Class Marks

Matthias Bartolo
The Archbishop's Seminary School
May 2019

Table of Contents

Definition of the Problem

Description of the Problem

The scope of my program is to be a database for all student information and marks obtained. The user can enter how many students he wants. The program finds the student's grades. The program also calculates the average, sum, maximum, minimum marks and offers a search menu to find students.

This program is intended for adults, who have a profession in teaching and want to keep a database on students.

Statement of Results Required

The program outputs the following:

1. Menu
2. Message of all of the student information and the grades obtained.
3. Message of the average and sum obtained from the student marks.
4. Message of the student information who got the highest and lowest marks.
5. Message of all the students and whether they were above or below average.
6. SearchMenu
7. Message to Search by mark and shows the user the information about the mark asked for.
8. Message to Search by name and shows the user the information about the name asked for.
9. Message to Search by class and shows the user the information about the class asked for.
10. If the Option doesn't exist, the program will display 'Invalid'

Details of the Input Information Required

To work the following inputs are required:

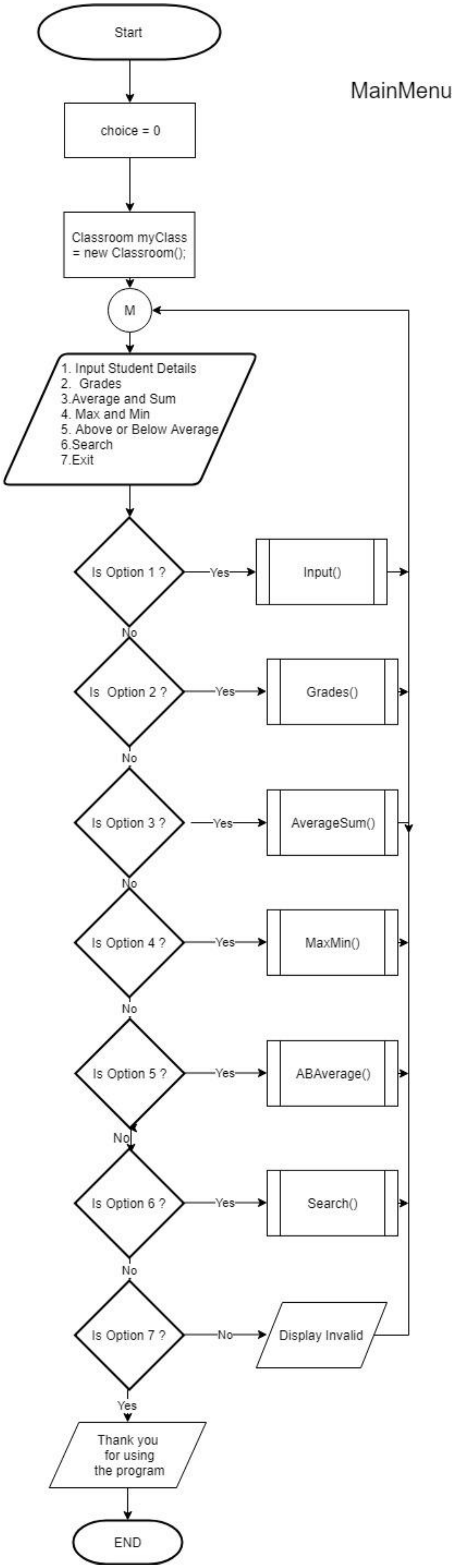
1. Menu option.
2. The number of students to input.
3. Names, marks, classes, locations of each student.
4. The SearchMenu option.
5. The mark to be searched for.
6. The name to be searched for.
7. The class to be searched for.

Solution of the Problem

Algorithm of the program

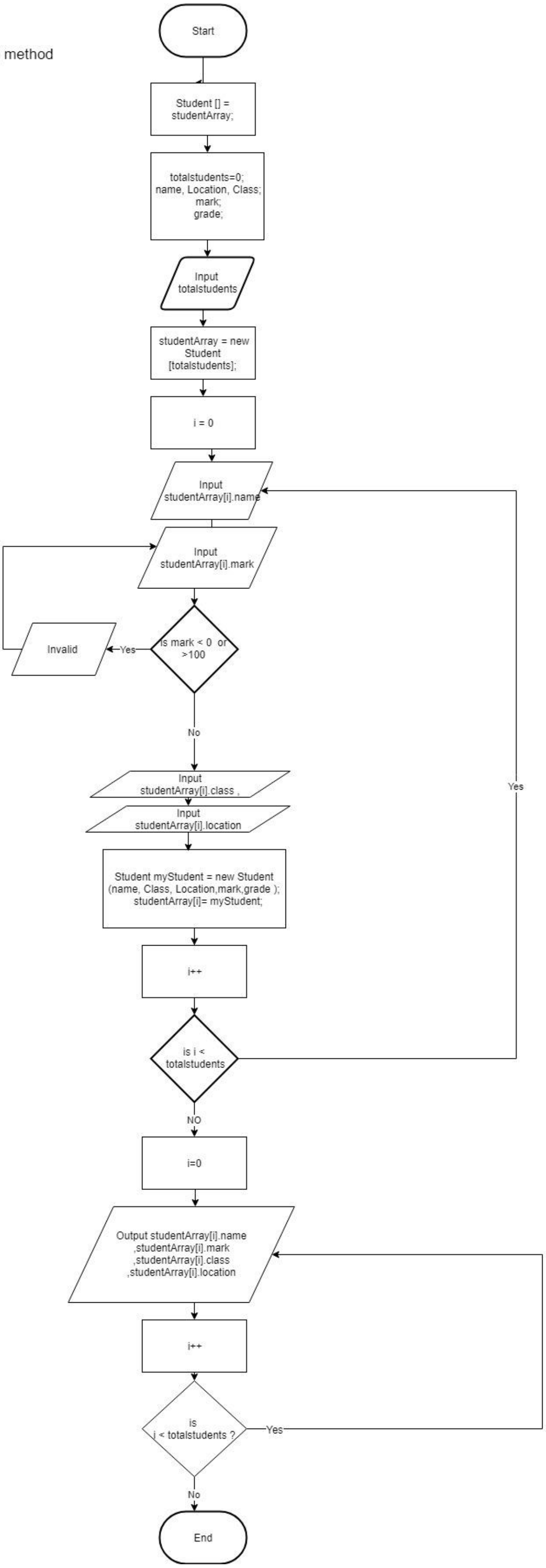
Main Menu

Class Marks

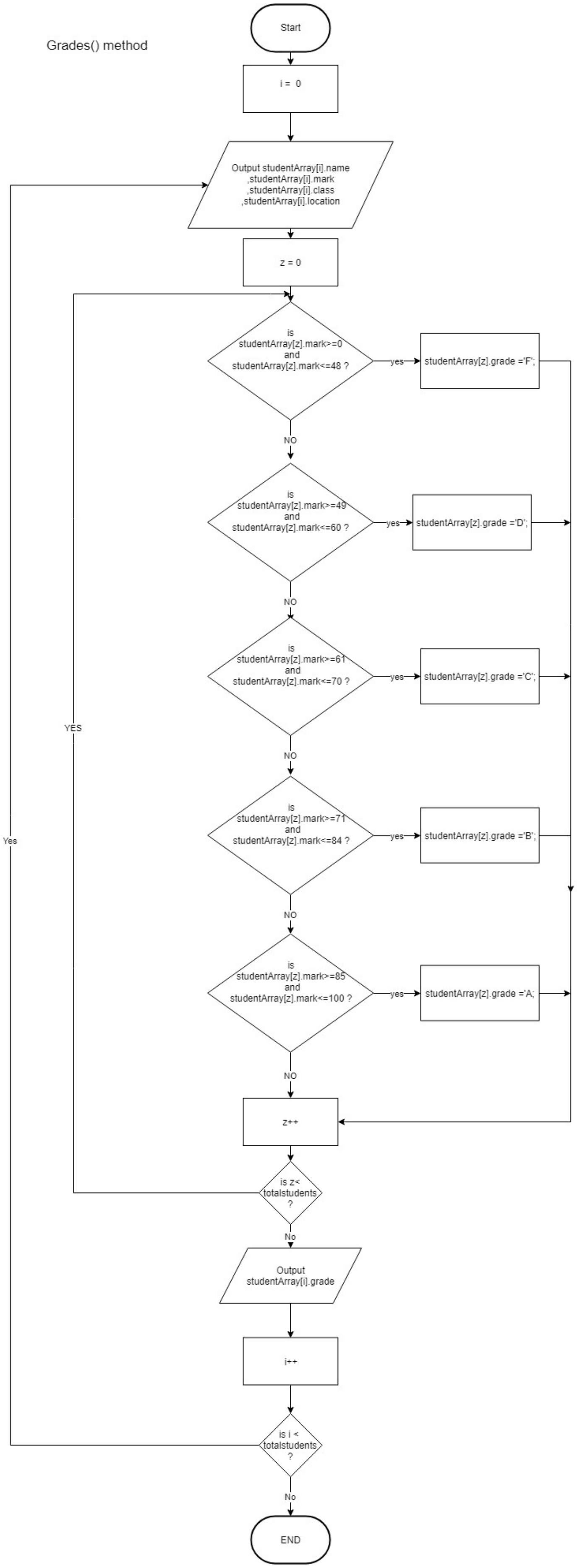

MainMenu
Start
choice = 0
Classroom myClass = new Classroom();
M
1. Input Student Details
2. Grades
3.Average and Sum
4. Max and Min
5. Above or Below Average
6.Search
7.Exit
Is Option 1 ?
Yes
Input()
No
Is Option 2 ?
Yes
Grades()
No
Is Option 3 ?
Yes
AverageSum()
No
Is Option 4 ?
Yes
MaxMin()
No
Is Option 5 ?
Yes
ABAverage()
No
Is Option 6 ?
Yes
Search()
No
Is Option 7 ?
No
Display Invalid
Yes
Thank you for using the program
END

Class Marks

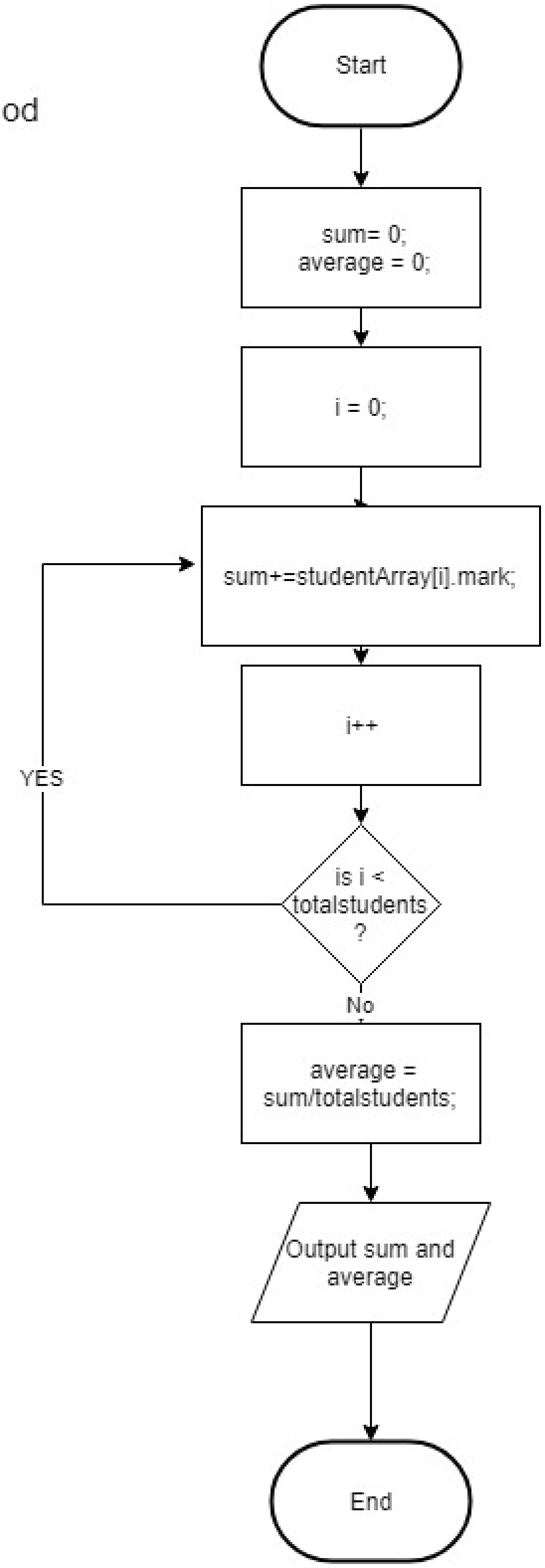
Input() method


Start
Student [] = studentArray;
totalstudents=0;
name, Location, Class;
mark;
grade;
Input totalstudents
studentArray = new Student [totalstudents];
i = 0
Input studentArray[i].name
Input studentArray[i].mark
is mark < 0 or >100
Yes
Invalid
No
Input studentArray[i].class ,
Input studentArray[i].location
Student myStudent = new Student (name, Class, Location,mark,grade);
studentArray[i]= myStudent;
i++
is i < totalstudents
Yes
NO
i=0
Output studentArray[i].name ,studentArray[i].mark ,studentArray[i].class ,studentArray[i].location
i++
is i < totalstudents ?
Yes
No
End

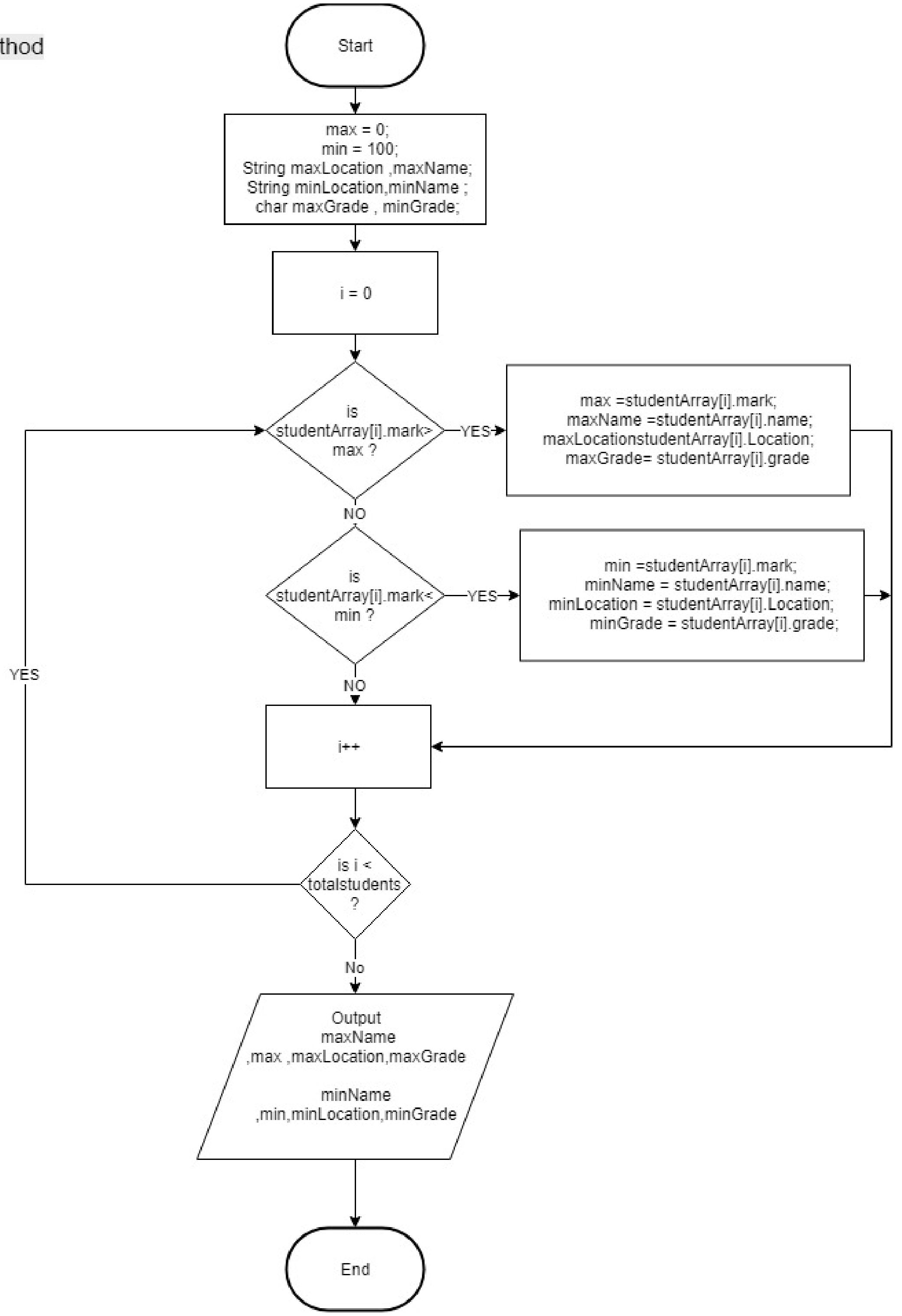
Class Marks

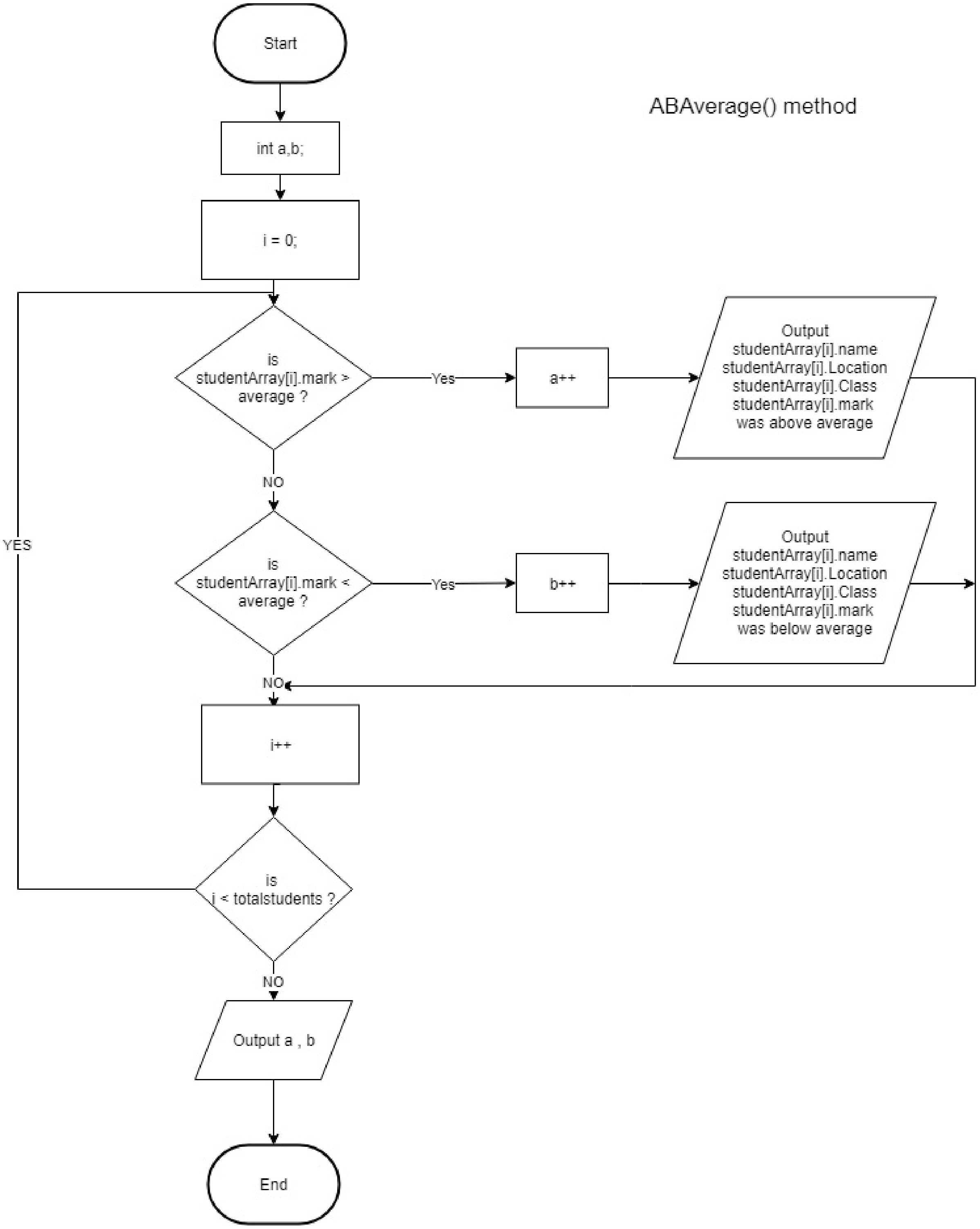

Grades() method
Start
i = 0
Output studentArray[i].name ,studentArray[i].mark ,studentArray[i].class ,studentArray[i].location
z = 0
is studentArray[z].mark>=0 and studentArray[z].mark<=48 ?
yes
studentArray[z].grade ='F';
NO
is studentArray[z].mark>=49 and studentArray[z].mark<=60 ?
yes
studentArray[z].grade ='D';
NO
is studentArray[z].mark>=61 and studentArray[z].mark<=70 ?
yes
studentArray[z].grade ='C';
NO
is studentArray[z].mark>=71 and studentArray[z].mark<=84 ?
yes
studentArray[z].grade ='B';
NO
is studentArray[z].mark>=85 and studentArray[z].mark<=100 ?
yes
studentArray[z].grade ='A;
NO
z++
is z< totalstudents ?
YES
No
Output studentArray[i].grade
i++
is i < totalstudents ?
Yes
No
END

AverageSum() method

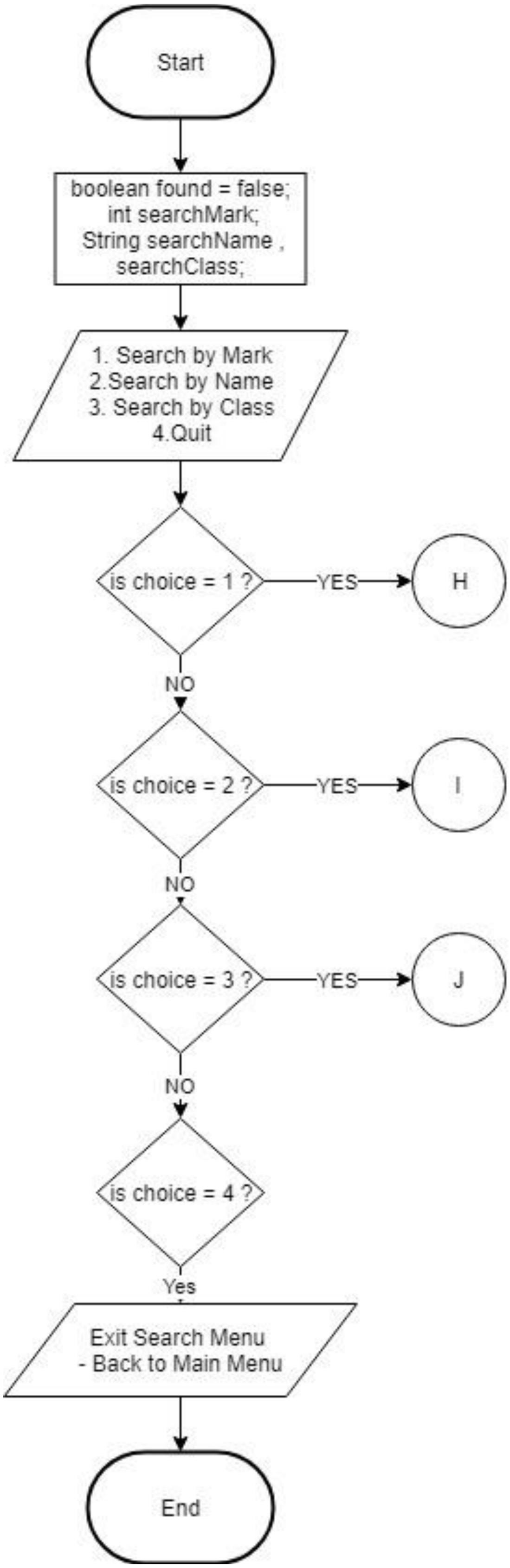


MaxMin() method

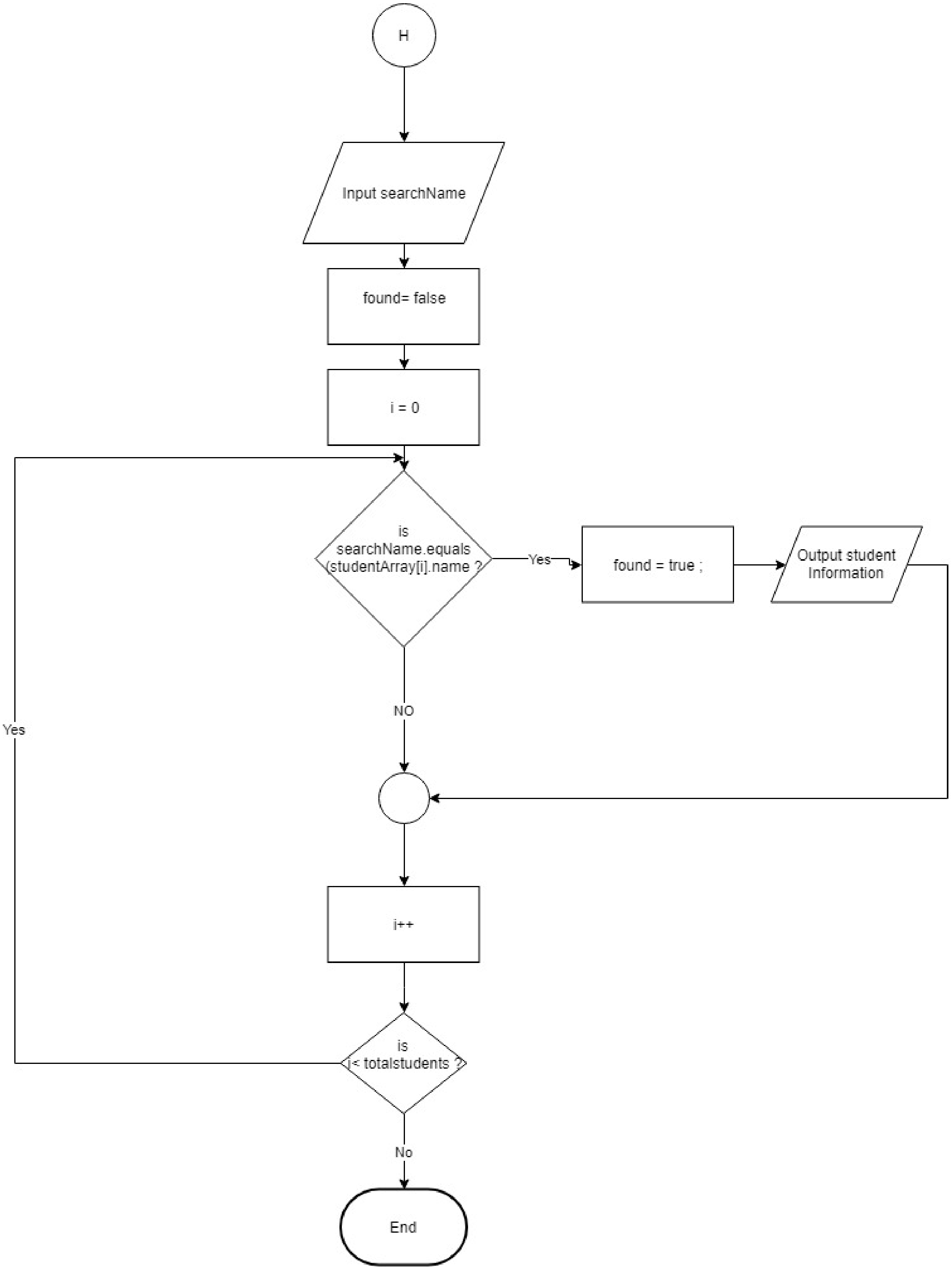

Start
max = 0;
min = 100;
String maxLocation ,maxName;
String minLocation,minName ;
char maxGrade , minGrade;
i = 0
is studentArray[i].mark> max ?
YES
max =studentArray[i].mark;
maxName =studentArray[i].name;
maxLocationstudentArray[i].Location;
maxGrade= studentArray[i].grade
NO
is studentArray[i].mark< min ?
YES
min =studentArray[i].mark;
minName = studentArray[i].name;
minLocation = studentArray[i].Location;
minGrade = studentArray[i].grade;
NO
YES
i++
is i < totalstudents ?
No
Output maxName ,max ,maxLocation,maxGrade
minName ,min,minLocation,minGrade
End

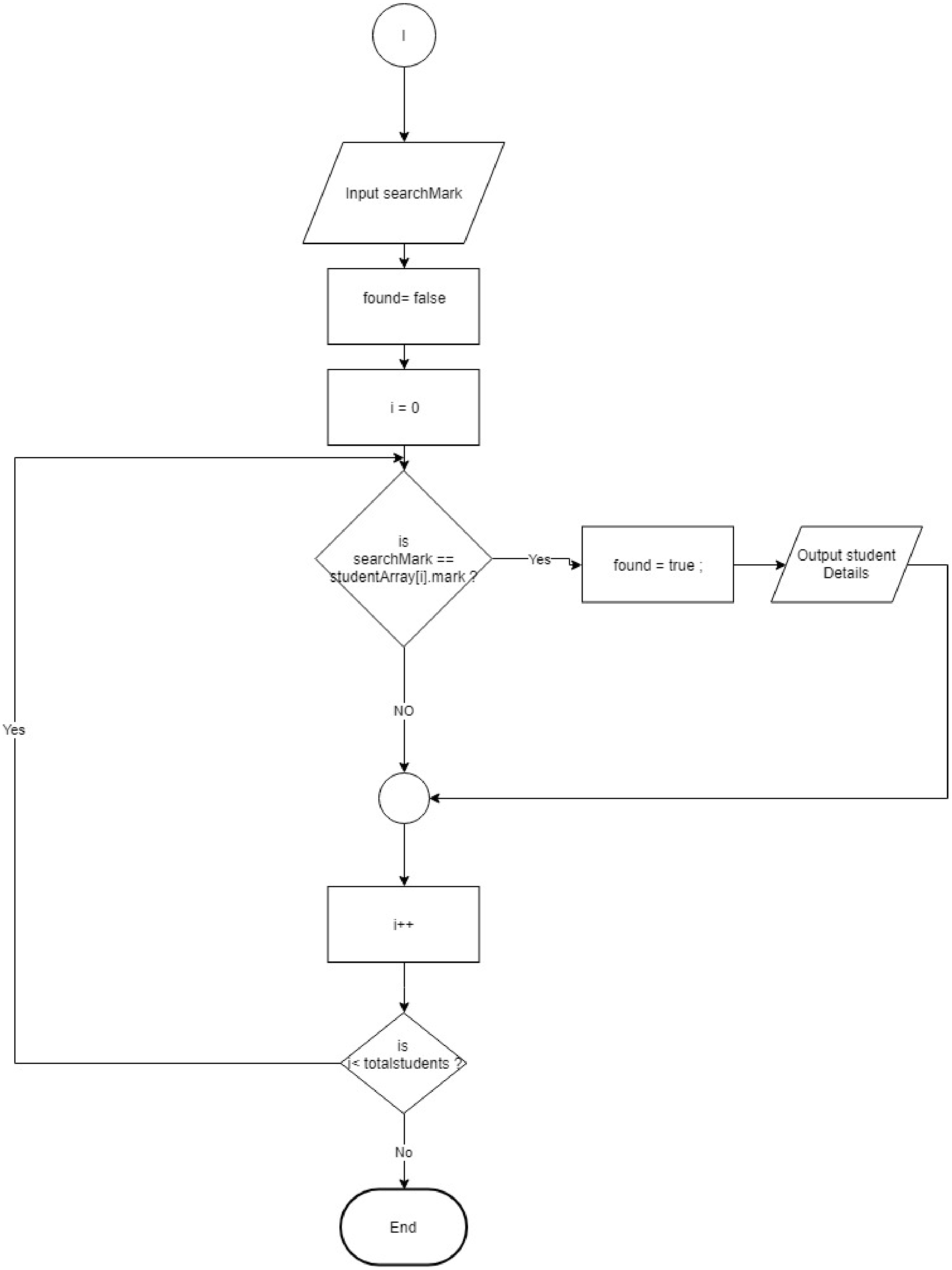

Start
int a,b;
i = 0;
is studentArray[i].mark > average ?
Yes
a++
Output studentArray[i].name studentArray[i].Location studentArray[i].Class studentArray[i].mark was above average
NO
YES
is studentArray[i].mark < average ?
Yes
b++
Output studentArray[i].name studentArray[i].Location studentArray[i].Class studentArray[i].mark was below average
NO
i++
is i < totalstudents ?
NO
Output a , b
End

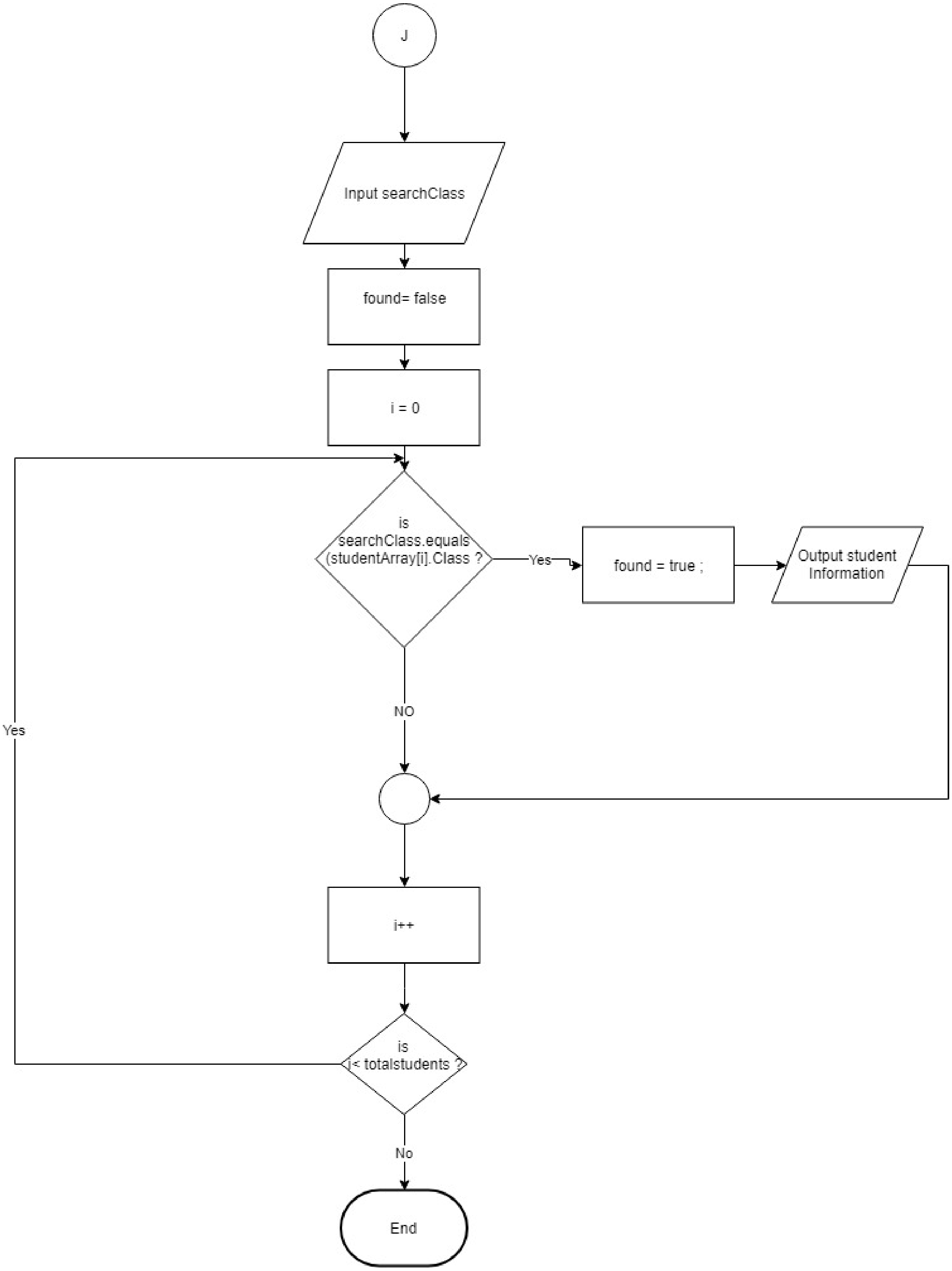
ABAverage() method



search() method

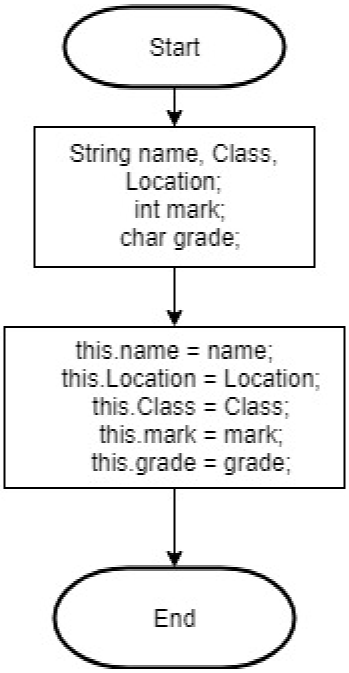

H
Input searchName
found= false
i = 0
is
searchName.equals
(studentArray[i].name ?
Yes
found = true ;
Output student
Information
NO
i++
is
i< totalstudents ?
Yes
No
End


I
Input searchMark
found= false
i = 0
is
searchMark ==
studentArray[i].mark ?
Yes
found = true ;
Output student
Details
NO
i++
is
i< totalstudents ?
Yes
No
End


J
Input searchClass
found= false
i = 0
is searchClass.equals (studentArray[i].Class ?
Yes
found = true ;
Output student Information
NO
i++
is i< totalstudents ?
Yes
No
End

Object Class

Student(name, Class, Location, mark, grade)



```
String name, Class,
Location;
int mark;
char grade;
```

```
this.name = name;
this.Location = Location;
this.Class = Class;
this.mark = mark;
this.grade = grade;
```

Computer listing of program

Main Class

```
import javax.swing.JOptionPane;
public class MainMenuClass {
  public static void main (String args[]){
    int choice = 0;
   //Creating an object of type Classroom//
   Classroom myClass = new Classroom();
  //The do while loop is being used to show the main menu options until the user quits//
   do{
    choice = Integer.parseInt(JOptionPane.showInputDialog("1. Input Student Details \n 2.  Grades \n 3.
Average and Sum \n "+
          "4.  Max and Min  \n 5.  Above or Below Average \n 6.  Search  \n 7.  Exit"));

     //The switch statement is being used to show the main menu according the user’s choice//
     switch (choice) {
       //These methhods are calling code from the Classroom Class//
       case 1 : myClass.Input();
           break;
       case 2: myClass.Grades();
           break;
       case 3 :myClass.AverageSum();
           break;
       case 4 :myClass.MaxMin();
           break;
       case 5 :myClass.ABAverage();
           break;
       case 6 :myClass.Search();
           break;
           //The user has terminated the program//
       case 7 :JOptionPane.showMessageDialog(null,"Thank you , for using the program");
           break;
           //Default is being used as an invalid option //
       default :JOptionPane.showMessageDialog(null,"Invalid option , please try again");
      }
     }while(choice!=7);
  }
```

Student Object Class

```
public class Student{
  //Creating variables of object//
  String name, Class, Location;
  int mark;
  char grade;

  //Constructor

  Student(String name, String Class, String Location, int mark, char grade){
     this.name = name;
     this.Location = Location;
     this.Class = Class;
     this.mark = mark;
     this.grade = grade;
    }
}
```

Classroom Object Class

```
import javax.swing.JOptionPane;
public class Classroom{
  //Creation of student array//
  Student[] studentArray;
 //Initial Variables //
 int totalstudents=0;
 String name, Location, Class;
 int mark;
 char grade;
  //Variables for option 3//
 int sum= 0;
 double average;
 //Variables for option 4//
 int max = 0;
 int min = 100;
 String maxLocation ,maxName;
 String minLocation,minName ;
 char maxGrade , minGrade;
 //Variables for option 5//
 int a,b;
 //Variables for option 6//
 int choice;
 boolean found = false;
 int searchMark;
 String searchName , searchClass;

 //This method is being used , as a user input //
 void Input(){
     //The user is requesting to enter the number of students in the class//
   String a1= JOptionPane.showInputDialog(null,"Please enter number of students" );
   totalstudents=Integer.parseInt(a1);
   //The student array is being created according to the value stored in the totalstudents variable//
   studentArray = new Student [totalstudents];
   //The for loop is being used to enter details in the student array//
   for (int i = 0 ; i< totalstudents ; i++){
     name= JOptionPane.showInputDialog(null,"Enter Student name " +(i+1));
     String a2= JOptionPane.showInputDialog(null,"Enter Mark  " +(i+1));
     mark =Integer.parseInt(a2);
    //The while loop is being used to check whether the  mark is invalid or correct//
     while((mark<0 )||(mark>100)){
```

```
   if ((mark<0 )||(mark>100))
     JOptionPane.showMessageDialog(null,"Invalid mark , Re enter ");
   //The user is re-entering the mark//
   mark =Integer.parseInt(a2);
  }
   Class = JOptionPane.showInputDialog(null,"Enter Class   " +(i+1));
   Location =JOptionPane.showInputDialog(null, "Enter Location   " +(i+1));
   //An object is being created to include the constructor//
   Student myStudent = new Student (name, Class, Location,mark,grade );
   //The object array' studentArray[] ' is being populated by the constructor //
   studentArray[i]= myStudent;
 }
 //Outputting //
 //The for loop is being used to output the Student Details //
 for (int i= 0; i< totalstudents; i++){
          JOptionPane.showMessageDialog(null,"Name  :" + studentArray[i].name +"\nMark  :" +
         studentArray[i].mark  +"\nClass  :"     + studentArray[i].Class + "\nLocation  :" +
         studentArray[i].Location);

     }
    }
  //This method is being used to show information of students and output the grade obtained by each student//
  void Grades() {
    //This for loop is being used to output the student Details//
    for (int i= 0; i< totalstudents; i++){
         //This for loop is being used to find the student ' s grade //
        //The parameters determine the Student's grade //
     for (int z = 0; z<totalstudents;z++){
           if((studentArray[z].mark>=0)&&(studentArray[z].mark<=48))
             studentArray[z].grade ='F';
           else if((studentArray[z].mark>=49)&&(studentArray[z].mark<=60))
             studentArray[z].grade ='D';
           else if((studentArray[z].mark>=61)&&(studentArray[z].mark<=70))
             studentArray[z].grade ='C';
           else if((studentArray[z].mark>=71)&&(studentArray[z].mark<=84))
             studentArray[z].grade='B';
           else if((studentArray[z].mark>=85)&&(studentArray[z].mark<=100))
             studentArray[z].grade ='A';
          }
    //The grade depending on the parameters is being outputted with the student details//
```

```
            JOptionPane.showMessageDialog(null,"Name  :" + studentArray[i].name  +"\nMark  :" +
           studentArray[i].mark  +"\nClass  :" + studentArray[i].Class + "\nLocation  :" +
           studentArray[i].Location + "\nGrade : " +  studentArray[i].grade );
      }
    }
    //This method is used to calculate the total mark of all students and to find the average //
    void AverageSum() {
      //This for loop is being used to add all the marks to the variable sum //
      for(int i= 0 ; i< totalstudents; i++){
       sum+=studentArray[i].mark;
          }
       //The sum is then divided by the number of students to find the mean //
       average = sum/(double)(totalstudents);
      JOptionPane.showMessageDialog(null,"The sum is " + sum+ "\n The average is  " + average);
    }
    //This method is being used to find the Maximum and Minimum mark//
    void MaxMin() {
     //This for loop is being used to check all the marks for the maximum and minimum //
     for ( int i= 0; i < totalstudents ; i++){
                //T if statement is being used to check whether the student mark is greater than the previous
student mark //
                if(studentArray[i].mark> max){
                 //If the student mark is greater , the max variable will be populated by the current student
mark//
                 max =studentArray[i].mark;
                 //These variables are used to output student details//
                 maxName = studentArray[i].name;
                 maxLocation = studentArray[i].Location;
                 maxGrade= studentArray[i].grade;
                }
                //The if statement is being used to check whether the student mark is smaller than the previous
student mark //
                if(studentArray[i].mark< min){
                 //If the student mark is smaller , the min variable will be populated by the current student
mark//
                 min =studentArray[i].mark;
                 //These variables are used to output student details//
                 minName = studentArray[i].name;
                 minLocation = studentArray[i].Location;
                 minGrade = studentArray[i].grade;
                }
               }
      //The student details of the highest and lowest mark are being outputted //
       JOptionPane.showMessageDialog(null, maxName + " from   " + maxLocation + "  got the highest mark
of  "  + max + "  and a grade  " + maxGrade +"\n" +
                                 minName + " from   " + minLocation + "  got the lowest mark of  "  + min + "  and
a grade  " + minGrade );
   }
   //This method is being used to check which students were above or below average , and how many
there were //
   void ABAverage (){
     //This for loop is being used to check all the students //
     for ( int i= 0; i <totalstudents;i++){
      //The if statement is being used to check whether the mark is above average //
      if (studentArray[i].mark > average ){
        //This counter is being used to find how many students were above average //
        a++;
        //Student Details are being outputted and also  above average //
             JOptionPane.showMessageDialog(null,studentArray[i].name + " from " +
            studentArray[i].Location + " in  class " + studentArray[i].Class + "  got " + studentArray[i].mark +
        "  marks and the grade " + studentArray[i].grade + " was above total average ");
              }
      //The if statement is being used to check whether the mark is below average //
      if (studentArray[i].mark < average ){
        //This counter is being used to find how many students were below average //
        b++;
        //Student Details are being outputted and also  below average //
              JOptionPane.showMessageDialog(null,studentArray[i].name + " from " +
            studentArray[i].Location + " in  class " + studentArray[i].Class + "  got "
        + studentArray[i].mark + "  marks and the grade " + studentArray[i].grade + " was below total average
");
              }
             }
             //Outputting of how many students were above or below average //
             JOptionPane.showMessageDialog(null,"There were " + a + " students above total average \n "+
                                  "There were " + b + " students below total average ");
   }
   //This method is being used to search //
    void Search(){
     //The do while loop is being used to execute a menu //
     do {
      choice = Integer.parseInt(JOptionPane.showInputDialog("Choose between the following options :
\n1. Search by mark\n2. Search by name \n3. Search by class\n4. Quit"));
       //The switch statement is being used to execute all  options //
```

```
switch(choice){

   case 1 : JOptionPane.showMessageDialog(null," 1. Search by mark ");
          //The user is being asked to enter mark//
         String a4 = JOptionPane.showInputDialog(null,"Input mark you want to search ");
         searchMark = Integer.parseInt(a4);
         //found is being populated as false to avoid it from interfering in the for loop//
         found = false;
         //The for loop is being used to check all the students//
         for (int i = 0 ; i < totalstudents ; i++){
          //The if statement is checking whether the searchMark is equal to the current mark //
           if (searchMark == studentArray[i].mark) {
             //The mark requested by the user has been found and the students details are shown //
             found = true ;
             JOptionPane.showMessageDialog(null,"This/ese student/s obtained the following mark ");
              JOptionPane.showMessageDialog(null,studentArray[i].name + " from " +
              studentArray[i].Location + " in  class " + studentArray[i].Class + "  got  this mark" );
          }
          }
          //The if statement is showing a message to the user , that the mark wasn't found //
           if(found = false)
              JOptionPane.showMessageDialog(null,"Not found");
          break;

   case 2 :JOptionPane.showMessageDialog(null," 2. Search by name ");
          //The user is being asked to enter name of student//

         searchName = JOptionPane.showInputDialog(null,"Input name you want to search ");
         //found is being populated as false to avoid it from interfering in the for loop//
         found = false;
          //The for loop is being used to check all the students//
         for (int i = 0 ; i < totalstudents ; i++){
          //The if statement is checking whether the searchName is equal to the current name //
           if (searchName.equals(studentArray[i].name)) {
             //The name requested by the user has been found and the students details are shown //
             found = true ;
             JOptionPane.showMessageDialog(null,"Student information :");
                 JOptionPane.showMessageDialog(null,studentArray[i].name + " from " +
               studentArray[i].Location + " in  class " + studentArray[i].Class + "  got "
             + studentArray[i].mark + "  marks and the grade " + studentArray[i].grade );
          }
          }
```

```
                 //The if statement is showing a message to the user , that the name wasn't found //
                 if(found = false)
                   JOptionPane.showMessageDialog(null,"Not found");
                break;

         case 3 :JOptionPane.showMessageDialog(null," 3. Search by class ");
                //The user is being asked to enter the class //

                searchClass =JOptionPane.showInputDialog(null,"Input class you want to search ");
                //found is being populated as false to avoid it from interfering in the for loop//
                found = false;
                //The for loop is being used to check all the students//
                for (int i = 0 ; i < totalstudents ; i++){
                //The if statement is checking whether the searchClass is equal to the current Class //
                 if (searchClass.equals(studentArray[i].Class)) {
                  //The Class requested by the user has been found and the students details are shown //
                  found = true ;
                  JOptionPane.showMessageDialog(null,"Student information  according to class :");
                   JOptionPane.showMessageDialog(null,studentArray[i].name     +     "         from     "     +
                   studentArray[i].Location +  "  got " + studentArray[i].mark + "  marks and the grade "
                  + studentArray[i].grade );
                }
                }
                //The if statement is showing a message to the user , that the Class wasn't found //
                 if(found = false)
                   JOptionPane.showMessageDialog(null,"Not found");
                break;
                //The user has  terminated this menu back to main menu //
         case 4 : JOptionPane.showMessageDialog(null," Exit Search Menu - Back to Main Menu ");
                break;
                //Default is being used as an Invalid option  //
         default :JOptionPane.showMessageDialog(null,"Invalid Option ");
     }
    }
     while(choice!=4);
   }
}
```

Details of any special design features

Arrays

Arrays are a collection of data items of the same data type. I used an **Object array** to save the inputted data in them. Object arrays are a collection of objects together. In the object array the following inputs are saved: name, Class, Location, mark ,grade. Refer to the following snippet.

```
   //An object is being created to include the constructor//
   Student myStudent = new Student (name, Class, Location,mark,grade );
   //The object array’ studentArray[] ' is being populated by the constructor //
   studentArray[i]= myStudent;
```

Switch statement

The **Switch Statement** is a multiway branch statement. The Switch Statement is used to execute a Menu . The Menu is a list of commands or facilities displayed on screen, whereby the user is requested to input the options he wants. I used the switch statement in both the Main Menu options and for the search menu options. Refer to the following snippet.

```
   do{
     choice = Integer.parseInt(JOptionPane.showInputDialog("1. Input Student Details \n 2.  Grades \n 3.
Average and Sum \n "+
               "4.  Max and Min  \n 5.  Above or Below Average \n 6.  Search  \n 7.  Exit"));

      //The switch statement is being used to show the main menu according the users’ choice//
      switch (choice) {
         //These methods are calling code from the Classroom Class//
         case 1 : myClass.Input();
               break;
         case 2: myClass.Grades();
               break;
         case 3 :myClass.AverageSum();
               break;
         case 4 :myClass.MaxMin();
               break;
         case 5 :myClass.ABAverage();
               break;
         case 6 :myClass.Search();
                break;
                //The user has terminated the program//
         case 7 :JOptionPane.showMessageDialog(null,"Thank you , for using the program");
                break;
                //Default is being used as an invalid option //
         default :JOptionPane.showMessageDialog(null,"Invalid option , please try again");
        }
      }while(choice!=7);
  }
```

Loops

The **For loop** is a loop which repeats the block of statements within it for a specific number of times. I used the for loop to let the user input the student information and then outputting it back, since arrays are a collection of the same data , a for loop is very beneficial in this area. Refer to the following snippet.

```
  for (int i= 0; i< totalstudents; i++){
 JOptionPane.showMessageDialog(null,"Name    :"    +    studentArray[i].name    +"\nMark    :"    +
studentArray[i].mark +"\nClass :"   + studentArray[i].Class + "\nLocation :" + studentArray[i].Location);
      }
```

The **Do while loop** is a loop which checks the condition at the end of the block statement .I used the Do while loop to execute the both in the Main Menu and in the Search Menu. The Do while loop is executed at least once. . Refer to the following snippet.

```
    do{
      choice = Integer.parseInt(JOptionPane.showInputDialog("1. Input Student Details \n 2.  Grades \n 3.
Average and Sum \n "+
                "4.  Max and Min  \n 5.  Above or Below Average \n 6.  Search  \n 7.  Exit"));

       //The switch statement is being used to show the main menu according the users' choice//
       switch (choice) {
          //These methods are calling code from the Classroom Class//
          case 1 : myClass.Input();
                break;
          case 2: myClass.Grades();
                break;
          case 3 :myClass.AverageSum();
```

```
            break;
        case 4 :myClass.MaxMin();
            break;
        case 5 :myClass.ABAverage();
            break;
        case 6 :myClass.Search();
            break;
            //The user has terminated the program//
        case 7 :JOptionPane.showMessageDialog(null,"Thank you , for using the program");
            break;
            //Default is being used as an invalid option //
        default :JOptionPane.showMessageDialog(null,"Invalid option , please try again");
        }
      }while(choice!=7);
  }
```

The **While loop** is a loop which check the condition at the beginning .I used the while loop to check whether the mark entered by the user was within the correct parameters. If the while loop is not executed it will become an infinite loop. Refer to the following snippet.

```
  while((mark<0 )||(mark>100)){
  if ((mark<0 )||(mark>100))
    JOptionPane.showMessageDialog(null,"Invalid mark , Re-enter ");
  //The user is re-entering the mark//
  mark =Integer.parseInt(a2);
  }
```

Objects and methods

An **Object** is an instance of a class. It has instance methods and instance variables .I used void methods to populate the Menu options with functions . In the class-based object-oriented programming paradigm, object refers to a particular instance of a class, where the object can be a combination of variables, functions, and data structures. Refer to the following snippets.

```
          case 1 : myClass.Input();
               break;
         case 2: myClass.Grades();
               break;
         case 3 :myClass.AverageSum();
```

```
public class Student{
 //Creating variables of object//
 String name, Class, Location;
 int mark;
 char grade;
```

A **Method** is a function which does a particular job when called. I used the void methods in order to provide the functions which were needed from the object class, to provide as Menu options. Refer to the following snippet.

```
void AverageSum() {
     //This for loop is being used to add all the marks to the variable sum //
     for(int i= 0 ; i< totalstudents; i++){
      sum+=studentArray[i].mark;
        }
```

A **Constructor** is a class-based object-oriented programming. It is a special type of subroutine called to create an object. It prepares the new object for use, often accepting arguments that the constructor uses to set required member variables. I used the constructor to create the object array. Refer to the following snippet.

```
Student(String name, String Class, String Location, int mark, char grade){
   this.name = name;
   this.Location = Location;
   this.Class = Class;
   this.mark = mark;
   this.grade = grade;
  }
```

Graphics User Interface

It is a user interface that includes graphical elements, such as windows, icons and buttons .I used GUI to input and output information. Refer to the following snippets.

```
   JOptionPane.showMessageDialog(null,"Student information  according to class :");
JOptionPane.showMessageDialog(null,studentArray[i].name + "  from " + studentArray[i].Location +  "  got
" + studentArray[i].mark + "  marks and the grade "
                        + studentArray[i].grade );
```

```
String a1= JOptionPane.showInputDialog(null,"Please enter number of students" );
   totalstudents=Integer.parseInt(a1);
```

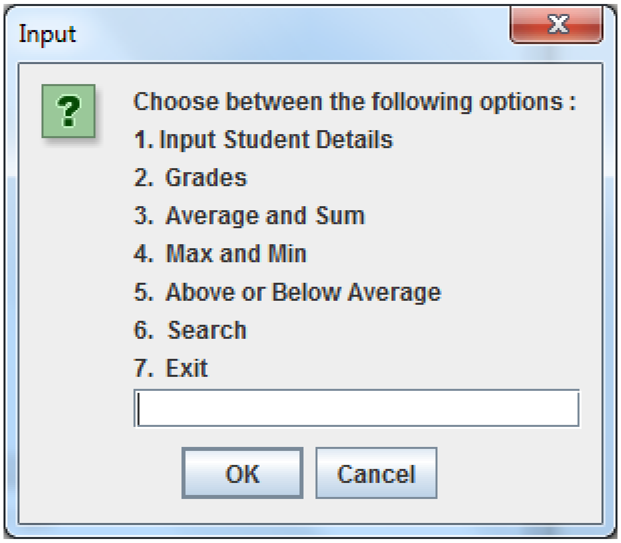
Running the program

Evidence the solution works

The following table was used to test the program

Number	Name	Class	Location	Mark
1	Joe	11B	Zebbug	90
2	Adam	11W	Marsa	45
3	Barry	11A	England	64
4	Jonothan	11C	Attard	98
5	Matthew	11W	Siggiewi	32
6	Luke	11S	Bugibba	78
7	Josh	11S	Burmarrad	73
8	Liam	11A	Qormi	40
9	Sean	11Q	Valletta	83
10	Maria	11B	Zebbug	92
11	Ryan	11B	Zabbar	85
12	Ben	11Q	Qawra	75
13	Jimmy	11A	San Pawl	89
14	Samuel	11S	Mosta	22
15	Miguel	11S	Naxxar	57

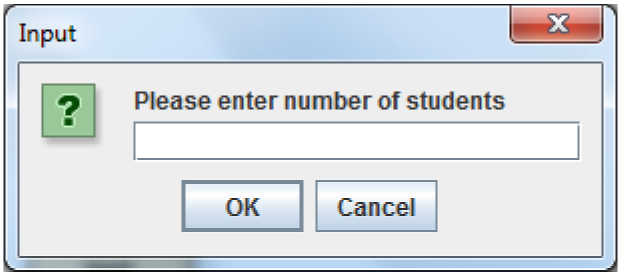
Main Menu



Choose between the following options :
1. Input Student Details
2. Grades
3. Average and Sum
4. Max and Min
5. Above or Below Average
6. Search
7. Exit

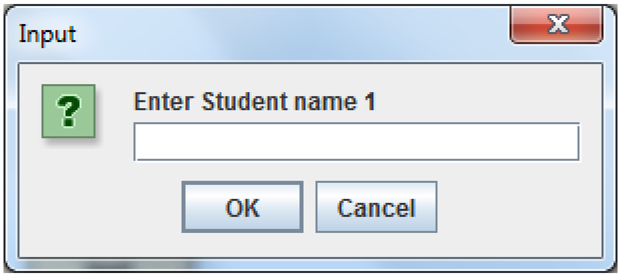
Figure 1 The Main Menu is being shown

Option 1 : Input



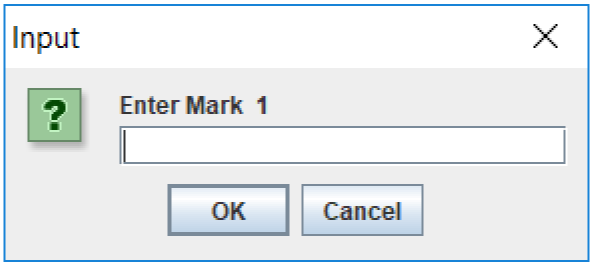
Please enter number of students

Figure 2 The user is being asked to enter the number of students



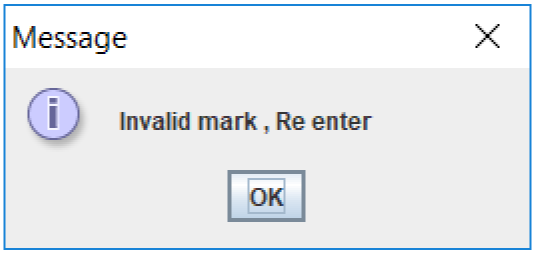
Enter Student name 1

Figure 3 The user is being asked to enter the student name



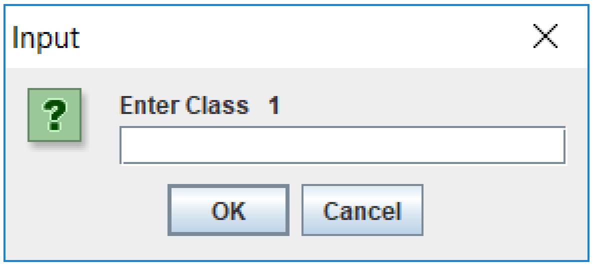
Enter Mark 1

Figure 4 The user is being asked to enter the student mark



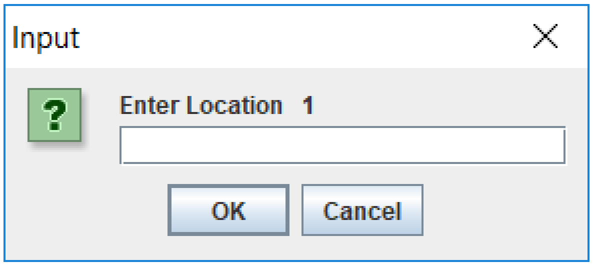
Invalid mark , Re enter

Figure 5 An Error message is shown whether the mark is not in the required parameters



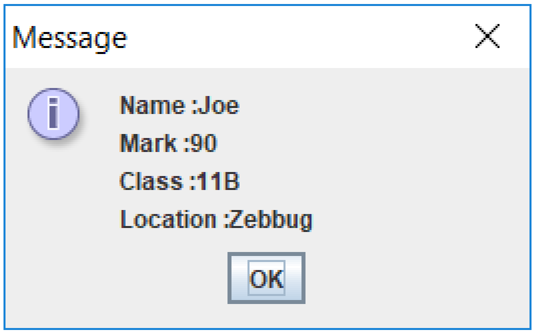
Enter Class 1

Figure 6 The user is being asked to enter the student class



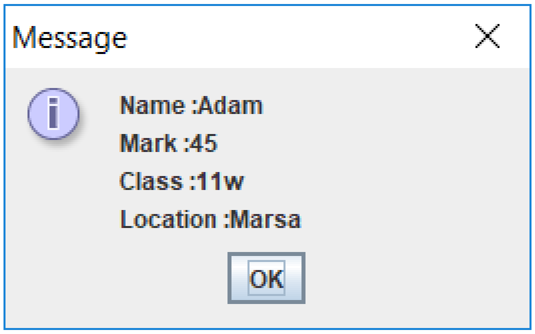
Enter Location 1

Figure 7 The user is being asked to enter the student location



Name :Joe
Mark :90
Class :11B
Location :Zebbug

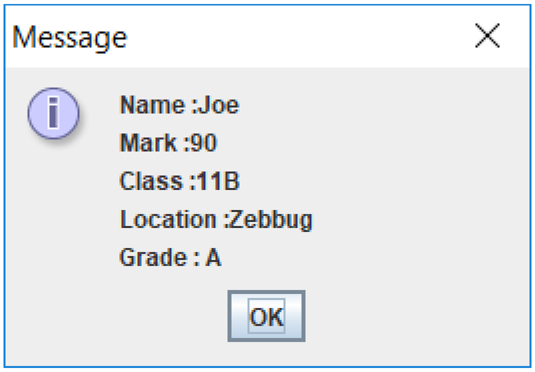
Figure 8 The user is being shown the details of student 1



Name :Adam
Mark :45
Class :11w
Location :Marsa

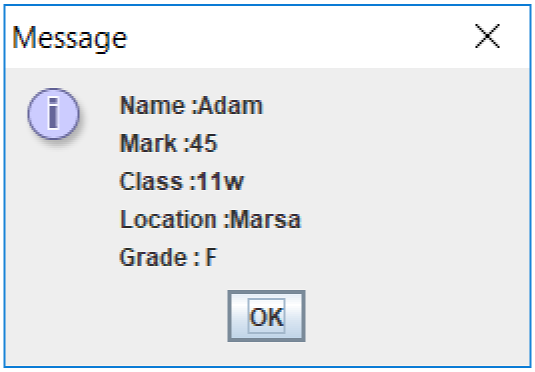
Figure 9 The user is being shown the details of student 2

Option 2 : Grades



Name :Joe
Mark :90
Class :11B
Location :Zebbug
Grade : A

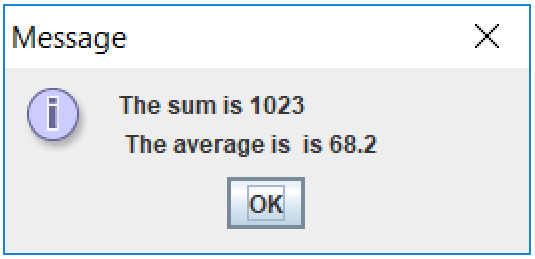
Figure 10 The user is being shown the details of student 1 and the grade



```
Name :Adam
Mark :45
Class :11w
Location :Marsa
Grade : F
```

Figure 11 The user is being shown the details of student 2 and the grade

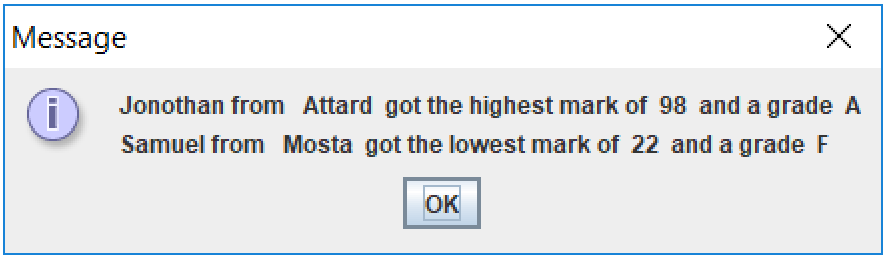
Option 3 : Sum and Average



```
The sum is 1023
 The average is  is 68.2
```

Figure 12 The Sum and Average is being displayed

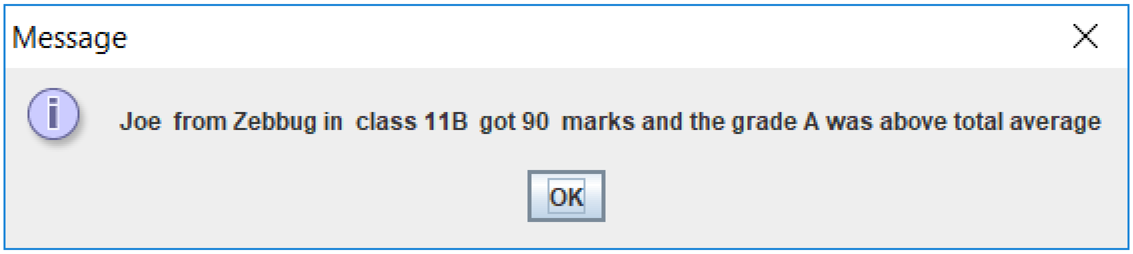
Option 4 : Maximum and Minimum Marks



```
Jonothan from   Attard  got the highest mark of  98  and a grade  A
Samuel from   Mosta  got the lowest mark of  22  and a grade  F
```

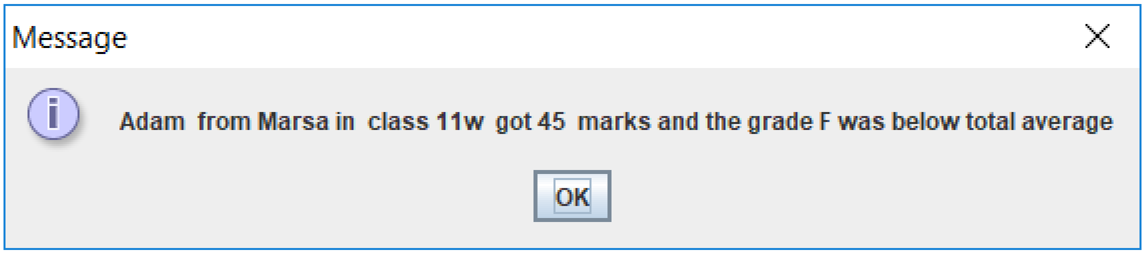
Figure 13 The Maximum and Minimum marks are being shown

Option 5 : Above or Below Average



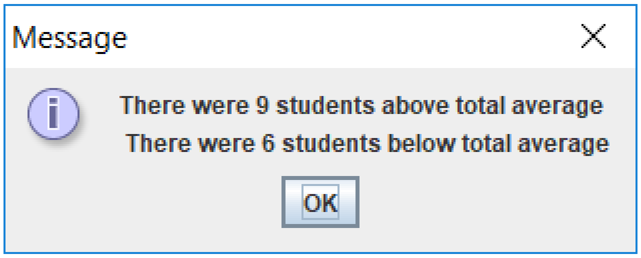
Joe from Zebbug in class 11B got 90 marks and the grade A was above total average

Figure 14 The student details of Joe and whether he was above or below average is being shown



Adam from Marsa in class 11w got 45 marks and the grade F was below total average

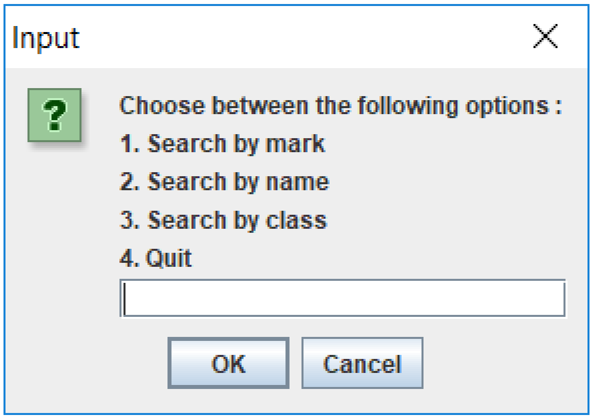
Figure 15 The student details of Adam and whether he was above or below average is being shown



There were 9 students above total average
There were 6 students below total average

Figure 16 The number of students which were above or below average is being shown

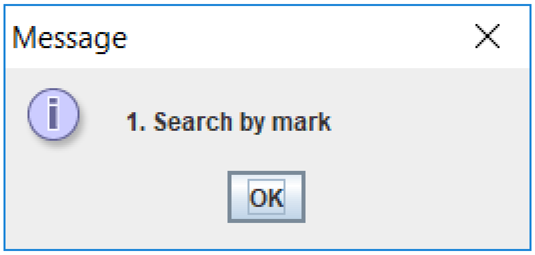
Option 6 : Search Menu



Choose between the following options :
1. Search by mark
2. Search by name
3. Search by class
4. Quit

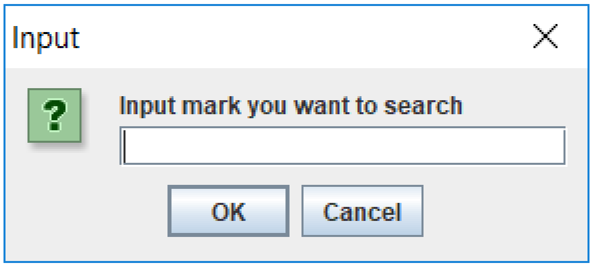
Figure 17 The Search Menu is being displayed to the user

Search Menu Option 1 : Search by Mark



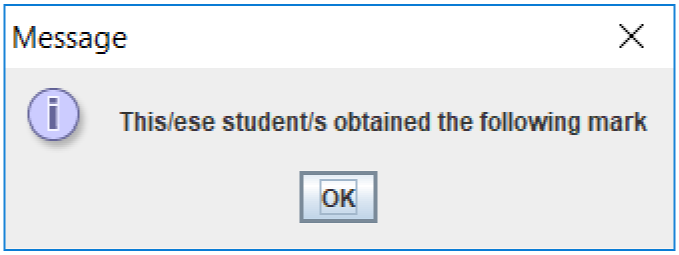
1. Search by mark

Figure 18 The option is being shown to the user



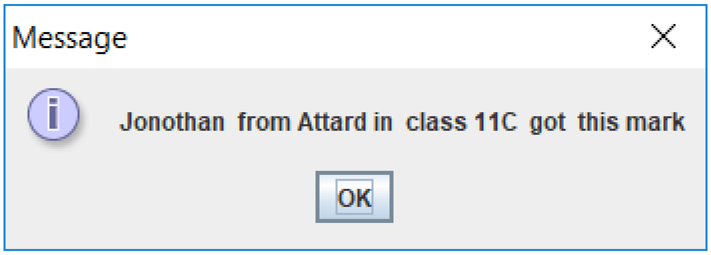
Input mark you want to search

Figure 19 The user is required to enter the mark to search for



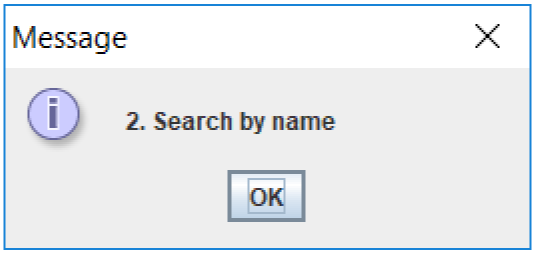
This/ese student/s obtained the following mark

Figure 20 The list of students who obtained the mark is being shown



Jonothan from Attard in class 11C got this mark

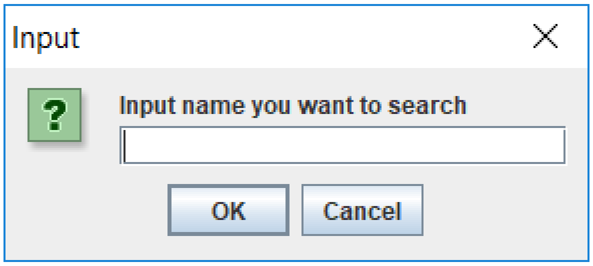
Search Menu Option 2 : Search by Name



Message

2. Search by name

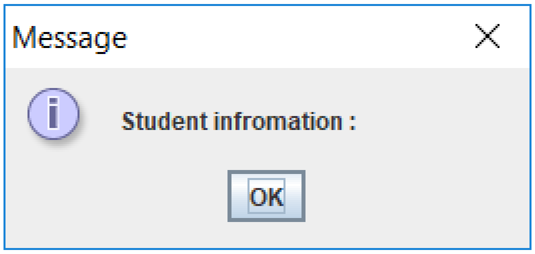
Figure 21 The option is being shown to the user



Input

Input name you want to search

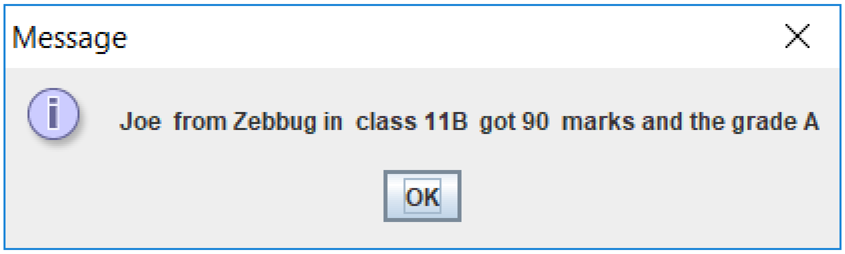
Figure 22 The user is required to enter the name to search for



Message

Student infromation :

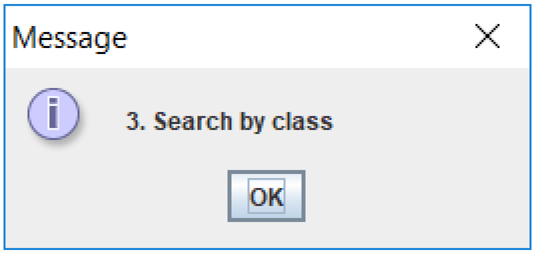
Figure 23 The list of students who have that name are being shown



Message

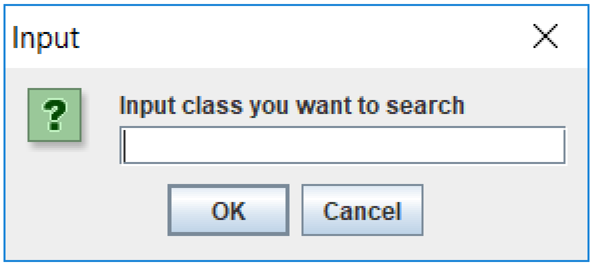
Joe from Zebbug in class 11B got 90 marks and the grade A

Search Menu Option 3 : Search by Class



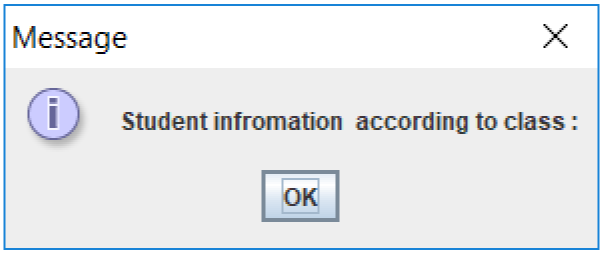
3. Search by class

Figure 24 The option is being shown to the user



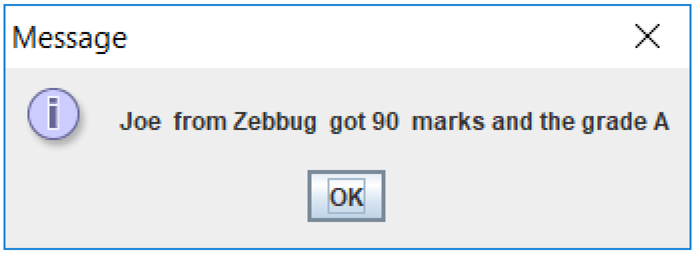
Input class you want to search

Figure 25 The user is required to enter the class to search for

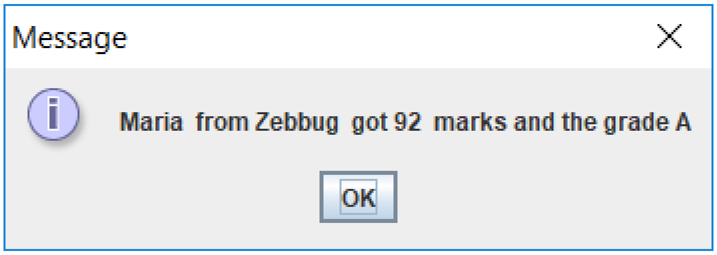


Student infromation according to class :

Figure 26 The list of students who have are in that class are being shown

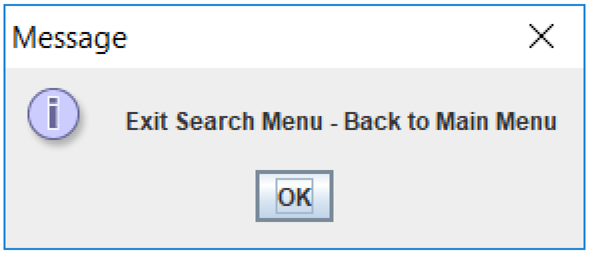


Joe from Zebbug got 90 marks and the grade A



Maria from Zebbug got 92 marks and the grade A

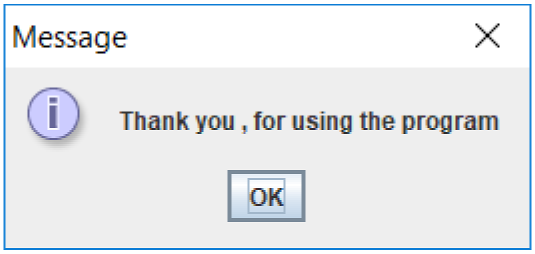
Search Menu Option 4 : Quit



Exit Search Menu - Back to Main Menu

Figure 27 The user is exiting the Search Menu and he is going back to the Main Menu

Option 7 : Quit from Main Menu



Thank you , for using the program

Figure 28 The user is exiting the Main Menu

Details of test data

Checking the Main Menu

Test data	Purpose of test	Expected result	Actual Result
1	To check that data at the boundaries is accepted	Accepted	Accepted
7			
3	To check that data within the boundaries is accepted.	Accepted	Accepted
2			
8	To check that data outside the boundaries is not accepted	Not Accepted	Not Accepted
-6			

To test the program the following data was used

The user can enter as much Inputs he wants, suppose that 15 users have been entered.

Number	Name	Class	Location	Mark
1	Joe	11B	Zebbug	90
2	Adam	11W	Marsa	45
3	Barry	11A	England	64
4	Jonothan	11C	Attard	98
5	Matthew	11W	Siggiewi	32
6	Luke	11S	Bugibba	78
7	Josh	11S	Burmarrad	73
8	Liam	11A	Qormi	40
9	Sean	11Q	Valletta	83

10	Maria	11B	Zebbug	92
11	Ryan	11B	Zabbar	85
12	Ben	11Q	Qawra	75
13	Jimmy	11A	San Pawl	89
14	Samuel	11S	Mosta	22
15	Miguel	11S	Naxxar	57

Checking the Grades

Based on the data below the testing was carried out, the following are the grades obtained by each student.

Number	Name	Class	Location	Mark	Grades: Expected	Grades: Actual
1	Joe	11B	Zebbug	90	A	A
2	Adam	11W	Marsa	45	F	F
3	Barry	11A	England	64	C	C
4	Jonothan	11C	Attard	98	A	A
5	Matthew	11W	Siggiewi	32	F	F
6	Luke	11S	Bugibba	78	B	B
7	Josh	11S	Burmarrad	73	B	B
8	Liam	11A	Qormi	40	F	F
9	Sean	11Q	Valletta	83	B	B
10	Maria	11B	Zebbug	92	A	A
11	Ryan	11B	Zabbar	85	A	A
12	Ben	11Q	Qawra	75	B	B
13	Jimmy	11A	San Pawl	89	A	A

14	Samuel	11S	Mosta	22	F	F
15	Miguel	11S	Naxxar	57	D	D

Checking the Sum and Average

Based on the data below the testing was carried out, the following are the sum of all students and the average of all students.

Sum

Expected Result	Actual Result
1023	1023

Average

Expected Result	Actual Result
68.2	68.2

Checking Maximum and Minimum Marks

Based on the data below the testing was carried out, the following are the maximum and minimum marks obtained.

Maximum Mark

Expected Result				
Jonothan	11C	Attard	98	A
Actual Result				
Jonothan	11C	Attard	98	A

Class Marks

Minimum Mark

Expected Result				
Samuel	11S	Mosta	22	F
Actual Result				
Samuel	11S	Mosta	22	F

Checking Above or Below Average

Based on the data below the testing was carried out, the following are whether the student was above or below average.

Student	Expected Result	Actual Result
Joe	Above Average	Above Average
	Expected Result	**Actual Result**
Liam	Below Average	Below Average

Search Menu

Checking the Search Menu

Test data	Purpose of test	Expected result	Actual Result
1	To check that data at the boundaries is accepted	Accepted	Accepted
4			
3	To check that data within the boundaries is accepted.	Accepted	Accepted
2			

8	To check that data outside the boundaries is not accepted	Not Accepted	Not Accepted
-6			

Checking the Search by mark

Based on the data below the testing was carried out, the following is to check whether the mark was found. The mark of Joe was found using search by mark.

Purpose of test				
To check that the mark is found				
Expected Result				
Joe	11B	Zebbug	90	A
Actual Result				
Joe	11B	Zebbug	90	A

Checking the Search by name

Based on the data below the testing was carried out, the following is to check whether the name was found. The name of Joe was found using search by name.

Purpose of test
To check that the name is found
Expected Result

Class Marks

Joe	11B	Zebbug	90	A
Actual Result				
Joe	11B	Zebbug	90	A

Checking the Search by Class

Based on the data below the testing was carried out, the following is to check whether the class was found. The class of Joe was found using search by class.

Purpose of test				
To check that the class is found				
Expected Result				
Joe	11B	Zebbug	90	A
Actual Result				
Joe	11B	Zebbug	90	A

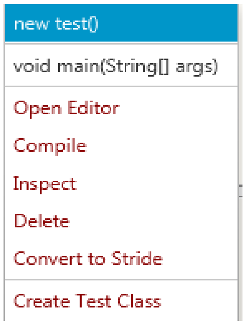
User Instructions

Loading the program

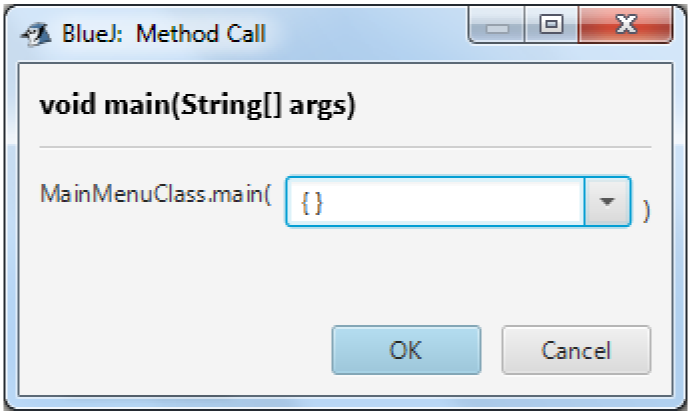
Following the instructions to load the program:

1. Open BlueJ
2. On the top left of the screen select project >Open Project

3. Find the location where the project is saved, and select the file “MatthiasComputingProjectClassMarks “
4. Right Click all the classes and select compile.



5. Right Click on MainMenuClass and select void main (String[] args) on the drop-down menu.
6. Press ‘OK’

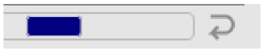

BlueJ: Method Call

void main(String[] args)

MainMenuClass.main({ })

This will run the program in the new window:

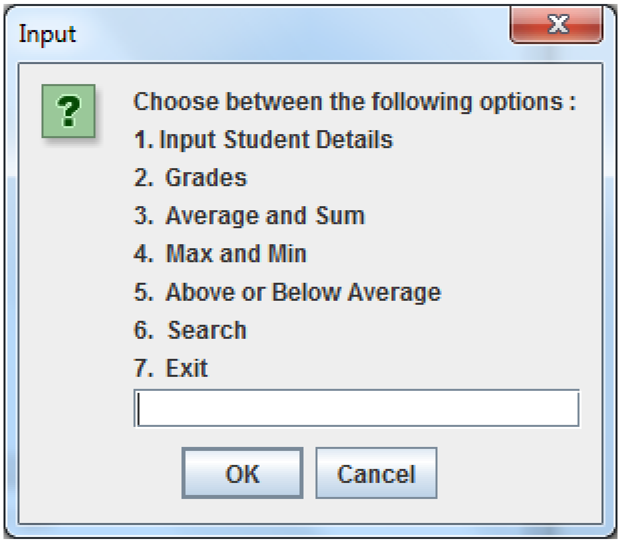
To disable the program, right click on the bar, and from the drop-down Menu click ‘Reset Java Virtual Machine’.



Operating the program

The program will display the following **Menu:**

Option 1: Input
Option 2: Grades
Option 3: Average and Sum
Option 4: Max and Min
Option 5: Above or Below Average
Option 6: Search
Option 7: Exit



Choose between the following options :
1. Input Student Details
2. Grades
3. Average and Sum
4. Max and Min
5. Above or Below Average
6. Search
7. Exit

Option 1:
The program will ask the user to input the number of students in the class. User is to input names, marks, locations and classes. The program then displays all the details. If the mark is not within the correct parameters an error is showed and the user is required to renter mark.

Option 2:
The program will calculate the grades of each student and will display the name, mark, location, class and grade for every student.

Option 3:
The program will calculate the sum and average of all students' marks. The program then will display the sum and average.

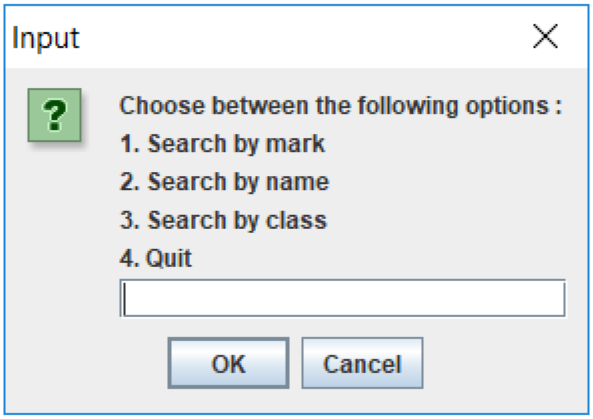
Option 4:
The program will calculate the maximum and the minimum mark. The program will also display the name of the student with the maximum mark, class, grade and location. The program will also display the name of the student with the minimum mark, class, grade and location.

Option 5:

The program will check if the mark is above or below the average. The program will then display all the students' details and whether their mark was above or below the average.

Option 6:
The program will enable the user to engage in a search menu, comprising of four options, three search options and one exit facility.

Search Menu – Option 1 – search by mark
Option 2 – search by name
Option 3 -- search by class
Option 4 – Exit



Choose between the following options :
1. Search by mark
2. Search by name
3. Search by class
4. Quit

Option 6.1
The program will ask the user to input the mark to search for. The program will find the mark in location and will display the student's details who attained that particular mark. If the mark is not found, the user will be returned to the Search Menu.

Option 6.2
The program will ask the user to input the name to search for. The program will find the name in location and will display the student's details who has that particular name. If the name is not found, the user will be returned to the Search Menu.

Option 6.3
The program will ask the user to input the class to search for. The program will find the class in location and displays the student's details who is in that particular class. If the class is not found, the user will be returned to the Search Menu.

Option 6.4
The program will enable the user to exit the search menu and revert back to the main menu.

Option 7
The program will display an ending phrase.

If The Option doesn't exist, the program will display 'Invalid'

Comments and Conclusion:

Limitations of the program

The aim of the program was reached whereby the program executed various options which resulted in no syntax, runtime and logical errors. However, there were the following shortcomings:

- Unfortunately, the window cannot be closed from any buttons which results in system crashes
- If the user enters a different data type variable the program will crash.
- There are a limited number of options.

Possible improvements

The following improvements are to be taken into consideration. There needs to be:

- An increase in the number of options such as a new option whereby there is an average for each different class.
- An increase in the number of marks, being the program only working on the final mark such as making a mark for each core subject like maths.
- The possibility that the windows can be closed by buttons.
- The possibility that the Options can be accessed by specific buttons.