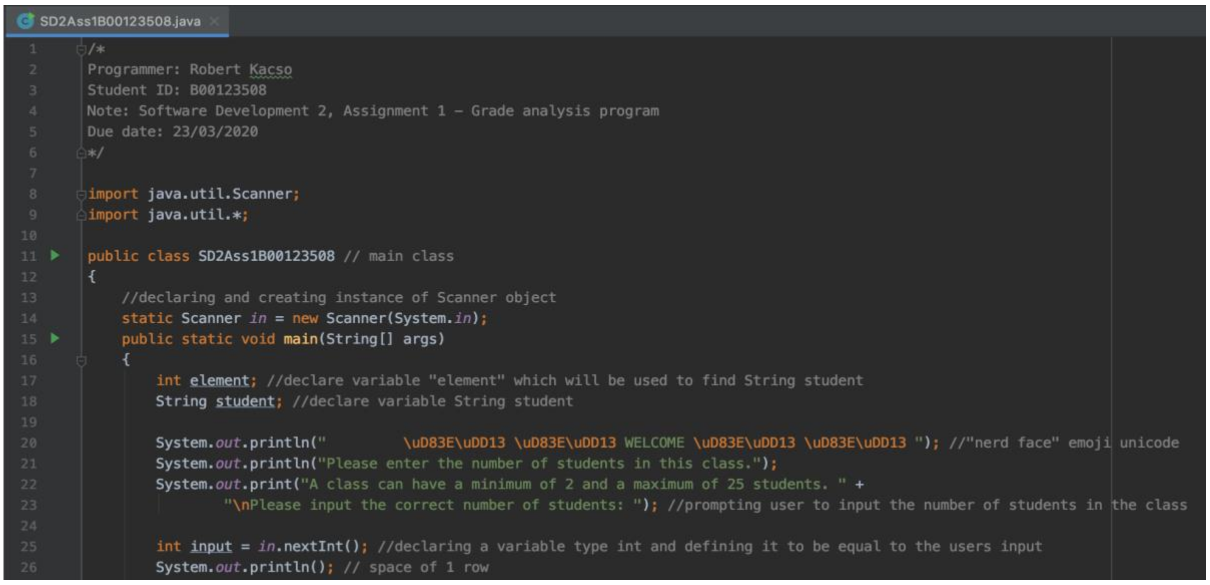
3. When data entry is complete, display a menu with the following options:

- Display average class grade
- Display lowest class grade
- Display highest class grade
- Sort & Display the grades in ascending order
- Search for an individual student by name

4. For each of the above functions, you are required to implement an appropriate method

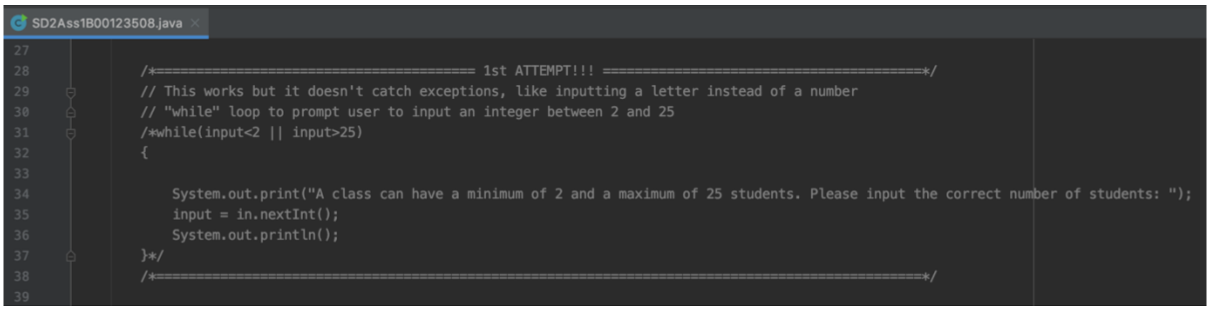
The Code

Imports, main class, scanner and the beginning of the main method in which we declare variables and prompt user for inputs.



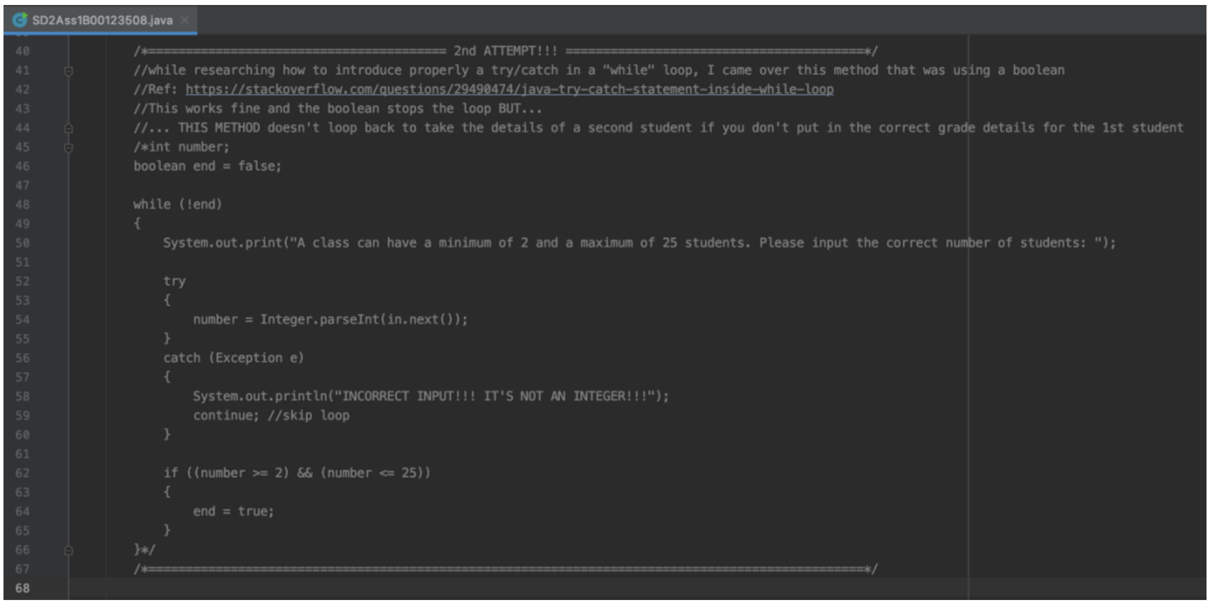
```java
/*
Programmer: Robert Kacso
Student ID: B00123508
Note: Software Development 2, Assignment 1 - Grade analysis program
Due date: 23/03/2020
*/

import java.util.Scanner;
import java.util.*;

public class SD2Ass1B00123508 // main class
{
    //declaring and creating instance of Scanner object
    static Scanner in = new Scanner(System.in);
    public static void main(String[] args)
    {
        int element; //declare variable "element" which will be used to find String student
        String student; //declare variable String student

        System.out.println("         \uD83E\uDD13 \uD83E\uDD13 WELCOME \uD83E\uDD13 \uD83E\uDD13 "); //"nerd face" emoji unicode
        System.out.println("Please enter the number of students in this class.");
        System.out.print("A class can have a minimum of 2 and a maximum of 25 students. " +
                "\nPlease input the correct number of students: "); //prompting user to input the number of students in the class

        int input = in.nextInt(); //declaring a variable type int and defining it to be equal to the users input
        System.out.println(); // space of 1 row
```

1st attempt at inputting the number of students without try/catch



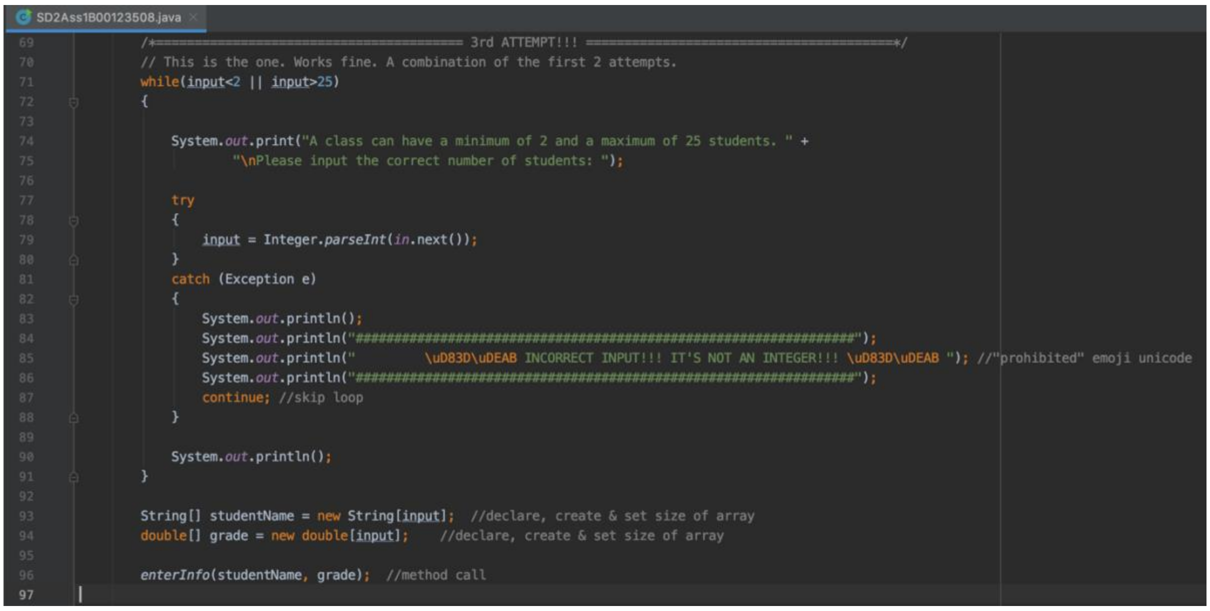
```java

        /*======================================== 1st ATTEMPT!!! ========================================*/
        // This works but it doesn't catch exceptions, like inputting a letter instead of a number
        // "while" loop to prompt user to input an integer between 2 and 25
        /*while(input<2 || input>25)
        {

            System.out.print("A class can have a minimum of 2 and a maximum of 25 students. Please input the correct number of students: ");
            input = in.nextInt();
            System.out.println();
        }*/
        /*===============================================================================================*/

```

2nd attempt at inputting the number of students with try/catch and Boolean (not successful)



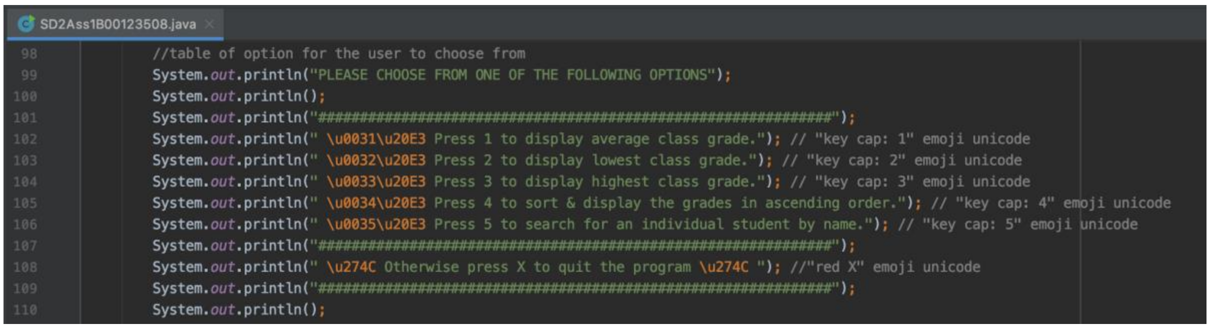
```java
        /*======================================== 2nd ATTEMPT!!! ========================================*/
        //while researching how to introduce properly a try/catch in a "while" loop, I came over this method that was using a boolean
        //Ref: https://stackoverflow.com/questions/29490474/java-try-catch-statement-inside-while-loop
        //This works fine and the boolean stops the loop BUT...
        //... THIS METHOD doesn't loop back to take the details of a second student if you don't put in the correct grade details for the 1st student
        /*int number;
        boolean end = false;

        while (!end)
        {
            System.out.print("A class can have a minimum of 2 and a maximum of 25 students. Please input the correct number of students: ");

            try
            {
                number = Integer.parseInt(in.next());
            }
            catch (Exception e)
            {
                System.out.println("INCORRECT INPUT!!! IT'S NOT AN INTEGER!!!");
                continue; //skip loop
            }

            if ((number >= 2) && (number <= 25))
            {
                end = true;
            }
        }*/
        /*===============================================================================================*/

```

3rd attempt and the successful one plus declaring the two arrays and calling in our first method


SD2Ass1B00123508.java

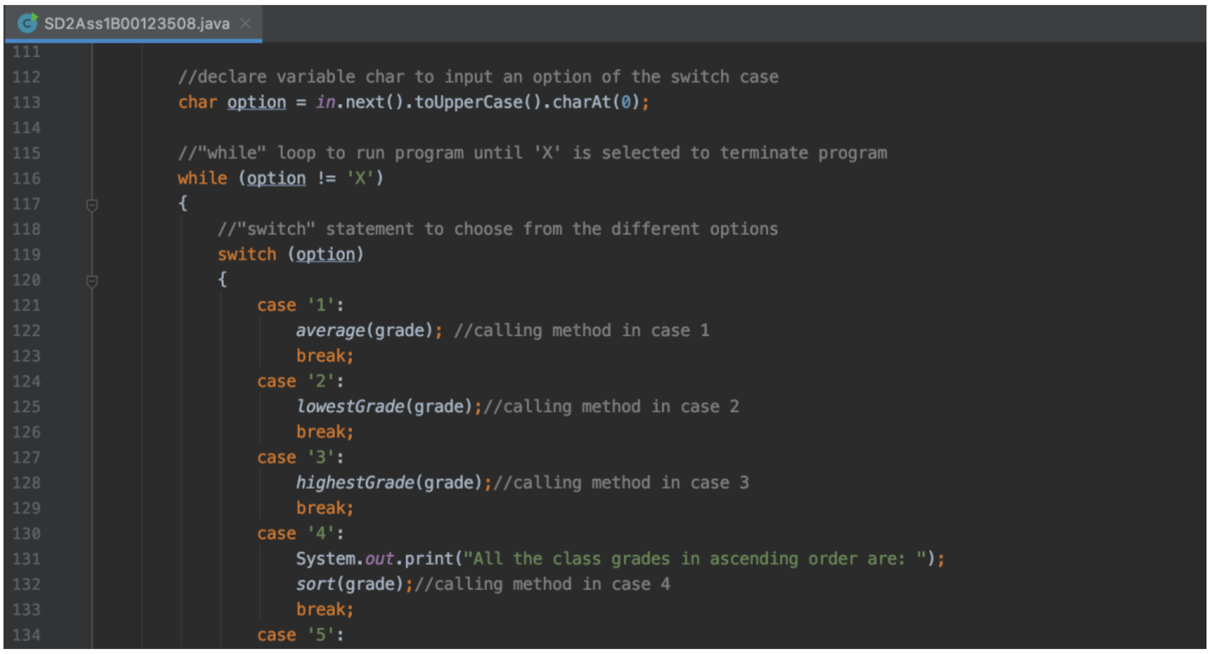
```java
        /*========================================= 3rd ATTEMPT!!! ========================================*/
        // This is the one. Works fine. A combination of the first 2 attempts.
        while(input<2 || input>25)
        {

            System.out.print("A class can have a minimum of 2 and a maximum of 25 students. " +
                    "\nPlease input the correct number of students: ");

            try
            {
                input = Integer.parseInt(in.next());
            }
            catch (Exception e)
            {
                System.out.println();
                System.out.println("#################################################################");
                System.out.println("         \uD83D\uDEAB INCORRECT INPUT!!! IT'S NOT AN INTEGER!!! \uD83D\uDEAB "); //"prohibited" emoji unicode
                System.out.println("#################################################################");
                continue; //skip loop
            }

            System.out.println();
        }

        String[] studentName = new String[input];  //declare, create & set size of array
        double[] grade = new double[input];    //declare, create & set size of array

        enterInfo(studentName, grade);  //method call

```

The option menu


SD2Ass1B00123508.java

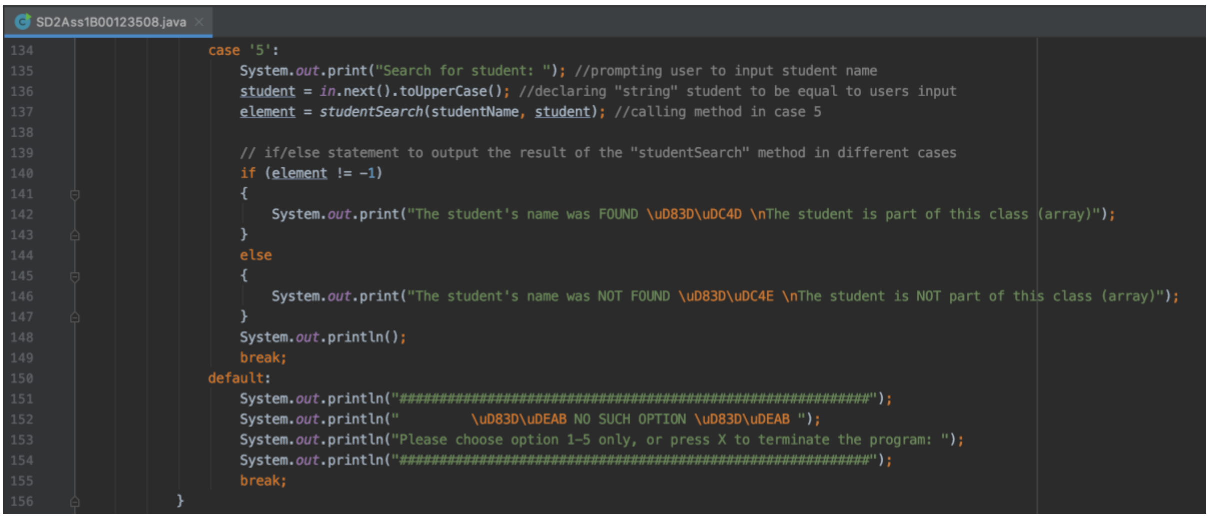
```java
        //table of option for the user to choose from
        System.out.println("PLEASE CHOOSE FROM ONE OF THE FOLLOWING OPTIONS");
        System.out.println();
        System.out.println("#############################################################");
        System.out.println(" \u0031\u20E3 Press 1 to display average class grade."); // "key cap: 1" emoji unicode
        System.out.println(" \u0032\u20E3 Press 2 to display lowest class grade."); // "key cap: 2" emoji unicode
        System.out.println(" \u0033\u20E3 Press 3 to display highest class grade."); // "key cap: 3" emoji unicode
        System.out.println(" \u0034\u20E3 Press 4 to sort & display the grades in ascending order."); // "key cap: 4" emoji unicode
        System.out.println(" \u0035\u20E3 Press 5 to search for an individual student by name."); // "key cap: 5" emoji unicode
        System.out.println("#############################################################");
        System.out.println(" \u274C Otherwise press X to quit the program \u274C "); //"red X" emoji unicode
        System.out.println("#############################################################");
        System.out.println();
```

Switch statement with the 6 cases within a “while” loop


SD2Ass1B00123508.java

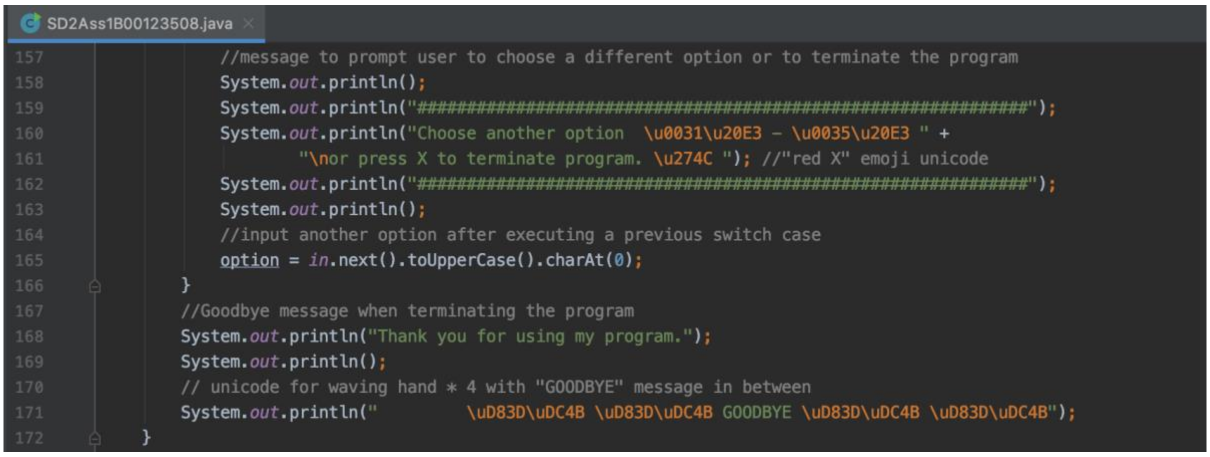
```java

        //declare variable char to input an option of the switch case
        char option = in.next().toUpperCase().charAt(0);

        //"while" loop to run program until 'X' is selected to terminate program
        while (option != 'X')
        {
            //"switch" statement to choose from the different options
            switch (option)
            {
                case '1':
                    average(grade); //calling method in case 1
                    break;
                case '2':
                    lowestGrade(grade);//calling method in case 2
                    break;
                case '3':
                    highestGrade(grade);//calling method in case 3
                    break;
                case '4':
                    System.out.print("All the class grades in ascending order are: ");
                    sort(grade);//calling method in case 4
                    break;
                case '5':
```

Switch continued



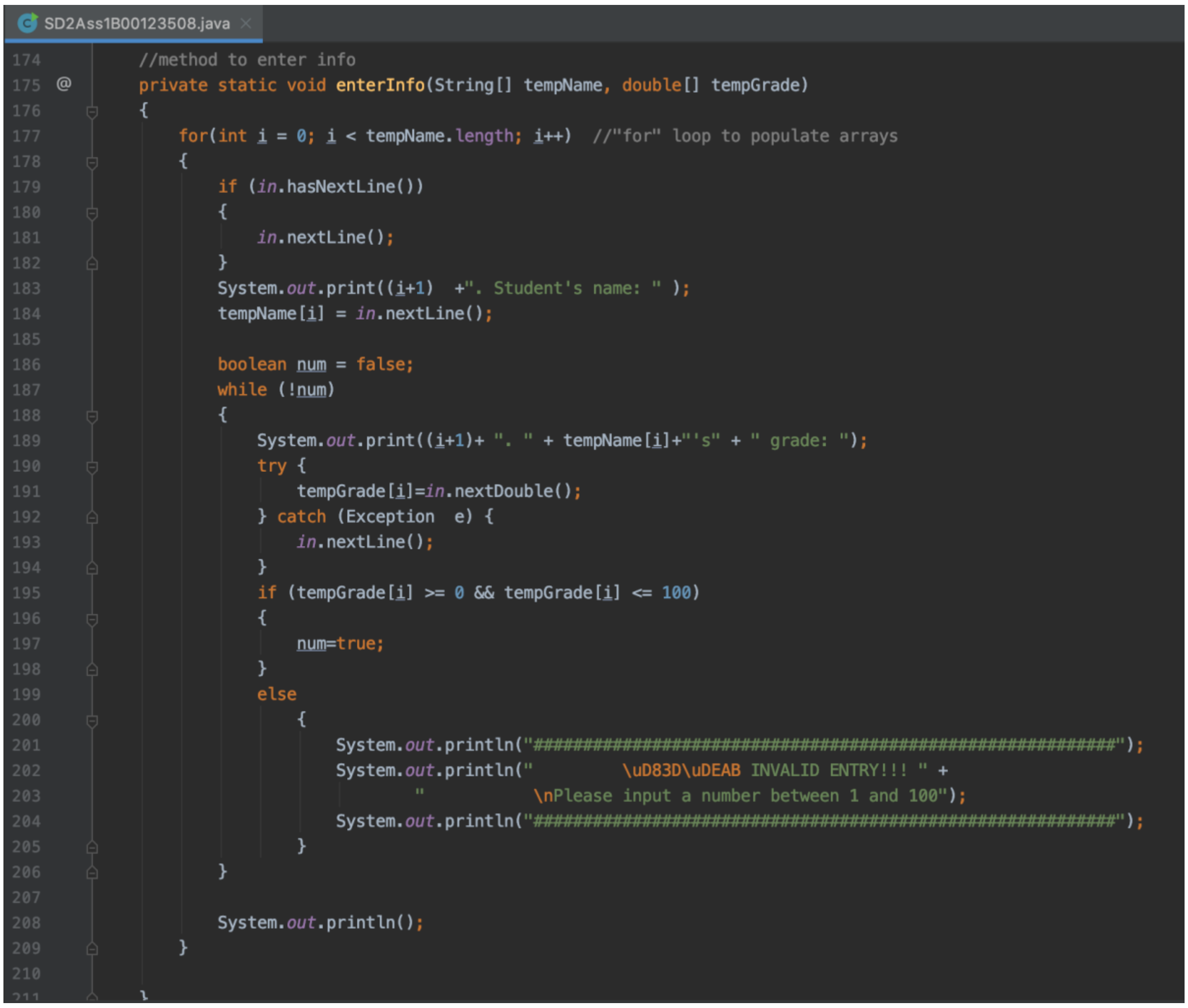
```java
SD2Ass1B00123508.java
            case '5':
                System.out.print("Search for student: "); //prompting user to input student name
                student = in.next().toUpperCase(); //declaring "string" student to be equal to users input
                element = studentSearch(studentName, student); //calling method in case 5

                // if/else statement to output the result of the "studentSearch" method in different cases
                if (element != -1)
                {
                    System.out.print("The student's name was FOUND \uD83D\uDC4D \nThe student is part of this class (array)");
                }
                else
                {
                    System.out.print("The student's name was NOT FOUND \uD83D\uDC4E \nThe student is NOT part of this class (array)");
                }
                System.out.println();
                break;
            default:
                System.out.println("############################################################");
                System.out.println("         \uD83D\uDEAB NO SUCH OPTION \uD83D\uDEAB ");
                System.out.println("Please choose option 1-5 only, or press X to terminate the program: ");
                System.out.println("############################################################");
                break;
        }
```

Switched continued and exit of the program at the end of the “while” loop



```java
SD2Ass1B00123508.java
            //message to prompt user to choose a different option or to terminate the program
            System.out.println();
            System.out.println("############################################################");
            System.out.println("Choose another option  \u0031\u20E3 - \u0035\u20E3 " +
                    "\nor press X to terminate program. \u274C "); //"red X" emoji unicode
            System.out.println("############################################################");
            System.out.println();
            //input another option after executing a previous switch case
            option = in.next().toUpperCase().charAt(0);
        }
        //Goodbye message when terminating the program
        System.out.println("Thank you for using my program.");
        System.out.println();
        // unicode for waving hand * 4 with "GOODBYE" message in between
        System.out.println("         \uD83D\uDC4B \uD83D\uDC4B GOODBYE \uD83D\uDC4B \uD83D\uDC4B");
    }
```

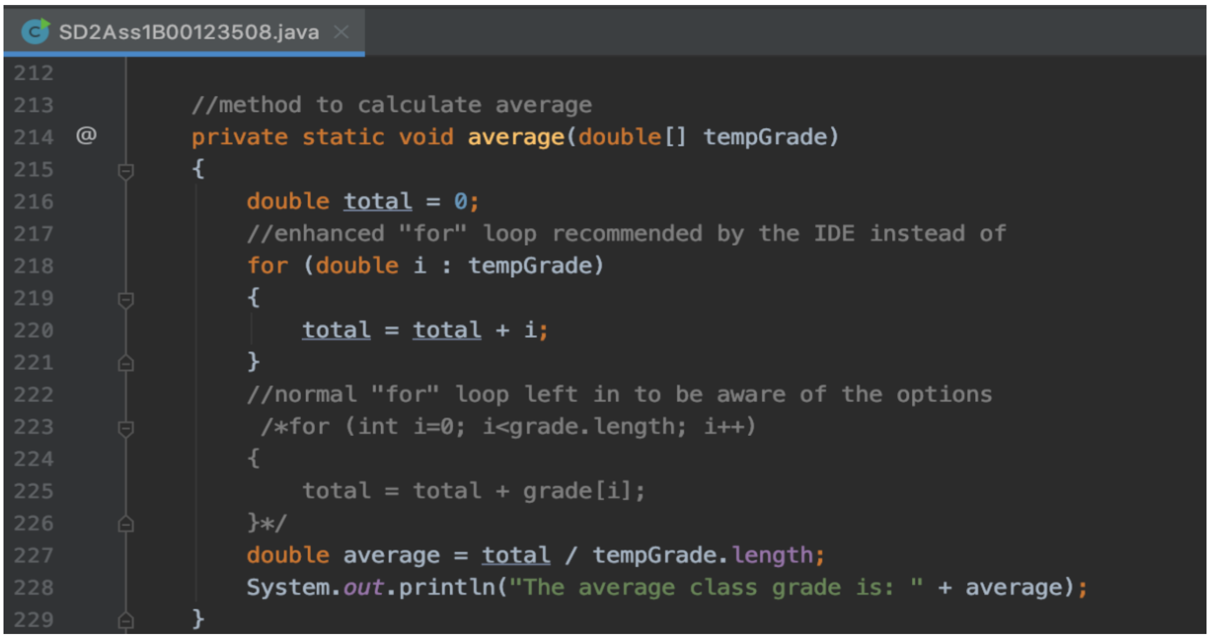
enterInfo() method



SD2Ass1B00123508.java

```java
    //method to enter info
    private static void enterInfo(String[] tempName, double[] tempGrade)
    {
        for(int i = 0; i < tempName.length; i++)  //"for" loop to populate arrays
        {
            if (in.hasNextLine())
            {
                in.nextLine();
            }
            System.out.print((i+1)  +". Student's name: " );
            tempName[i] = in.nextLine();

            boolean num = false;
            while (!num)
            {
                System.out.print((i+1)+ ". " + tempName[i]+"'s" + " grade: ");
                try {
                    tempGrade[i]=in.nextDouble();
                } catch (Exception  e) {
                    in.nextLine();
                }
                if (tempGrade[i] >= 0 && tempGrade[i] <= 100)
                {
                    num=true;
                }
                else
                    {
                        System.out.println("###########################################################");
                        System.out.println("         \uD83D\uDEAB INVALID ENTRY!!! " +
                                "           \nPlease input a number between 1 and 100");
                        System.out.println("###########################################################");
                    }
            }

            System.out.println();
        }

    }
```

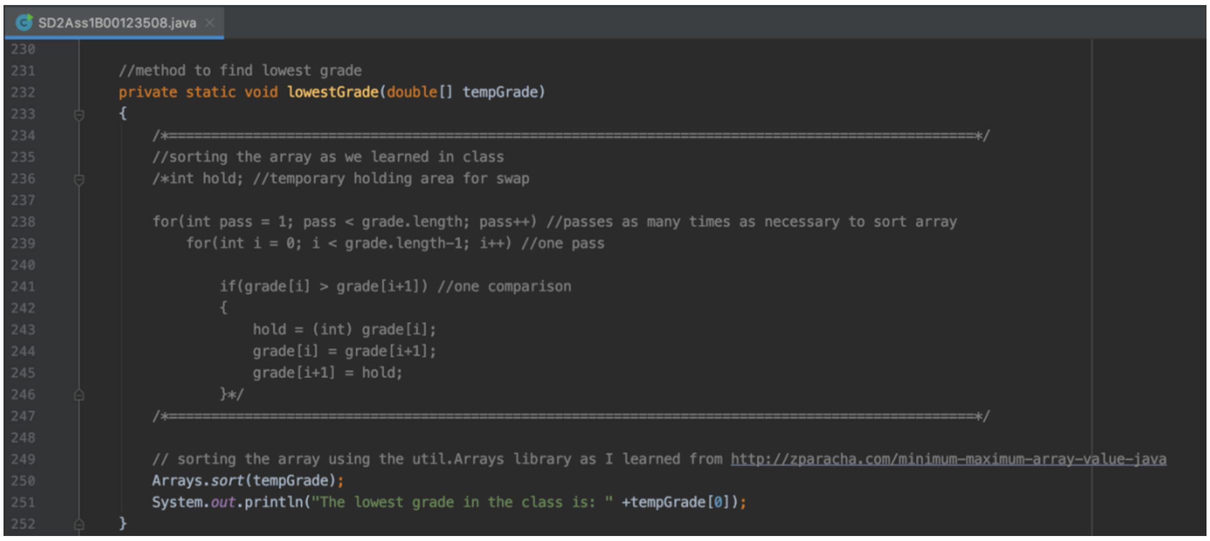
average() method



SD2Ass1B00123508.java

```java

    //method to calculate average
    private static void average(double[] tempGrade)
    {
        double total = 0;
        //enhanced "for" loop recommended by the IDE instead of
        for (double i : tempGrade)
        {
            total = total + i;
        }
        //normal "for" loop left in to be aware of the options
         /*for (int i=0; i<grade.length; i++)
        {
            total = total + grade[i];
        }*/
        double average = total / tempGrade.length;
        System.out.println("The average class grade is: " + average);
    }
```

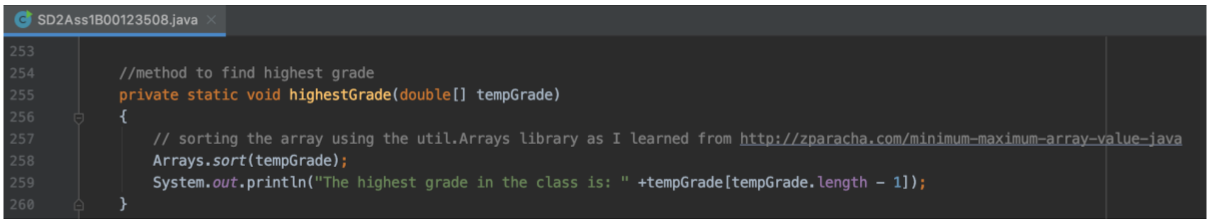
lowestGrade() method



SD2Ass1B00123508.java

```java

    //method to find lowest grade
    private static void lowestGrade(double[] tempGrade)
    {
        /*==========================================================================================*/
        //sorting the array as we learned in class
        /*int hold; //temporary holding area for swap

        for(int pass = 1; pass < grade.length; pass++) //passes as many times as necessary to sort array
            for(int i = 0; i < grade.length-1; i++) //one pass

                if(grade[i] > grade[i+1]) //one comparison
                {
                    hold = (int) grade[i];
                    grade[i] = grade[i+1];
                    grade[i+1] = hold;
                }*/
        /*==========================================================================================*/

        // sorting the array using the util.Arrays library as I learned from http://zparacha.com/minimum-maximum-array-value-java
        Arrays.sort(tempGrade);
        System.out.println("The lowest grade in the class is: " +tempGrade[0]);
    }
```

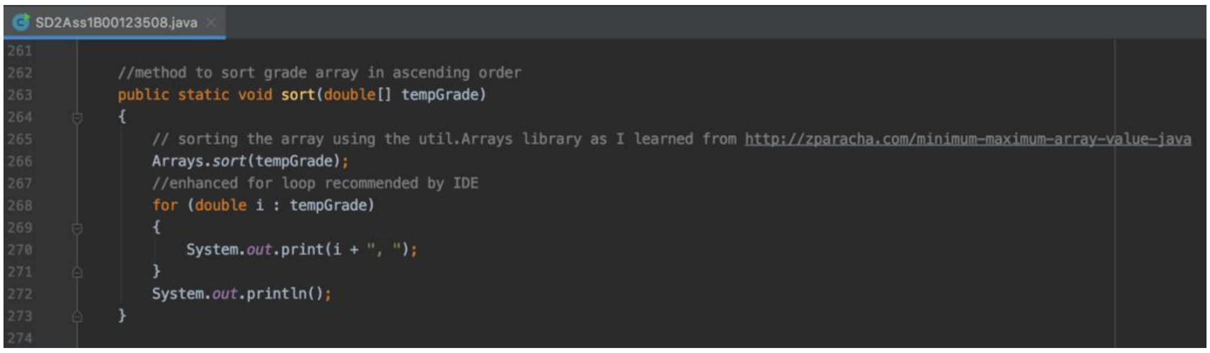
highestGrade() method



SD2Ass1B00123508.java

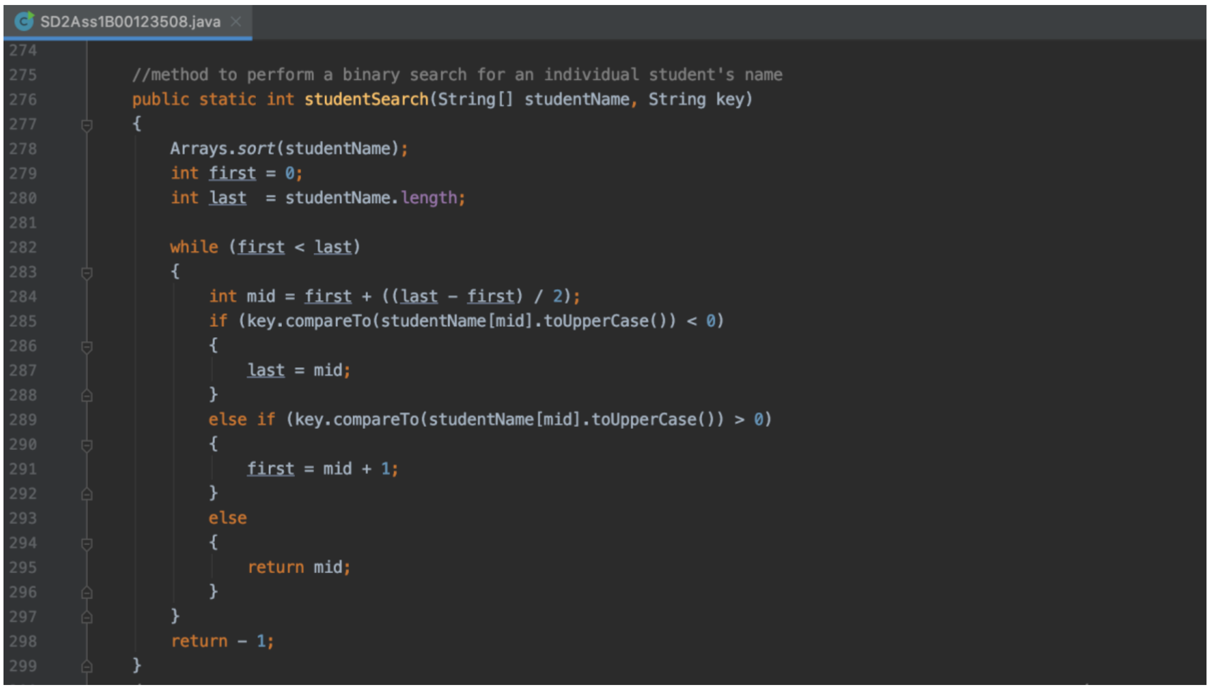
```java

    //method to find highest grade
    private static void highestGrade(double[] tempGrade)
    {
        // sorting the array using the util.Arrays library as I learned from http://zparacha.com/minimum-maximum-array-value-java
        Arrays.sort(tempGrade);
        System.out.println("The highest grade in the class is: " +tempGrade[tempGrade.length - 1]);
    }
```

sort() method


SD2Ass1B00123508.java

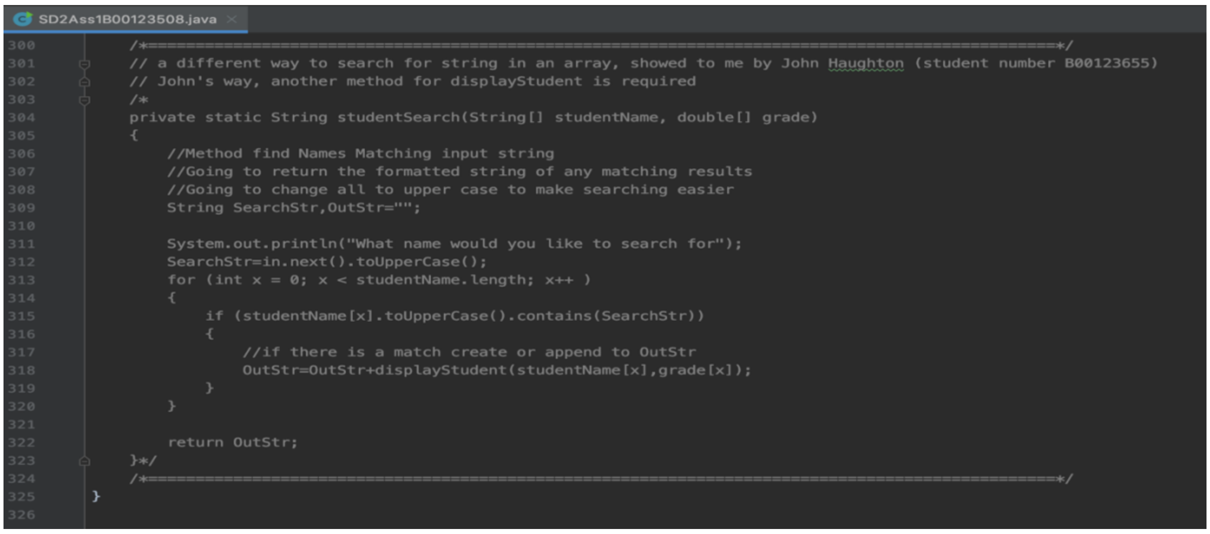
```java

    //method to sort grade array in ascending order
    public static void sort(double[] tempGrade)
    {
        // sorting the array using the util.Arrays library as I learned from http://zparacha.com/minimum-maximum-array-value-java
        Arrays.sort(tempGrade);
        //enhanced for loop recommended by IDE
        for (double i : tempGrade)
        {
            System.out.print(i + ", ");
        }
        System.out.println();
    }

```

studentSearch() method


SD2Ass1B00123508.java

```java

    //method to perform a binary search for an individual student's name
    public static int studentSearch(String[] studentName, String key)
    {
        Arrays.sort(studentName);
        int first = 0;
        int last  = studentName.length;

        while (first < last)
        {
            int mid = first + ((last - first) / 2);
            if (key.compareTo(studentName[mid].toUpperCase()) < 0)
            {
                last = mid;
            }
            else if (key.compareTo(studentName[mid].toUpperCase()) > 0)
            {
                first = mid + 1;
            }
            else
            {
                return mid;
            }
        }
        return - 1;
    }
```

A different way to search for a string array. Not used but kept it as a reference for future


SD2Ass1B00123508.java

```java
    /*==========================================================================================*/
    // a different way to search for string in an array, showed to me by John Haughton (student number B00123655)
    // John's way, another method for displayStudent is required
    /*
    private static String studentSearch(String[] studentName, double[] grade)
    {
        //Method find Names Matching input string
        //Going to return the formatted string of any matching results
        //Going to change all to upper case to make searching easier
        String SearchStr,OutStr="";

        System.out.println("What name would you like to search for");
        SearchStr=in.next().toUpperCase();
        for (int x = 0; x < studentName.length; x++ )
        {
            if (studentName[x].toUpperCase().contains(SearchStr))
            {
                //if there is a match create or append to OutStr
                OutStr=OutStr+displayStudent(studentName[x],grade[x]);
            }
        }

        return OutStr;
    }*/
    /*==========================================================================================*/
}

```

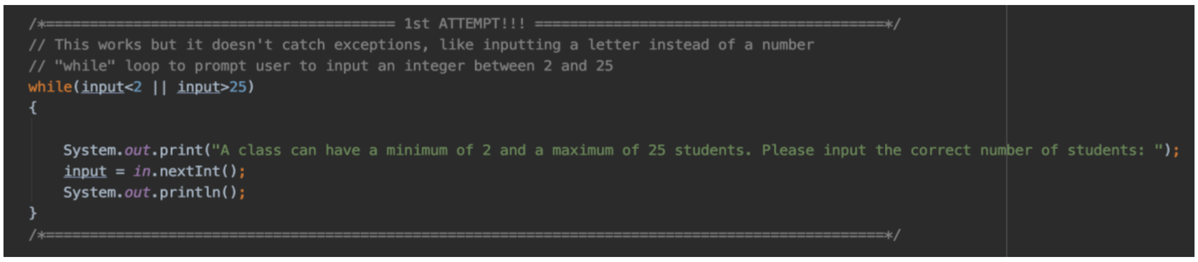
Problems encountered

In the first part of the program, where the user needs to input the number of students, I used a “while” statement to ask users to input a integer number between 2 and 25. If a number outside this range is inputted, a statement will print to the screen prompting the user again to input a number between 2 and 25 and asking to stay in that range.

The problem that I encountered was when the user inputs a letter instead of a number, so I introduced a try/catch for my second attempt.

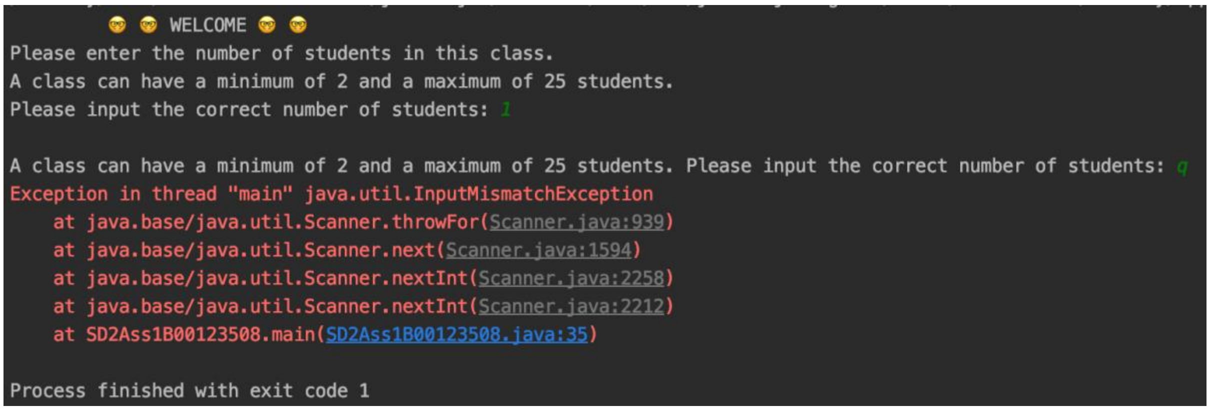
1st attempt.

Here is the code without the try/catch:



```
/*======================================== 1st ATTEMPT!!! ========================================*/
// This works but it doesn't catch exceptions, like inputting a letter instead of a number
// "while" loop to prompt user to input an integer between 2 and 25
while(input<2 || input>25)
{

    System.out.print("A class can have a minimum of 2 and a maximum of 25 students. Please input the correct number of students: ");
    input = in.nextInt();
    System.out.println();
}
/*================================================================================================*/
```

And here is the error when running the program and the user inputs a letter instead of a number:

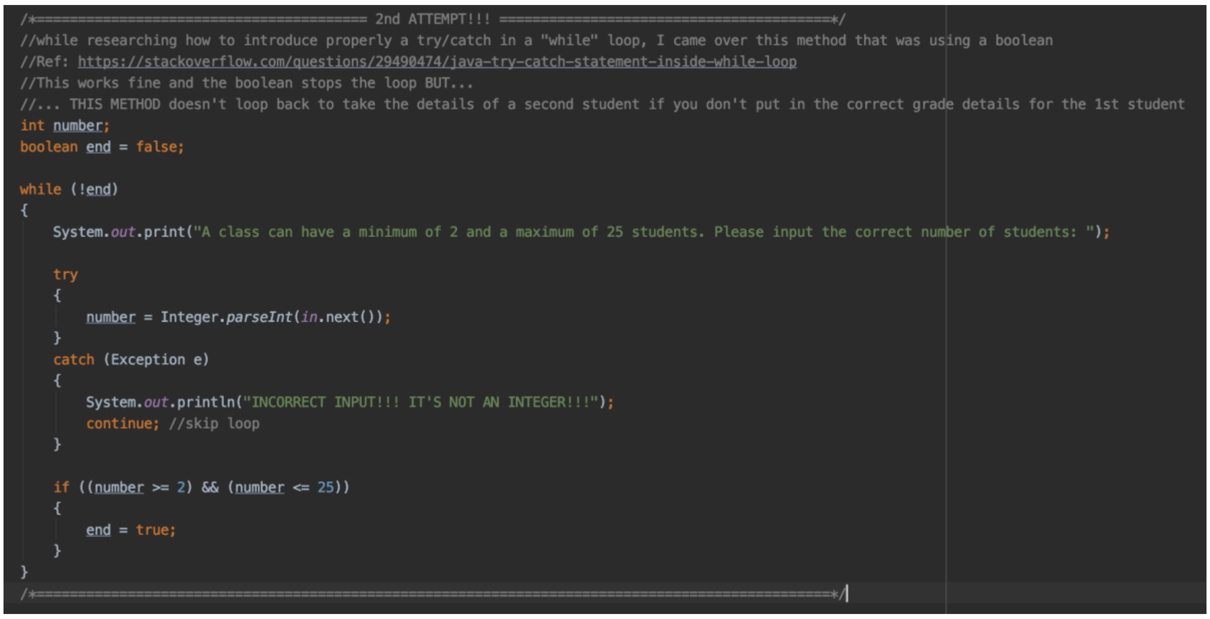


```
        🤓 🤓 WELCOME 🤓 🤓
Please enter the number of students in this class.
A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students: 1

A class can have a minimum of 2 and a maximum of 25 students. Please input the correct number of students: q
Exception in thread "main" java.util.InputMismatchException
	at java.base/java.util.Scanner.throwFor(Scanner.java:939)
	at java.base/java.util.Scanner.next(Scanner.java:1594)
	at java.base/java.util.Scanner.nextInt(Scanner.java:2258)
	at java.base/java.util.Scanner.nextInt(Scanner.java:2212)
	at SD2Ass1B00123508.main(SD2Ass1B00123508.java:35)

Process finished with exit code 1
```

As you can see, when the user inputs an integer out of range (less than 2 or more than 25), a message comes up prompting the user to input a number between 2 and 25. If a letter is pressed by mistake, an error occurs and the program terminates.

2nd attempt.

For my second attempt I introduced a try/catch (method used is referenced in the program code). Almost worked, but another problem occurred. Let’s have a look at the code first:

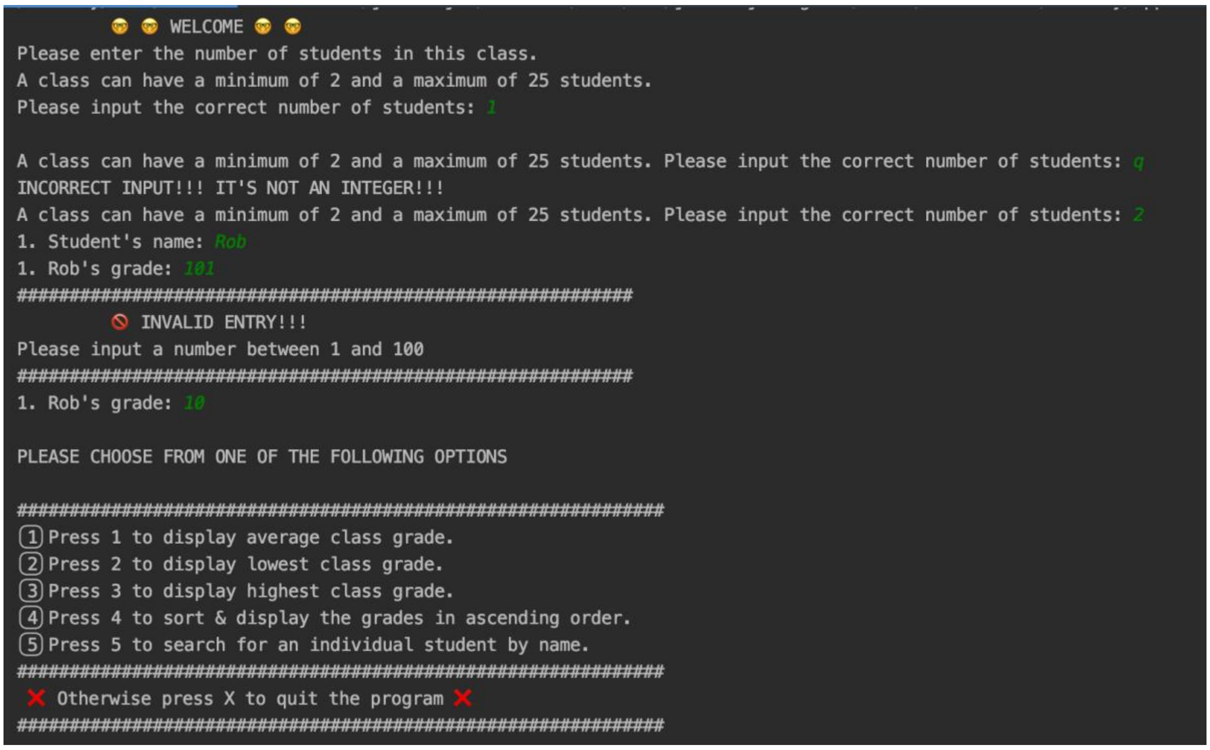


```
/*======================================= 2nd ATTEMPT!!! =======================================*/
//while researching how to introduce properly a try/catch in a "while" loop, I came over this method that was using a boolean
//Ref: https://stackoverflow.com/questions/29490474/java-try-catch-statement-inside-while-loop
//This works fine and the boolean stops the loop BUT...
//... THIS METHOD doesn't loop back to take the details of a second student if you don't put in the correct grade details for the 1st student
int number;
boolean end = false;

while (!end)
{
    System.out.print("A class can have a minimum of 2 and a maximum of 25 students. Please input the correct number of students: ");

    try
    {
        number = Integer.parseInt(in.next());
    }
    catch (Exception e)
    {
        System.out.println("INCORRECT INPUT!!! IT'S NOT AN INTEGER!!!");
        continue; //skip loop
    }

    if ((number >= 2) && (number <= 25))
    {
        end = true;
    }
}
/*===============================================================================================*/
```

Now as you can see in the screenshot bellow, everything works as it should until a wrong grade is being inputted for the 1st student.

Another thing I realised is that I didn't need to declare another variable type "int" for my input. (Corrected it in my 3rd attempt)

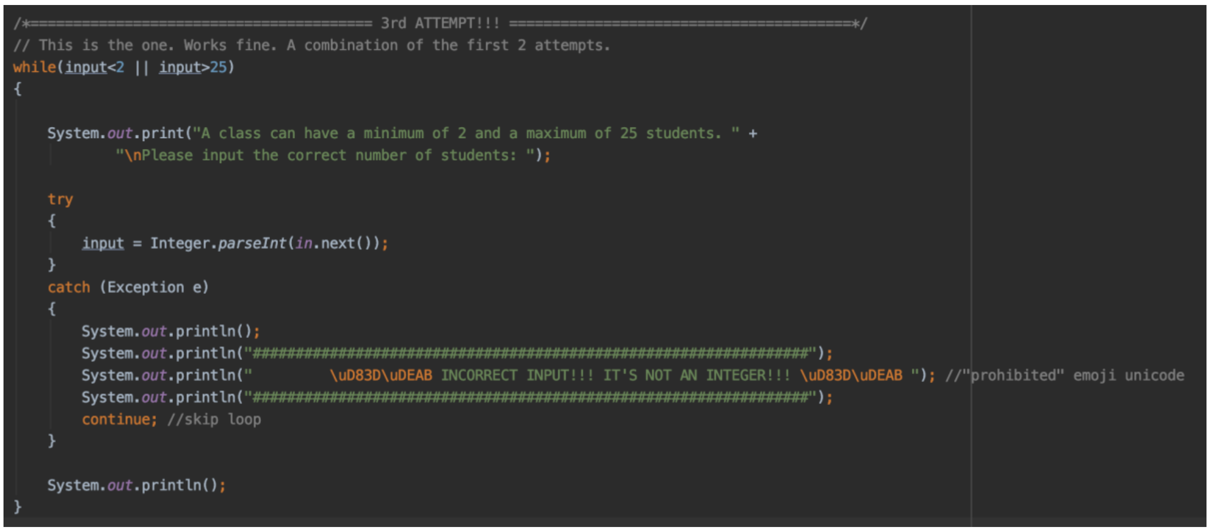
Program asks to input a correct grade, between 1 and 100, but after the user inputs the correct grade, the program should continue to ask to input the name and grade of the second student, but it won't. Instead jumps to the next phase of the program which displays an option menu to the user. I think is because of the Boolean terminating the loop once it has received the correct input.



```
        🤓 🤓 WELCOME 🤓 🤓
Please enter the number of students in this class.
A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students: 1

A class can have a minimum of 2 and a maximum of 25 students. Please input the correct number of students: q
INCORRECT INPUT!!! IT'S NOT AN INTEGER!!!
A class can have a minimum of 2 and a maximum of 25 students. Please input the correct number of students: 2
1. Student's name: Rob
1. Rob's grade: 101
############################################################
        🚫 INVALID ENTRY!!!
Please input a number between 1 and 100
############################################################
1. Rob's grade: 10

PLEASE CHOOSE FROM ONE OF THE FOLLOWING OPTIONS

###############################################################
①Press 1 to display average class grade.
②Press 2 to display lowest class grade.
③Press 3 to display highest class grade.
④Press 4 to sort & display the grades in ascending order.
⑤Press 5 to search for an individual student by name.
###############################################################
 ❌ Otherwise press X to quit the program ❌
###############################################################
```

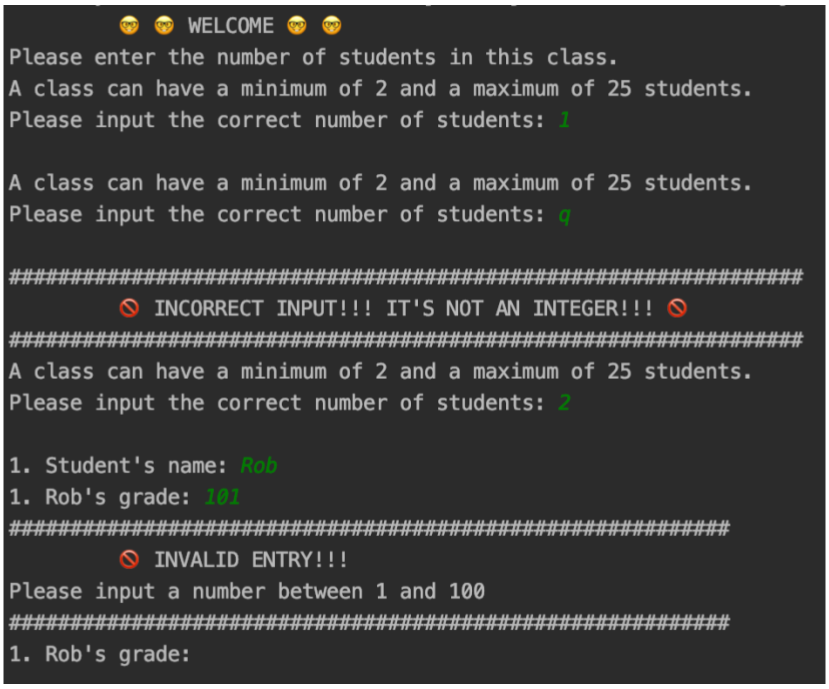
3rd attempt.

So I tried to combine the two previous attempts and it worked. Instead of using a Boolean to terminate the loop, I just used a range in the “while” loop. Let’s see the changes in the code:



```java
/*======================================= 3rd ATTEMPT!!! ========================================*/
// This is the one. Works fine. A combination of the first 2 attempts.
while(input<2 || input>25)
{

    System.out.print("A class can have a minimum of 2 and a maximum of 25 students. " +
            "\nPlease input the correct number of students: ");

    try
    {
        input = Integer.parseInt(in.next());
    }
    catch (Exception e)
    {
        System.out.println();
        System.out.println("#################################################################");
        System.out.println("         \uD83D\uDEAB INCORRECT INPUT!!! IT'S NOT AN INTEGER!!! \uD83D\uDEAB "); //"prohibited" emoji unicode
        System.out.println("#################################################################");
        continue; //skip loop
    }

    System.out.println();
}
```

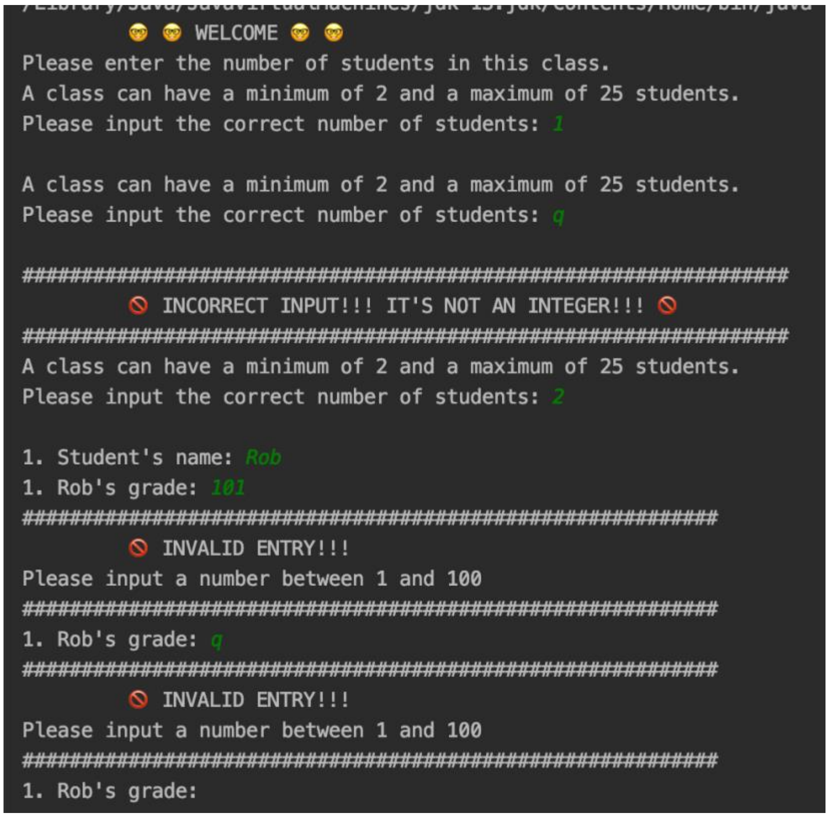
And the running program:



```
        🤓 🤓 WELCOME 🤓 🤓
Please enter the number of students in this class.
A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students: 1

A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students: q

#################################################################
         🚫 INCORRECT INPUT!!! IT'S NOT AN INTEGER!!! 🚫
#################################################################
A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students: 2

1. Student's name: Rob
1. Rob's grade: 101
###########################################################
         🚫 INVALID ENTRY!!!
Please input a number between 1 and 100
###########################################################
1. Rob's grade:
```

In the above image we can see that incorrect number has been input for the number of students, then letter instead of number, then a grade out of range and the program works as it should and prompts the user for the correct inputs.

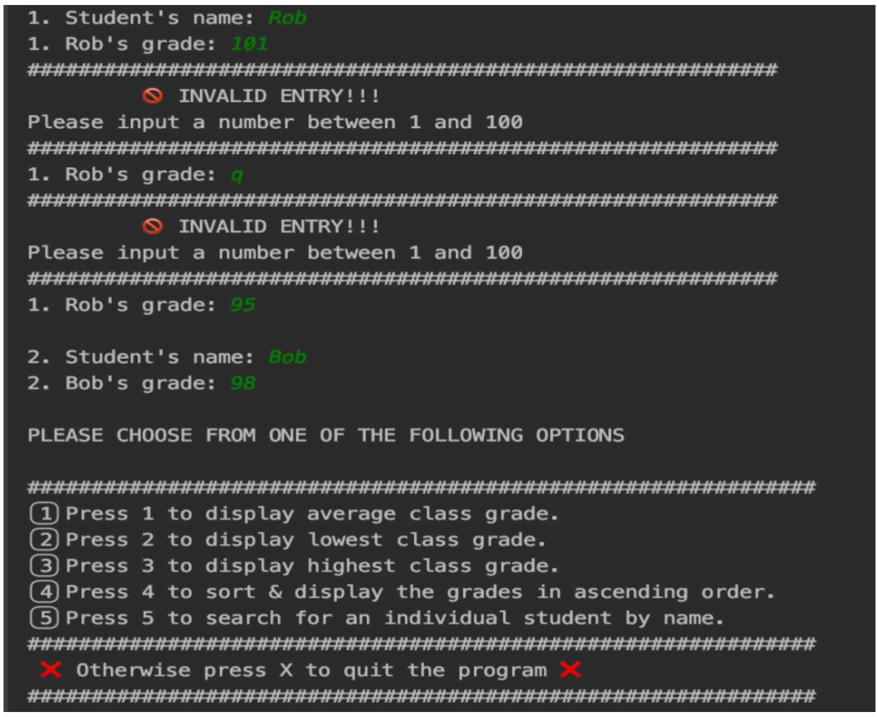
Next we will input letter instead of number for grade:



```
/Library/Java/JavaVirtualMachines/jdk-13.jdk/Contents/Home/bin/java
         🤓 🤓 WELCOME 🤓 🤓
Please enter the number of students in this class.
A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students: 1

A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students: q

##################################################################
         🚫 INCORRECT INPUT!!! IT'S NOT AN INTEGER!!! 🚫
##################################################################
A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students: 2

1. Student's name: Rob
1. Rob's grade: 101
############################################################
         🚫 INVALID ENTRY!!!
Please input a number between 1 and 100
############################################################
1. Rob's grade: q
############################################################
         🚫 INVALID ENTRY!!!
Please input a number between 1 and 100
############################################################
1. Rob's grade:
```

Still on the right track as the program has a try/catch in the method used to enter the student info (see enterInfo() method below). As a result the same message pops up stating that we have an invalid input and to try again.

Next we will input a correct grade range and see if the program asks us to input the next student and grade or jumps ahead as in the 2nd attempt.



```
1. Student's name: Rob
1. Rob's grade: 101
############################################################
         🚫 INVALID ENTRY!!!
Please input a number between 1 and 100
############################################################
1. Rob's grade: q
############################################################
         🚫 INVALID ENTRY!!!
Please input a number between 1 and 100
############################################################
1. Rob's grade: 95

2. Student's name: Bob
2. Bob's grade: 98

PLEASE CHOOSE FROM ONE OF THE FOLLOWING OPTIONS

###############################################################
①Press 1 to display average class grade.
②Press 2 to display lowest class grade.
③Press 3 to display highest class grade.
④Press 4 to sort & display the grades in ascending order.
⑤Press 5 to search for an individual student by name.
###############################################################
 ❌ Otherwise press X to quit the program ❌
###############################################################
```

As you can see in the screenshot above, after inputting a grade from the correct range, the program loops back as it should and asks to input the details of the second student.

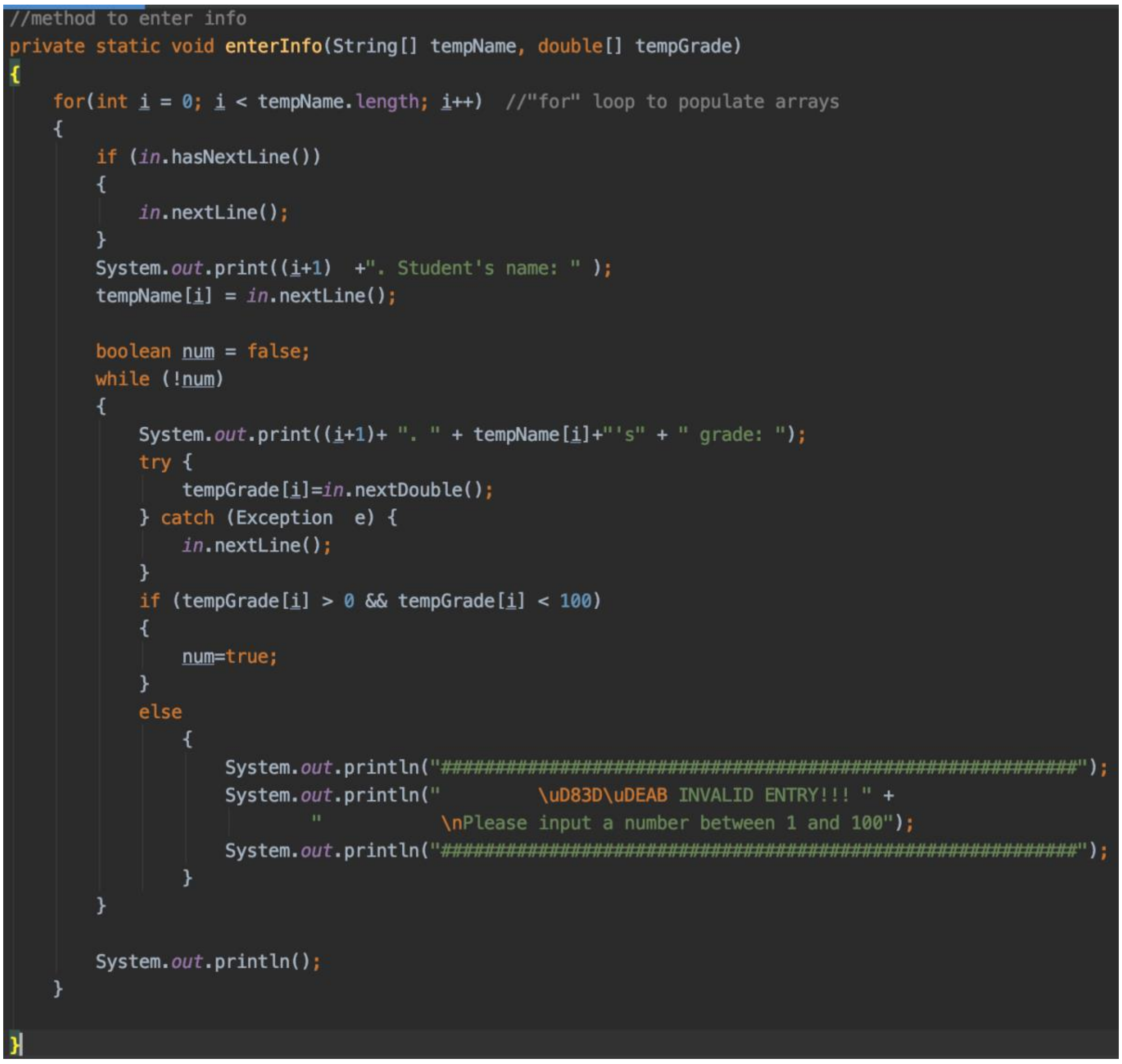
These were all the problems that I encountered.

Methods

Method to *enterInfo()*

This method is used to populate two arrays. One with the student's name and the other one with the student's grade.

A try/catch is used for inputting the student's grade, to catch all exceptions other than integer numbers between 1 and 100.

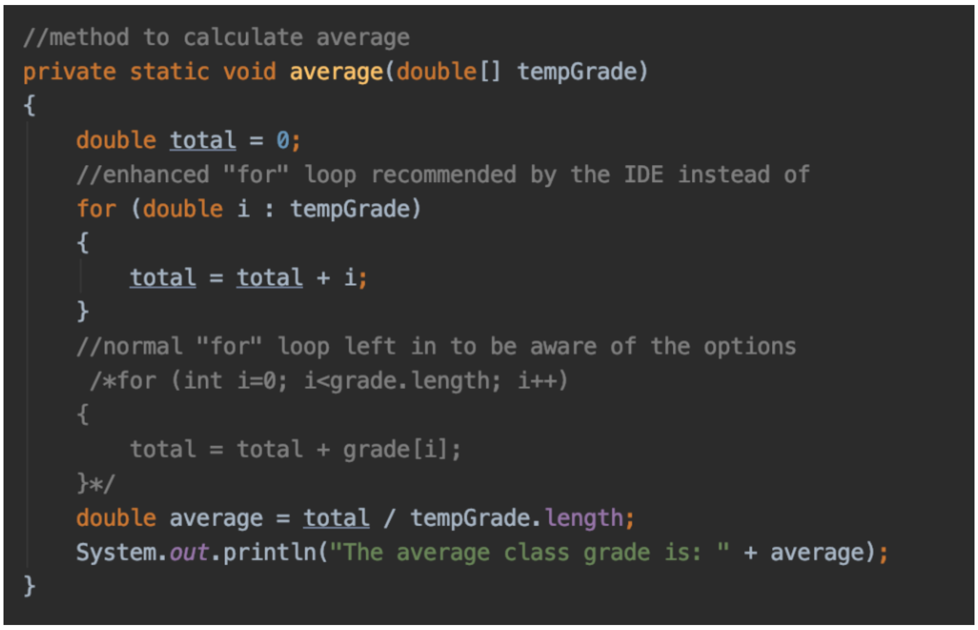


```java
//method to enter info
private static void enterInfo(String[] tempName, double[] tempGrade)
{
    for(int i = 0; i < tempName.length; i++)  //"for" loop to populate arrays
    {
        if (in.hasNextLine())
        {
            in.nextLine();
        }
        System.out.print((i+1)  +". Student's name: " );
        tempName[i] = in.nextLine();

        boolean num = false;
        while (!num)
        {
            System.out.print((i+1)+ ". " + tempName[i]+"'s" + " grade: ");
            try {
                tempGrade[i]=in.nextDouble();
            } catch (Exception  e) {
                in.nextLine();
            }
            if (tempGrade[i] > 0 && tempGrade[i] < 100)
            {
                num=true;
            }
            else
                {
                    System.out.println("###########################################################");
                    System.out.println("         \uD83D\uDEAB INVALID ENTRY!!! " +
                            "           \nPlease input a number between 1 and 100");
                    System.out.println("###########################################################");
                }
        }

        System.out.println();
    }

}
```

Method to calculate *average()*

This method calculates the average grade of the class. So it actually ads up all numbers from within the grade array and divides it to the number of elements from that array. I used an enhanced “for” loop instead of a normal “for” loop, as it was suggested to me by the IDE that I use (IntelliJ). I commented out the normal “for” loop and left it in the code to know all options available for me, if I ever come back to this program for examples.

Here is the code:

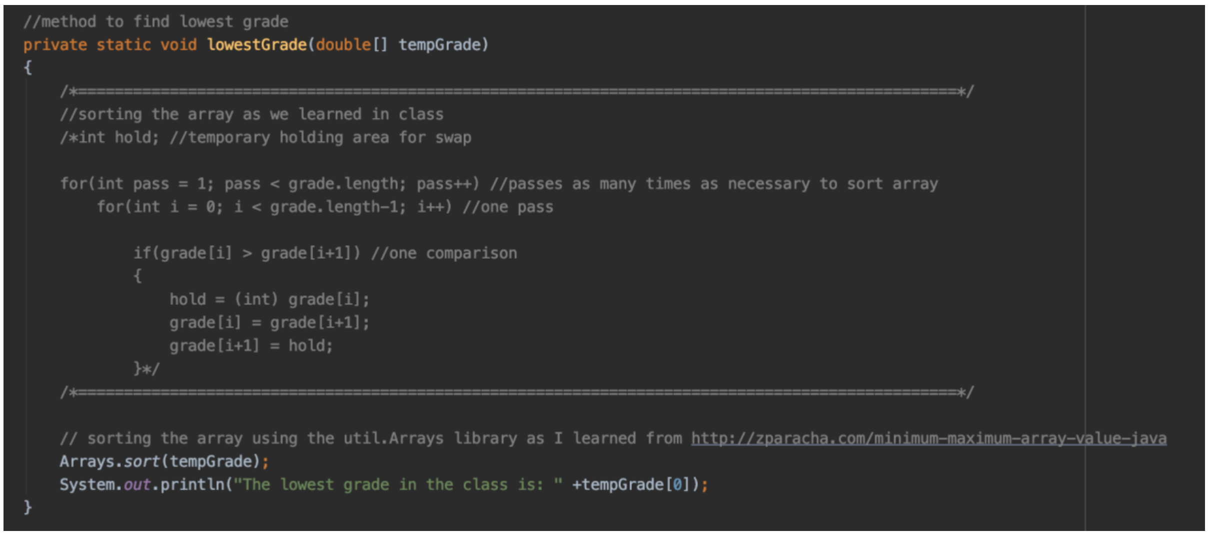


```
//method to calculate average
private static void average(double[] tempGrade)
{
    double total = 0;
    //enhanced "for" loop recommended by the IDE instead of
    for (double i : tempGrade)
    {
        total = total + i;
    }
    //normal "for" loop left in to be aware of the options
     /*for (int i=0; i<grade.length; i++)
    {
        total = total + grade[i];
    }*/
    double average = total / tempGrade.length;
    System.out.println("The average class grade is: " + average);
}
```

Method to display *lowestGrade()*

This method is to display the lowest grade from the class/grade array. First we sort the numbers within the array into ascending order and then output to the screen the first element of the array. To sort the array I used the “util.Array” library, but I left in and commented out the method of sorting arrays in the way we learned in class too, just for future reference, if I ever need it.

Code:

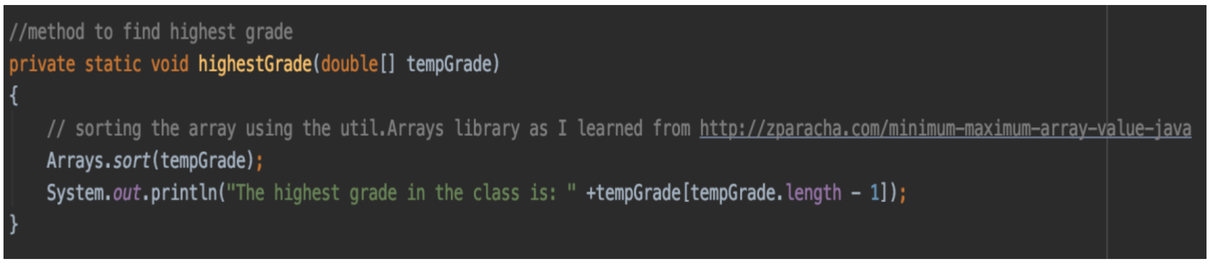


```java
//method to find lowest grade
private static void lowestGrade(double[] tempGrade)
{
    /*==========================================================================================*/
    //sorting the array as we learned in class
    /*int hold; //temporary holding area for swap

    for(int pass = 1; pass < grade.length; pass++) //passes as many times as necessary to sort array
        for(int i = 0; i < grade.length-1; i++) //one pass

            if(grade[i] > grade[i+1]) //one comparison
            {
                hold = (int) grade[i];
                grade[i] = grade[i+1];
                grade[i+1] = hold;
            }*/
    /*==========================================================================================*/

    // sorting the array using the util.Arrays library as I learned from http://zparacha.com/minimum-maximum-array-value-java
    Arrays.sort(tempGrade);
    System.out.println("The lowest grade in the class is: " +tempGrade[0]);
}
```

Method to display *highestGrade()*

Same way as the lowest grade above only that it will display the last element of the array (array.lenght -1).

Code:

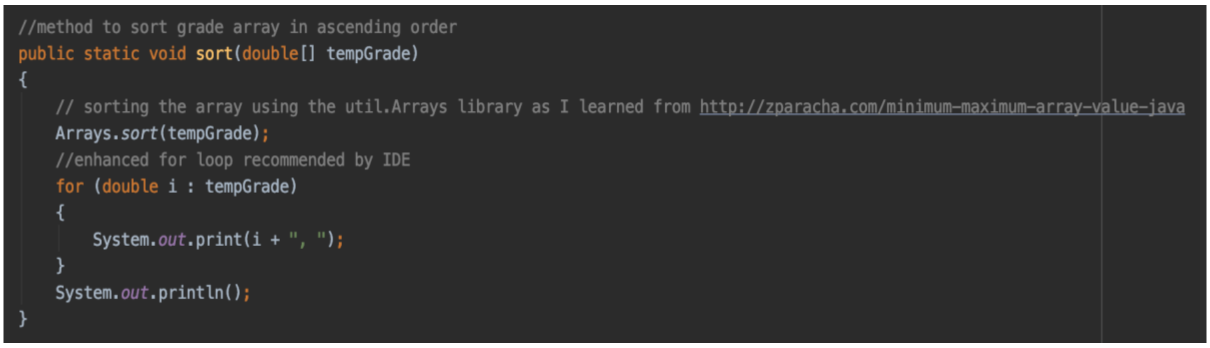


```java
//method to find highest grade
private static void highestGrade(double[] tempGrade)
{
    // sorting the array using the util.Arrays library as I learned from http://zparacha.com/minimum-maximum-array-value-java
    Arrays.sort(tempGrade);
    System.out.println("The highest grade in the class is: " +tempGrade[tempGrade.length - 1]);
}
```

Method to *sort()* the grade array

This method sorts the array in ascending order the same way as in the above 2 methods and then outputs the elements of the array to the screen using a “for” loop.

Code:

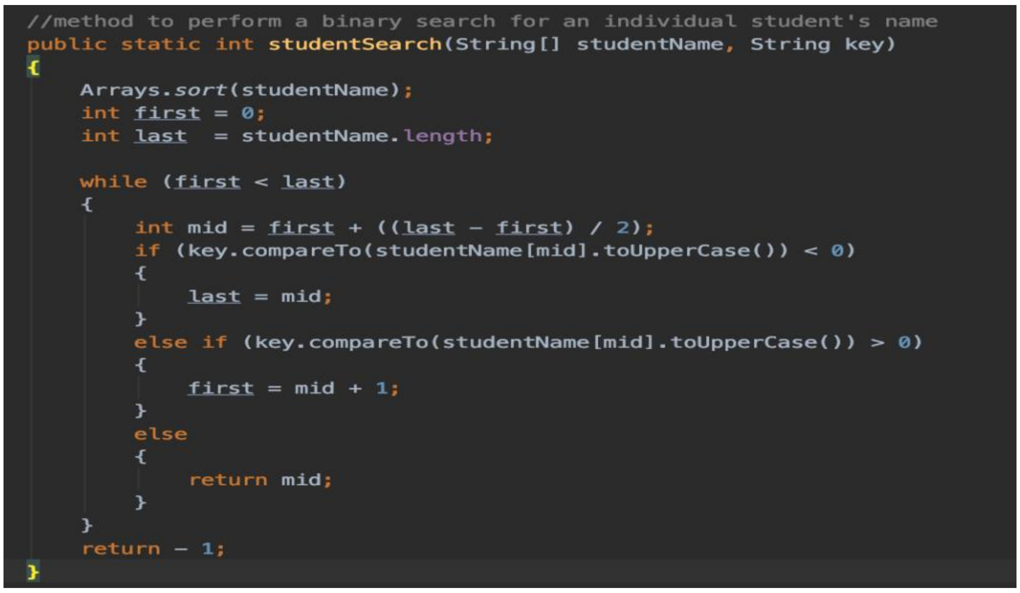


```java
//method to sort grade array in ascending order
public static void sort(double[] tempGrade)
{
    // sorting the array using the util.Arrays library as I learned from http://zparacha.com/minimum-maximum-array-value-java
    Arrays.sort(tempGrade);
    //enhanced for loop recommended by IDE
    for (double i : tempGrade)
    {
        System.out.print(i + ", ");
    }
    System.out.println();
}
```

Method to allow to perform a *studentSearch()*

This method allows the user to perform a binary search of all the users from the String array. To be more efficient, we first sort the elements of the array.

Code:

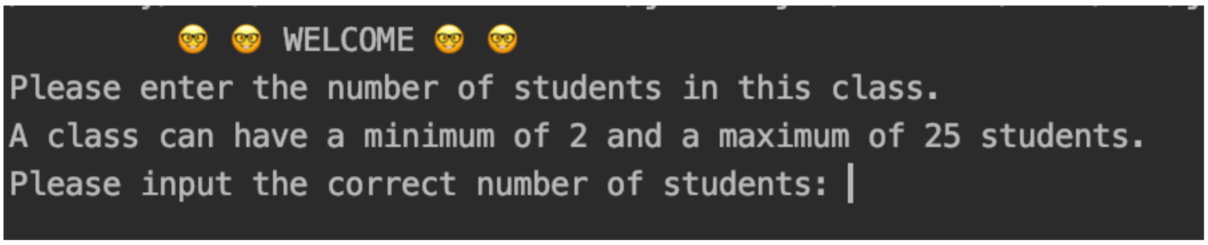


```java
//method to perform a binary search for an individual student's name
public static int studentSearch(String[] studentName, String key)
{
    Arrays.sort(studentName);
    int first = 0;
    int last  = studentName.length;

    while (first < last)
    {
        int mid = first + ((last - first) / 2);
        if (key.compareTo(studentName[mid].toUpperCase()) < 0)
        {
            last = mid;
        }
        else if (key.compareTo(studentName[mid].toUpperCase()) > 0)
        {
            first = mid + 1;
        }
        else
        {
            return mid;
        }
    }
    return - 1;
}
```

Illustration of the program running

Inputting number of students

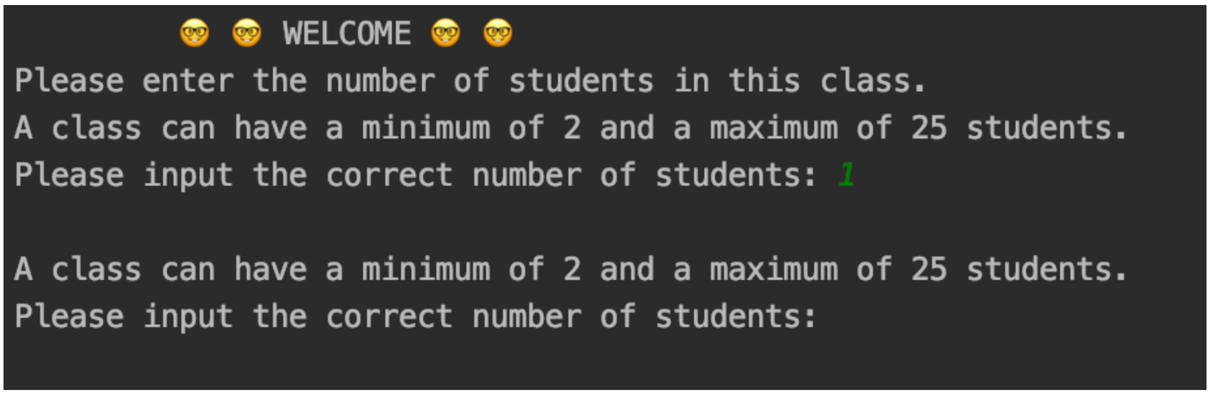
First we are asked by the program to input a number between 2 and 25.



```
       🤓 🤓 WELCOME 🤓 🤓
Please enter the number of students in this class.
A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students: |
```

Inputting the incorrect number of students

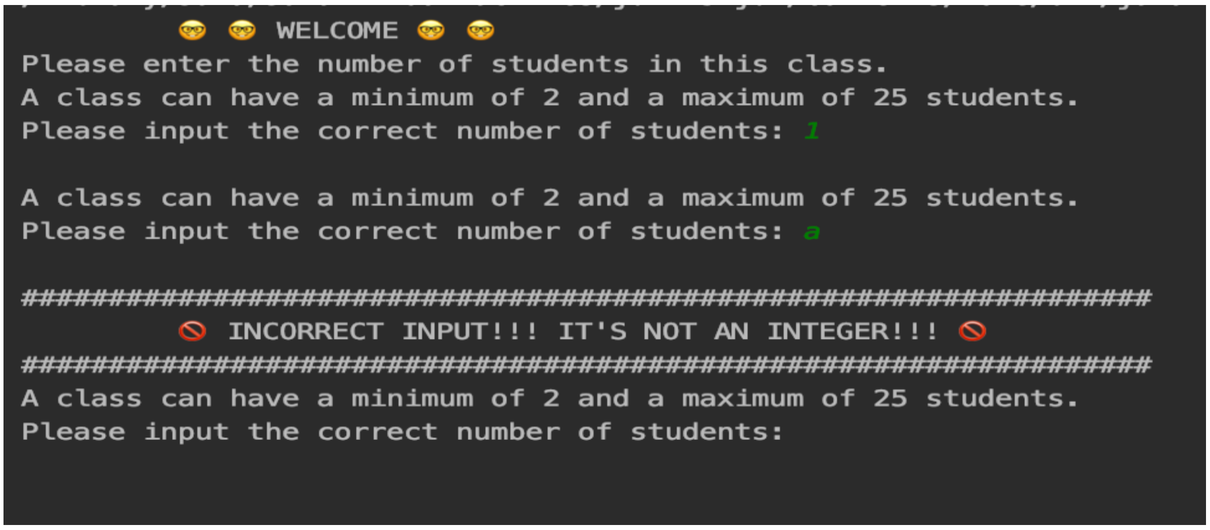
By inputting a number outside the recommended range we get a message prompting the user to try again.



```
        🤓 🤓 WELCOME 🤓 🤓
Please enter the number of students in this class.
A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students: 1

A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students:
```

Inputting a letter instead of a number

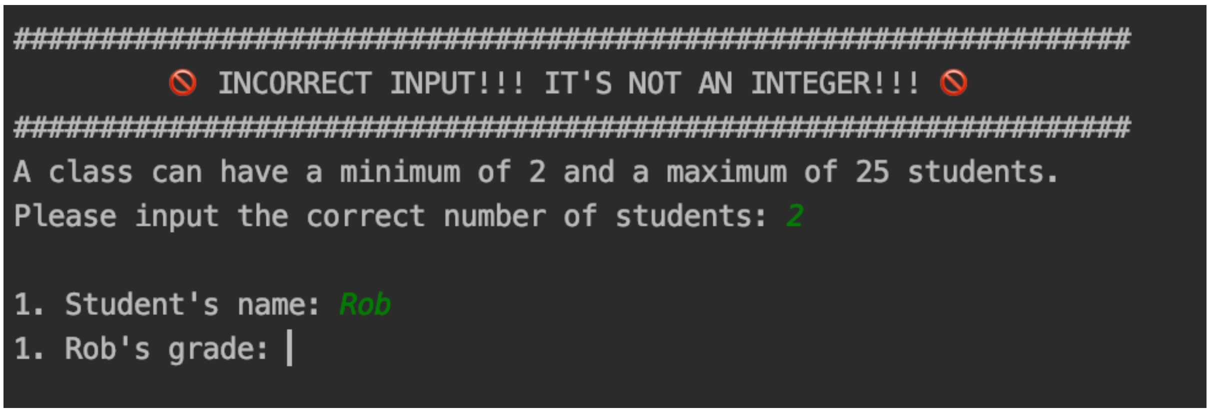
If by mistake we input a letter instead of a number, we get another message warning us that it is an incorrect input and prompting us to try again



```
          🤓 🤓 WELCOME 🤓 🤓
Please enter the number of students in this class.
A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students: 1

A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students: a

#################################################################
         🚫 INCORRECT INPUT!!! IT'S NOT AN INTEGER!!! 🚫
#################################################################
A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students:
```

Entering student's name and grade

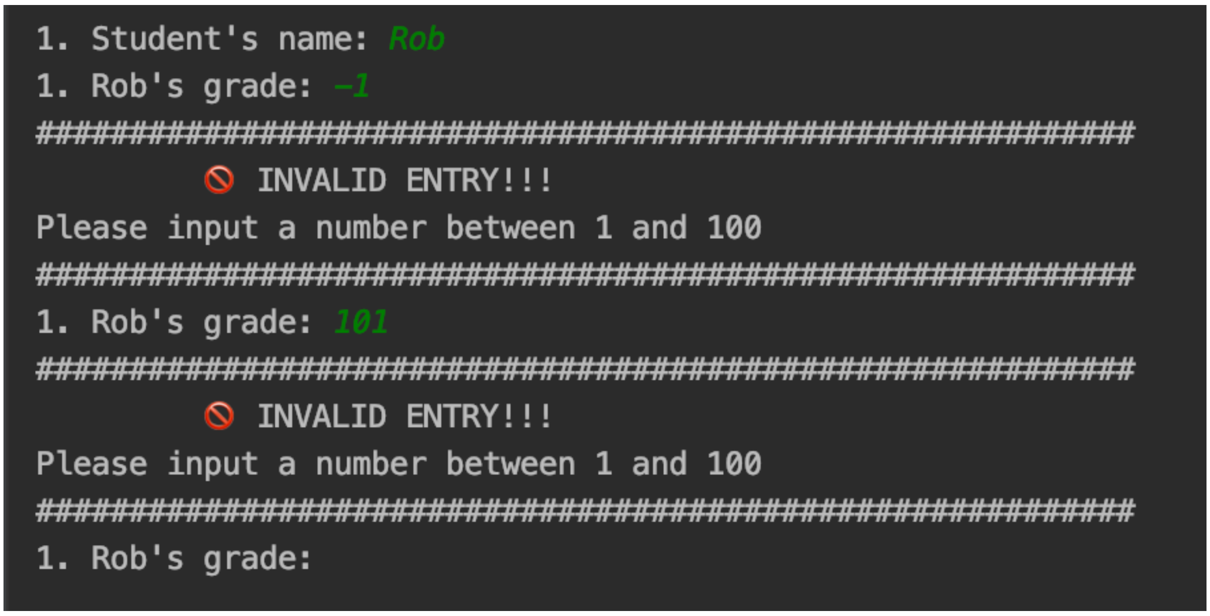
Once we input a number between 2 and 25 (from the correct range), we are prompted to input the student's name and grade.



```
#################################################################
         🚫 INCORRECT INPUT!!! IT'S NOT AN INTEGER!!! 🚫
#################################################################
A class can have a minimum of 2 and a maximum of 25 students.
Please input the correct number of students: 2

1. Student's name: Rob
1. Rob's grade: |
```

Inputting a grade out of range (grade <1 && grade >100)

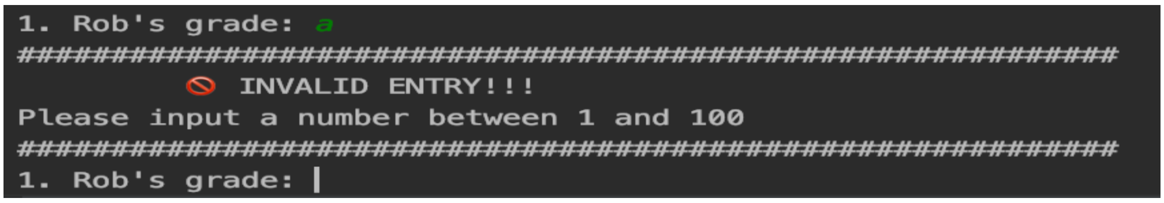
If the user inputs a grade lower than 1 or higher than 100, a message pops up prompting user to input a grade between 1 and 100.



```
1. Student's name: Rob
1. Rob's grade: -1
###########################################################
         🚫 INVALID ENTRY!!!
Please input a number between 1 and 100
###########################################################
1. Rob's grade: 101
###########################################################
         🚫 INVALID ENTRY!!!
Please input a number between 1 and 100
###########################################################
1. Rob's grade:
```

Inputting a letter instead of a number

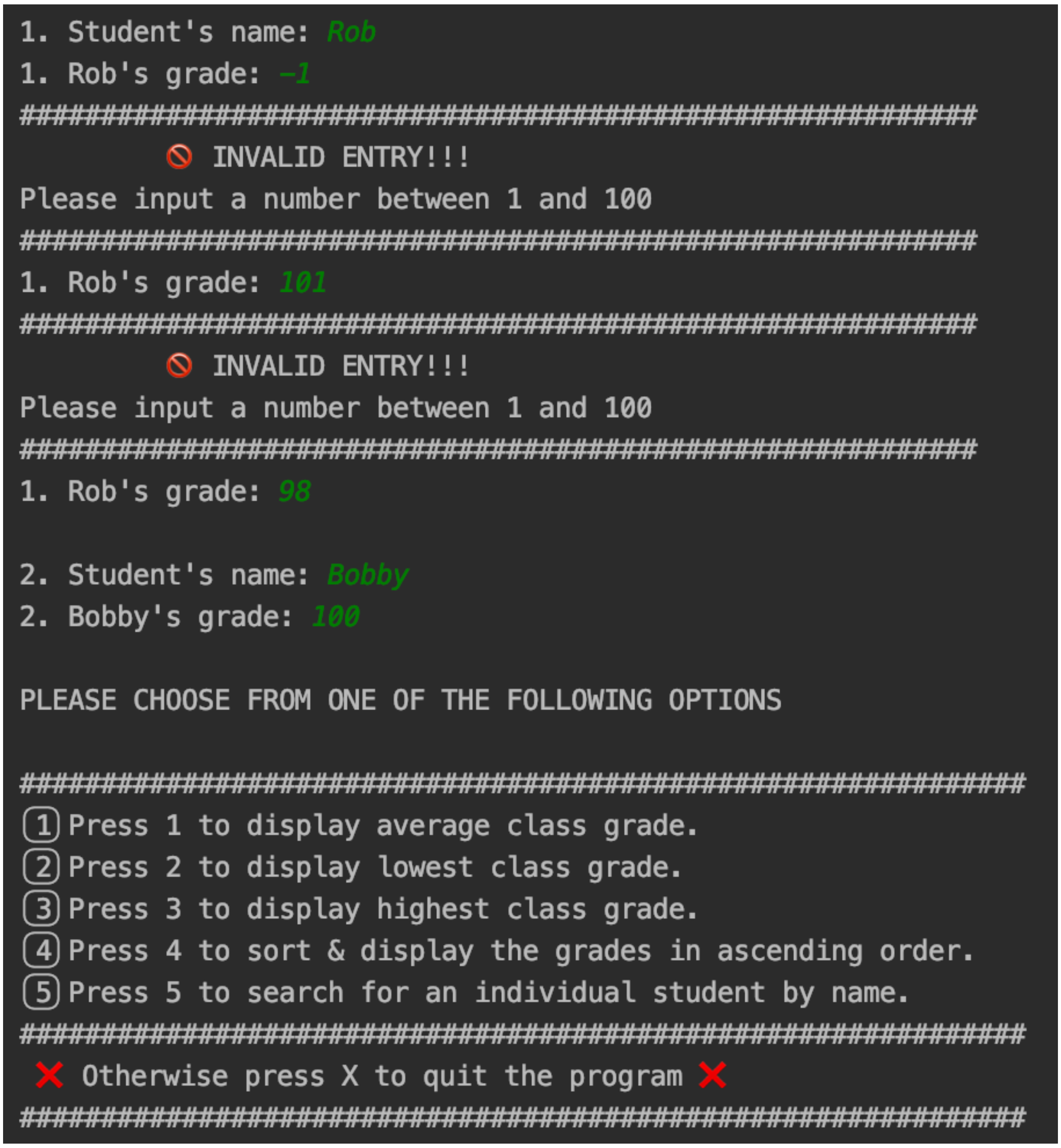
If a letter is input by mistake, instead of a grade, the same thing happens.



```
1. Rob's grade: a
###########################################################
         🚫 INVALID ENTRY!!!
Please input a number between 1 and 100
###########################################################
1. Rob's grade: |
```

Displaying a menu of options

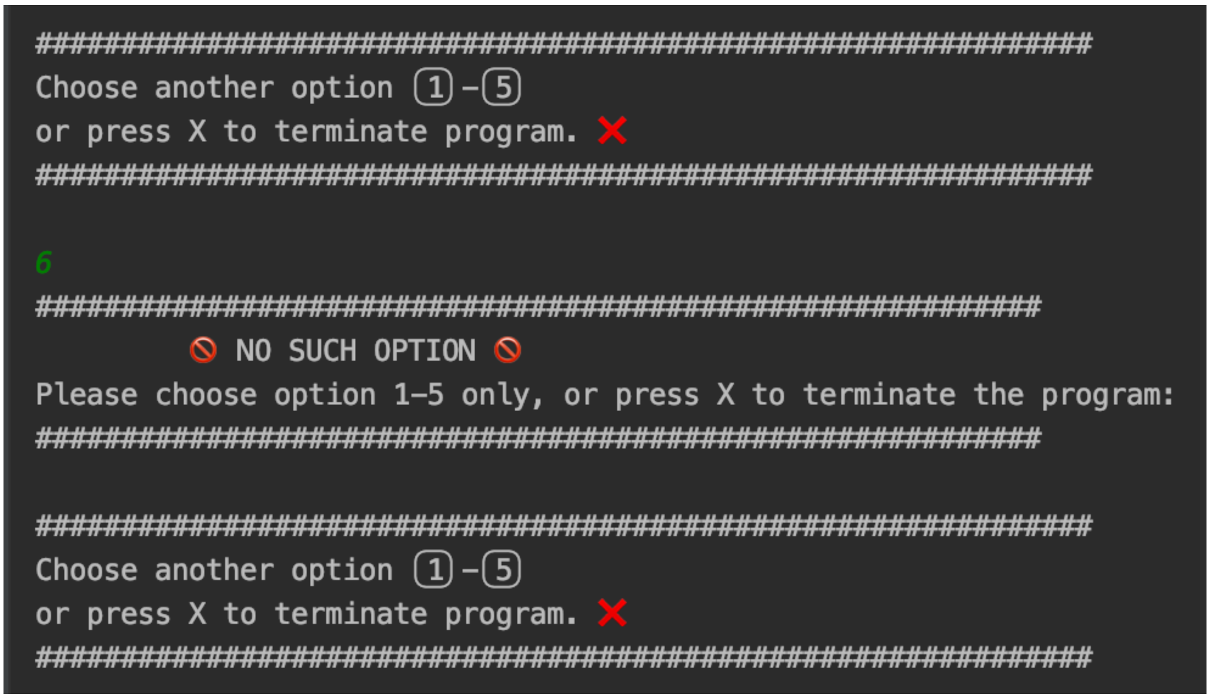
Once all the student’s have been entered and all of them were assigned a correct grade, an option menu appears with 6 option. Option 1 to 5 will run one of the above mentioned methods, while the sixth option is to press ”X” and terminate the program.



```
1. Student's name: Rob
1. Rob's grade: -1
############################################################
         🚫 INVALID ENTRY!!!
Please input a number between 1 and 100
############################################################
1. Rob's grade: 101
############################################################
         🚫 INVALID ENTRY!!!
Please input a number between 1 and 100
############################################################
1. Rob's grade: 98

2. Student's name: Bobby
2. Bobby's grade: 100

PLEASE CHOOSE FROM ONE OF THE FOLLOWING OPTIONS

###############################################################
①Press 1 to display average class grade.
②Press 2 to display lowest class grade.
③Press 3 to display highest class grade.
④Press 4 to sort & display the grades in ascending order.
⑤Press 5 to search for an individual student by name.
###############################################################
 ❌ Otherwise press X to quit the program ❌
###############################################################
```

Wrong option

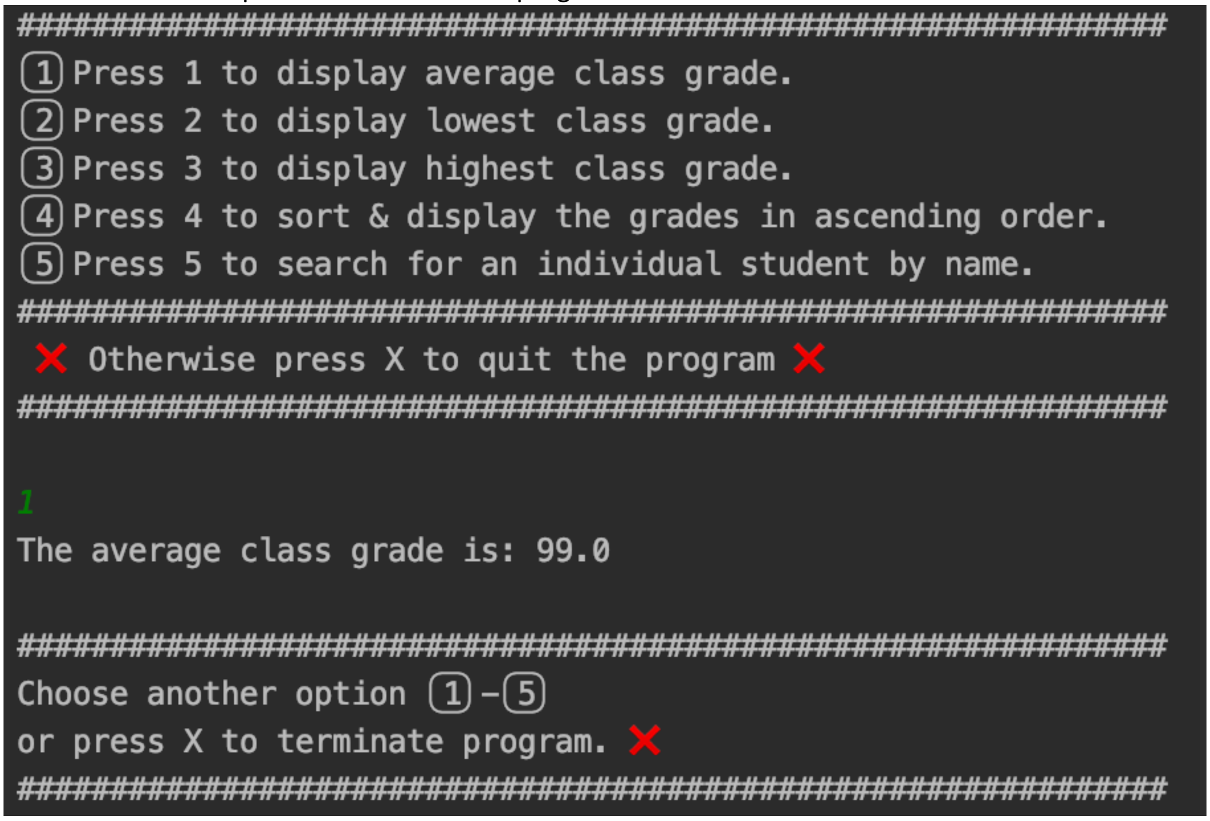
If user inputs another option other than 1to 5 or “X”, a message will come up stating it.



```
##############################################################
Choose another option ①-⑤
or press X to terminate program. ❌
##############################################################


6
###########################################################
         🚫 NO SUCH OPTION 🚫
Please choose option 1-5 only, or press X to terminate the program:
###########################################################

##############################################################
Choose another option ①-⑤
or press X to terminate program. ❌
##############################################################
```

Option 1

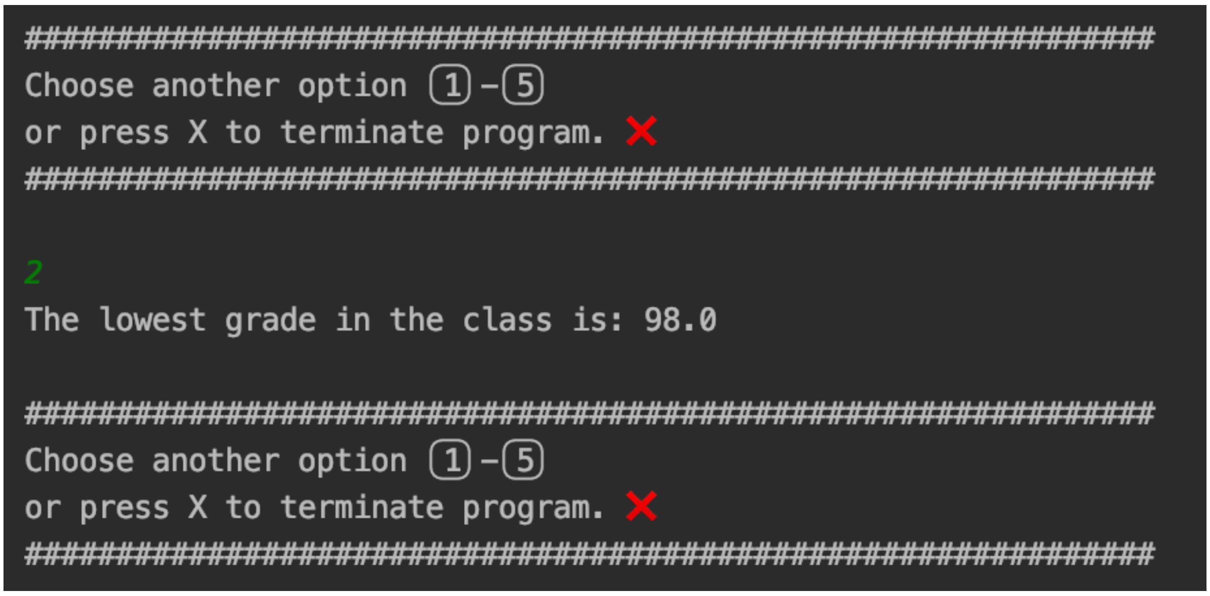
Will run method average() and will display the class average grade. After it will prompt user to choose a different option or terminate the program.



```
##############################################################
①Press 1 to display average class grade.
②Press 2 to display lowest class grade.
③Press 3 to display highest class grade.
④Press 4 to sort & display the grades in ascending order.
⑤Press 5 to search for an individual student by name.
##############################################################
 ❌ Otherwise press X to quit the program ❌
##############################################################


1
The average class grade is: 99.0

##############################################################
Choose another option ①-⑤
or press X to terminate program. ❌
##############################################################
```

Option 2

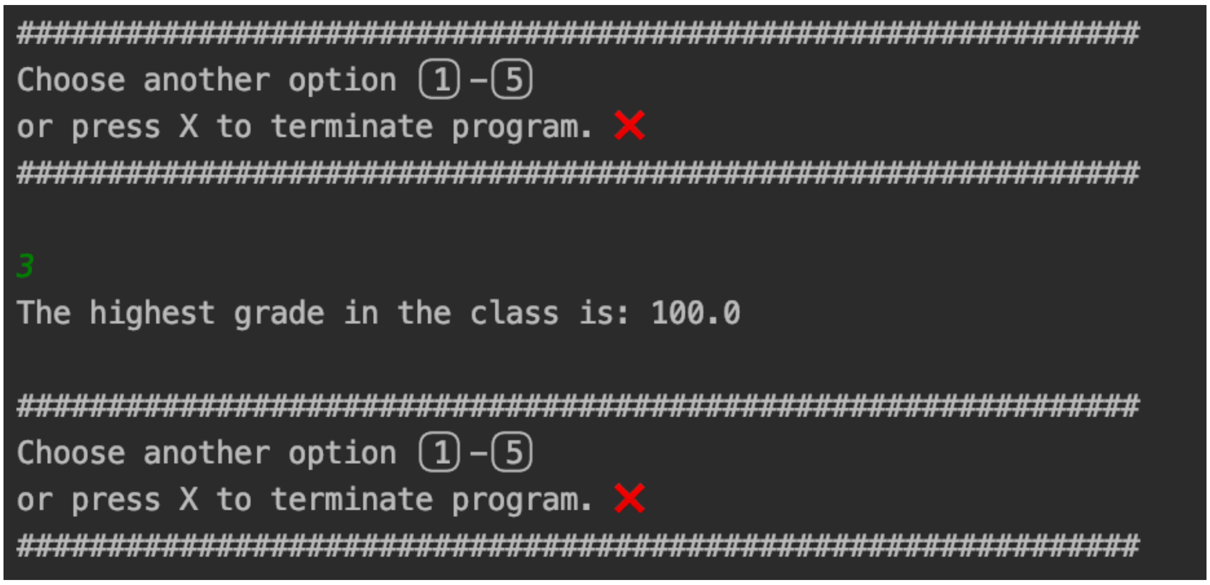
Will run method lowestGrade() and will display the lowest grade. After it the same prompt will appear, asking the user to choose a different option or terminate the program.



```
##############################################################
Choose another option ①-⑤
or press X to terminate program. ❌
##############################################################

2
The lowest grade in the class is: 98.0

##############################################################
Choose another option ①-⑤
or press X to terminate program. ❌
##############################################################
```

Option 3

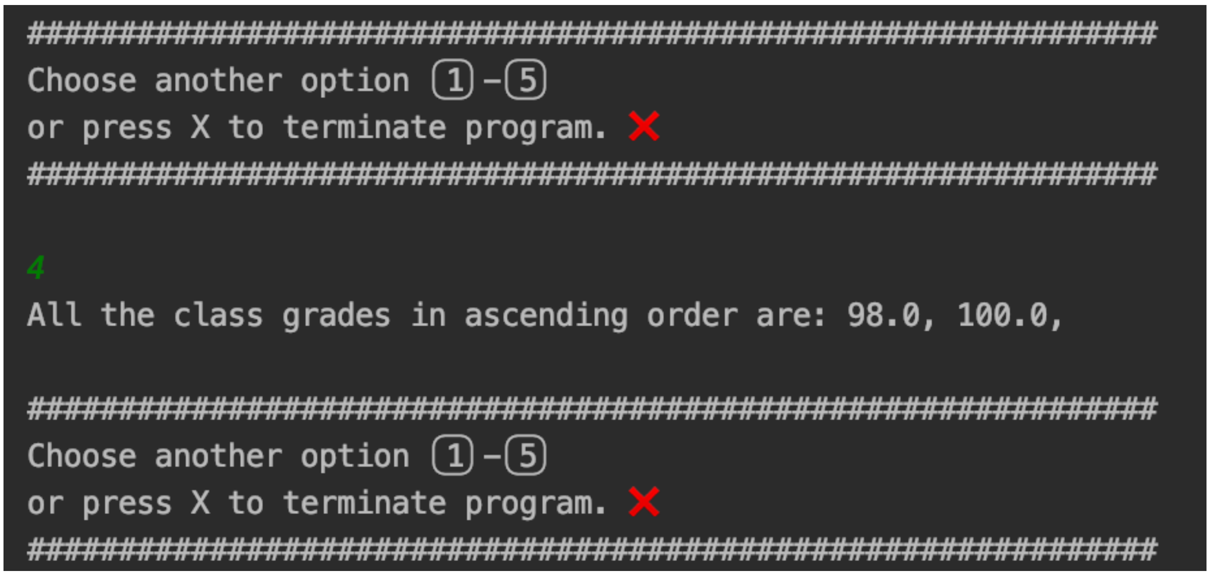
Will run method highestGrade() and will display the highest grade. After it the same prompt will appear, asking the user to choose a different option or terminate the program.



```
##############################################################
Choose another option ①-⑤
or press X to terminate program. ❌
##############################################################

3
The highest grade in the class is: 100.0

##############################################################
Choose another option ①-⑤
or press X to terminate program. ❌
##############################################################
```

Option 4

Will run method sort() and will display all the grades in ascending order. After it the same prompt will appear, asking the user to choose a different option or terminate the program.



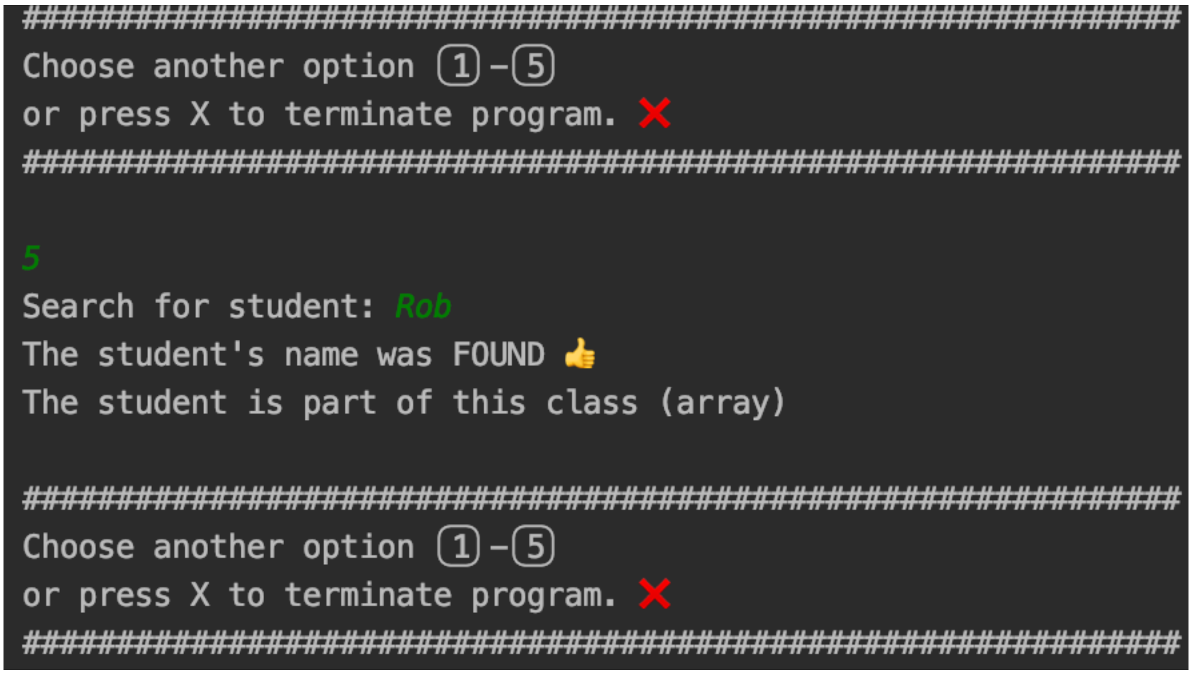
```
##############################################################
Choose another option 1️⃣-5️⃣
or press X to terminate program. ❌
##############################################################


4
All the class grades in ascending order are: 98.0, 100.0,

##############################################################
Choose another option 1️⃣-5️⃣
or press X to terminate program. ❌
##############################################################
```

Option 5

Will run method studentSearch() prompting user to search for a student from the class/string array.

Found

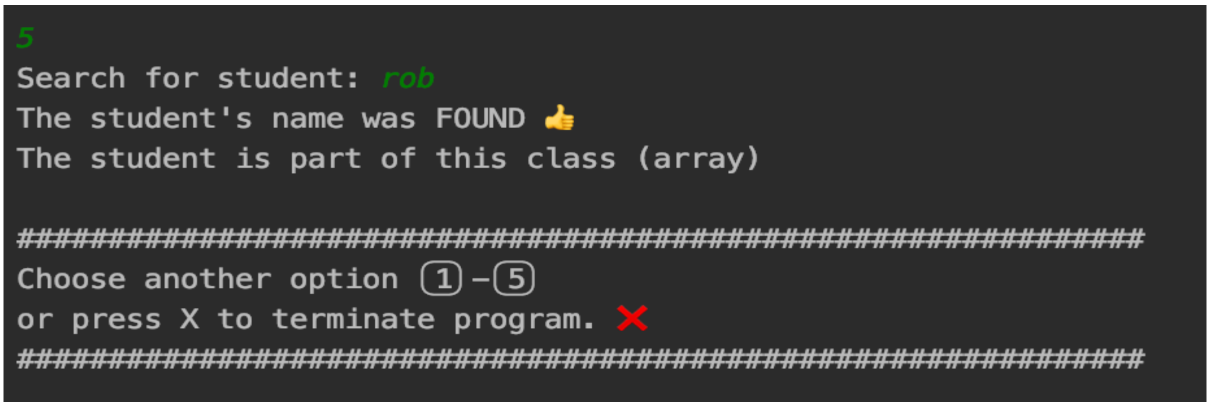
If found the following will be displayed:



```
##############################################################
Choose another option 1️⃣-5️⃣
or press X to terminate program. ❌
##############################################################


5
Search for student: Rob
The student's name was FOUND 👍
The student is part of this class (array)

##############################################################
Choose another option 1️⃣-5️⃣
or press X to terminate program. ❌
##############################################################
```

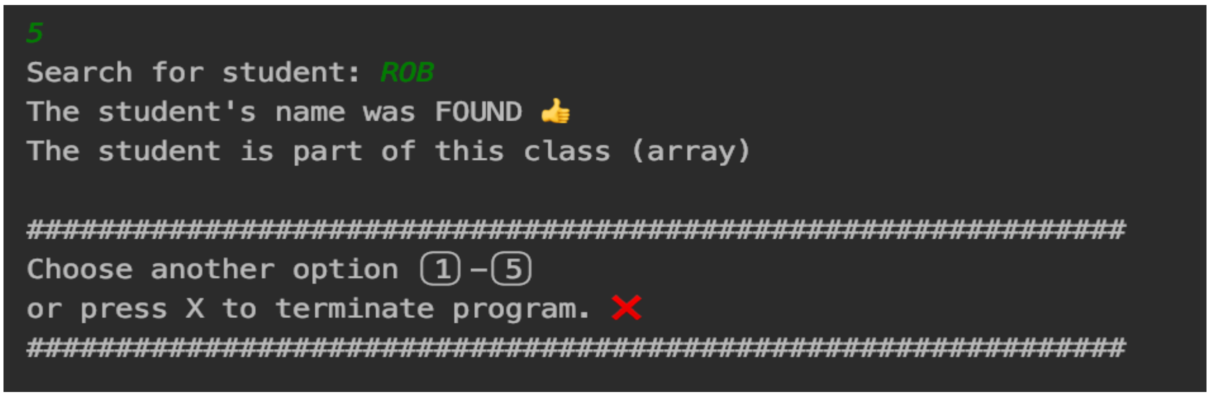
Even if the user don't use capital letters or uses only capital letter when searching for a student, the program will find the student as in the binary method the string array search is casted to upper case.

Searching by all lower case:



```
5
Search for student: rob
The student's name was FOUND 👍
The student is part of this class (array)

################################################################
Choose another option ①-⑤
or press X to terminate program. ❌
################################################################
```

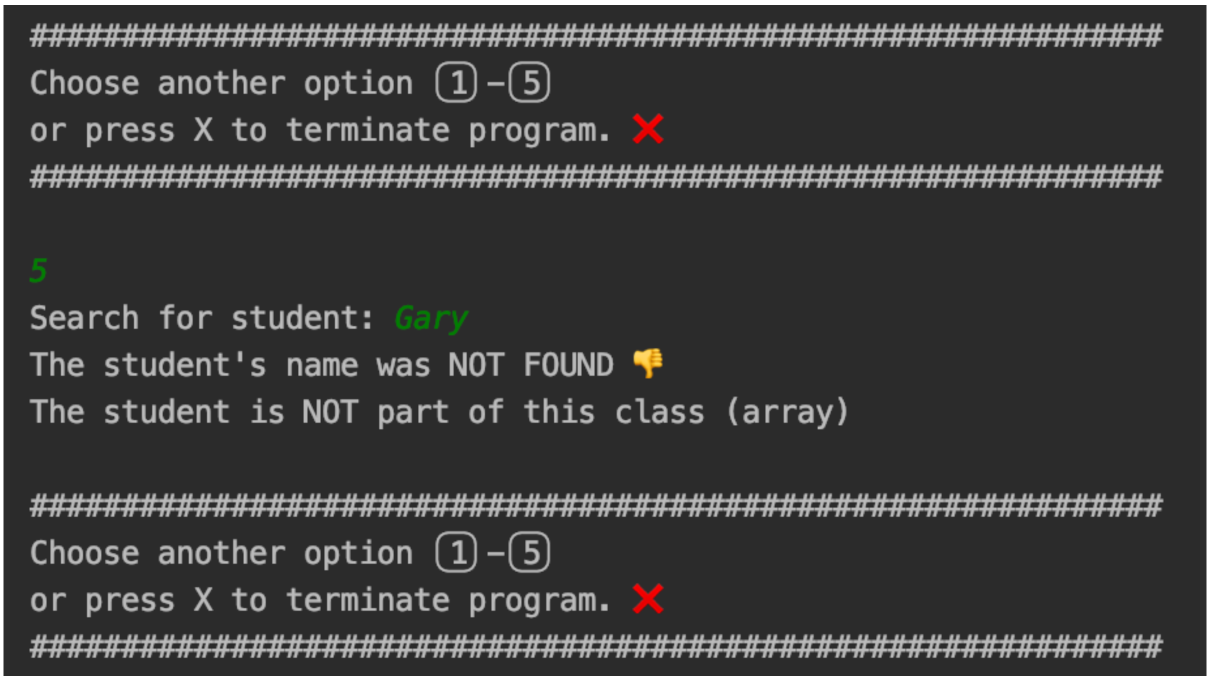
Searching by all upper case:



```
5
Search for student: ROB
The student's name was FOUND 👍
The student is part of this class (array)

################################################################
Choose another option ①-⑤
or press X to terminate program. ❌
################################################################
```

Not found

If not found the following will be displayed:



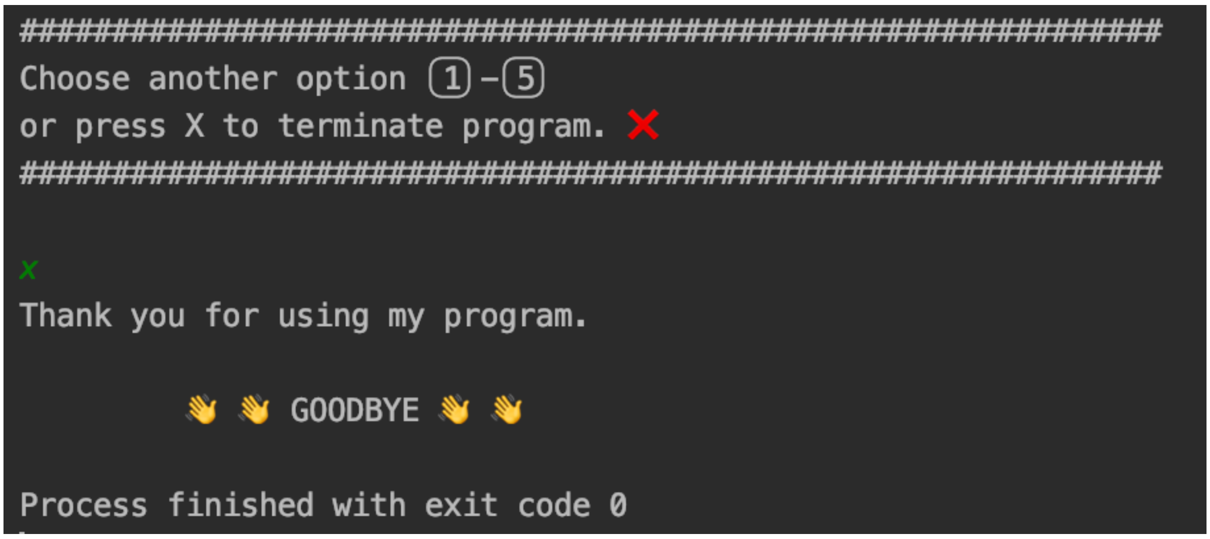
```
################################################################
Choose another option ①-⑤
or press X to terminate program. ❌
################################################################

5
Search for student: Gary
The student's name was NOT FOUND 👎
The student is NOT part of this class (array)

################################################################
Choose another option ①-⑤
or press X to terminate program. ❌
################################################################
```

Option X

By pressing “X” the program terminates and a goodbye message is displayed.

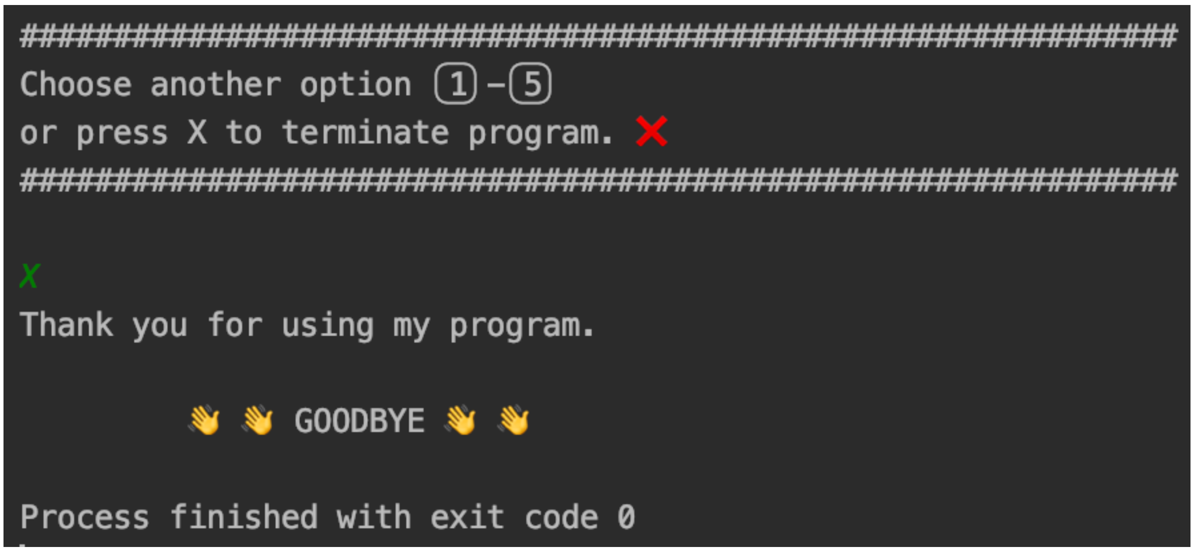
Same as in the student search method, regardless if a lower case or upper case “X” is pressed, the program will still terminate

Lower case “x”:



```
####################################################################
Choose another option ①-⑤
or press X to terminate program. ❌
####################################################################


x
Thank you for using my program.

         👋 👋 GOODBYE 👋 👋

Process finished with exit code 0
```

Upper case “X”:



```
####################################################################
Choose another option ①-⑤
or press X to terminate program. ❌
####################################################################


X
Thank you for using my program.

         👋 👋 GOODBYE 👋 👋

Process finished with exit code 0
```