**Name:** Donark Patel (DP663)

**Class:** IS114-451

**Professor:** Maura Deek

**Date:** 9/21/2019

**1. Formulating the Problem**

**1.1 Problem Description**

Create a program in Java that creates a roster of student information.

The roster should include Student ID, Student first name, Student last name, Student email address, Student age. The program should also have an array of grade. User should be able to print all student’s information or specific student information using Student ID.

**1.2 Verbalization**

*What is the goal?*

Create a roster of student’s information.

*What are the givens?*

Student ID, First name, Last name, E-mail address, Age, Array of grades are inputted by the user.

*What are the unknowns?*

Average of grades and Letter grade.

1.3 **Information Elicitation**

*Goal* : Create a roster of student’s information.

*Givens*: Student information and grades.

*Unknowns* : Average of grades and Letter grade.

*Conditions* : None

**2. Planning the Solution**

**2.1 Solution Strategy**

Ask the user to input the values of student ID, first name, last name, email address, and age. Using this value create a student object. Ask the user to input the grades for 5 different assignments. Then create an array for grades. Store the grades into an array. Once all the information is gathered from a user. Calculate the average and letter grade.

**)2.2 Goal Decomposition**

*Sub-goal 1*: Get inputs from the user.

*Sub-goal 2*: Create a Student object.

*Sub-goal 3*: Create an array for grades.

*Sub-goal 4*: Insert the value of grades into an array.

*Sub-goal 5*: Calculate the average.

*Sub-goal 6*: Calculate the letter grade.

*Sub-goal 7*: Give options to the user what they want to view and print. option 1: display all student information, option 2: display specific student information, option 3: exit the program.

*Sub-goal 8*: Print users request.

**2.3 Resources**

*Relevant formulas*

Average = sum of grade / number of grades

**2.4 Data Organization and Description**

Input:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Description | Origin | Used in Sub-goal # |
| studentID | Student’s ID | User | 1, 2 |
| firstName | Student’s first name | User | 1, 2 |
| lastName | Student’s last name | User | 1, 2 |
| emailAddress | Student’s email address | User | 1, 2 |
| age | Student’s age | User | 1, 2 |
| Assignment 1 grade | Assignment 1 grade | User | 1, 3, 4, 5 |
| Assignment 2 grade | Assignment 2 grade | User | 1, 3, 4, 5 |
| Assignment 3 grade | Assignment 3 grade | User | 1, 3, 4, 5 |
| Assignment 4 grade | Assignment 4 grade | User | 1, 3, 4, 5 |
| Assignment 5 grade | Assignment 5 grade | User | 1, 3, 4, 5 |

Output:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Description | Origin | Used in Sub-goal # |
| Student ID: | Student’s ID number | Screen | 8 |
| First name: | Student’s first name | Screen | 8 |
| Last name: | Student’s last name | Screen | 8 |
| Email address: | Student’s email address | Screen | 8 |
| Age: | Student’s age | Screen | 8 |
| Assignment 1 grade: | Assignment 1 grade | Screen | 8 |
| Assignment 2 grade: | Assignment 2 grade | Screen | 8 |
| Assignment 3 grade: | Assignment 3 grade | Screen | 8 |
| Assignment 4 grade: | Assignment 4 grade | Screen | 8 |
| Assignment 5 grade | Assignment 5 grade | Screen | 8 |
| Average: | Average of 5 grades | Screen | 8 |
| Letter grade: | Latter grade | Screen | 8 |

**3. Designing the Solution**

**)3.1 Structure Chart**

*First Level Decomposition*

*Goal Refinement*

**Sub-goal 1**

Ask the user “How many student’s information you want to enter?”

**Sub-Goal 1.1**

Get student’s information

**Sub-goal 1.2**

Validate the input.

**Sub-goal 2**

Create an object of the student.

**Sub-goal 2.1**

Create an array of student object.

**Sub-goal 2.2**

Insert the data of student object into the array of object.

**Sub-goal 3**

Create an array for grades.

**Sub-goal 4**

Insert the data of grade into the array using a loop

**Sub-goal 5**

Calculate the average.

**Sub-goal 6**

Calculate the letter grade**.**

**Sub-goal 7**

Give options to users to print student information.

**Sub-goal 8**

Print student information.

*Second Level Decomposition*

**()3.2 Module and Data Specifications**

**Name**: PrintStudentRecord – Print student’s information

**Input**: None

**Output**: Prints student’s information

**Logic**: Print out the board using the values current student object.

**Name**: getStudentID – Returns current student object ID.

**Input**: None

**Output**: None

**Logic**: Returns current student object ID.

**Name**: average – calculate average.

**Input**: None

**Output**: None

**Logic**: average = sum / 5

**Name**: LetterGrade – Provide a letter grade of the student’s.

**Input**: None

**Output**: None

**Logic**: Using an If statement determined the letter grade of the student.

**Name**: isNumber – Validate the user input. It determines if it is number or not.

**Input**: String

**Output**: Boolean

**Logic**: Using a try catch statement. Try to parse the string into an integer. It is pass then return true, otherwise return false.

**Name**: Main - That creates a roster of student information.

**Input**: None

**Output**: None

**Logic**: creates a roster of student information.

**Data:**

|  |  |  |
| --- | --- | --- |
| Name | Type | Structure |
| studentID | Integer | Variable |
| firstName | String | Variable |
| lastName | String | Variable |
| emailAddress | String | Variable |
| age | String | Variable |
| numberOfStudents | Integer | Variable |
| Input | String | Variable |
| validateInput | Boolean | Variable |

**3.3 Algorithm**

*Logic*

1.0: Ask the user how many student’s information wants to add.

1.1: Get the values from the user.

1.2: Validate the inputs.

2.0: Create an object of the student.

2.1: create an array of student object.

2.2: Insert the data of student object into the array of object.

3.0: Create an array for grades.

4.0: Insert the data of grade into the array using a loop.

5.0: Calculate the average by summing up student grade and dividing by 5.

6.0: Calculate the letter grade.

7.0: Display options to users to print student’s information

8.0: Display student information.

*Algorithm Description*

The program starts off by asking the user “How many student's information need to be stored?” Once the number input from a user. Program store that number into the variable numberOfStudents. Then the program asks the user to enter the student’s information. The program then validates each input from the user. If a user inputs incorrect information a user is prompt with a display of error code. The program also loops back to the previous question to re-enter the information. Then the program takes all the information and creates a Student object. Each student object is created are also stored in an object array. Then program stores the grades of the current student into an array. Then the program asks the user to choose what they want to display. If a user wants to display an entire roster. A loop goes through an object array and calls the print method (each time it loops). In the print method, the program calls an average() and letterGrad() method. Which calculate the current student grads and print the information. If a user wants to display particular students information by using a Student ID. Loops go through an object array and get the StudentID (Using getStudentID method). Then the program matches the Student ID with the user input. If it matches then it prints the student information of a current object.

**4. Translation**

**4.1** **Source Code**

//===================================================  
// Name : Donark Patel  
// SID : 31327058  
// Course : IT114  
// Section : 451  
// Instructor : Maura Deek  
// Assignment # : Programming Assignment 1  
// Date : 9/20/2019  
// Description : This program is too add and display student's information.  
//===================================================  
  
  
import java.util.Scanner;  
import javax.swing.JOptionPane;  
import java.util.regex.Pattern;  
  
public class Assignment1{  
  
public static void main (String[]args){  
  
 int studentID = 0; //studentID stores student's ID  
 String firstName = ""; //firstName stores student's first name  
 String lastName = ""; //lastName stores student's last name  
 String emailAddress = ""; //emailAddress stores student's email address  
 int age = 0; //age stores student's age.  
 int numberOfStudents = 0; // Number of student records  
  
 String input;  
 Boolean validateInput = false;  
  
  
  
 //Create a Scanner object to receive input from the user  
 Scanner keyboard = new Scanner (System.in);  
 do{  
 //Prompt users to indicate how many recodes to need to be created.  
 System.out.println("How many student’s information you want to enter? ");  
 input = keyboard.next();  
 if(isNumber(input))  
 {  
 int temp = Integer.parseInt(input);  
 if(temp < 11 && temp > 0)  
 {  
 numberOfStudents = temp;  
 validateInput = true;  
 }  
 else{  
 JOptionPane.showMessageDialog(null,"Input valid number, Enter number between 1 to 10");  
 validateInput = false;  
 }  
 }  
  
 else  
 {  
 JOptionPane.showMessageDialog(null,"Input valid number");  
 validateInput = false;  
  
 }  
 }while(validateInput == false);  
  
  
  
  
 //Array to store all student records.  
 Student[] studentArray = new Student[numberOfStudents];  
  
 //For loop to insert student information  
 for(int j = 0; j<numberOfStudents; j++ ){  
  
 System.out.println("\n\nStudent " + (j+1) + " information: ");  
 do{  
 //Prompt the user to enter student's ID  
 System.out.println("Enter student's ID");  
 input = keyboard.next();  
 if(isNumber(input))  
 {  
 studentID = Integer.parseInt(input);  
 validateInput = true;  
 }  
 else  
 {  
 JOptionPane.showMessageDialog(null,"Input valid number");  
 validateInput = false;  
 }  
 }while(validateInput == false);  
  
  
  
 do{  
 //Prompt the user to enter student's first name  
 System.out.println("Enter student's first name");  
 input = keyboard.next();  
 if(isOnlyAlphabet(input))  
 {  
 firstName = input;  
 validateInput = true;  
 }  
 else  
 {  
 JOptionPane.showMessageDialog(null,"Input valid first name");  
 validateInput = false;  
 }  
 }while(validateInput == false);  
  
  
 do{  
 //Prompt the user to enter student's last name  
 System.out.println("Enter student's last name");  
 input = keyboard.next();  
 if(isOnlyAlphabet(input))  
 {  
 lastName = input;  
 validateInput = true;  
 }  
 else  
 {  
 JOptionPane.showMessageDialog(null,"Input valid last name");  
 validateInput = false;  
 }  
 }while(validateInput == false);  
  
  
 do{  
 //Prompt the user to enter student's email address  
 System.out.println("Enter student's email address");  
 input = keyboard.next();  
 if(isValidEmail(input))  
 {  
 emailAddress = input;  
 validateInput = true;  
 }  
 else  
 {  
 JOptionPane.showMessageDialog(null,"Input valid email address");  
 validateInput = false;  
 }  
 }while(validateInput == false);  
  
 do{  
 //Prompt the user to enter student's age  
 System.out.println("Enter student's age");  
 input = keyboard.next();  
 if(isNumber(input))  
 {  
 int ageTemp = Integer.parseInt(input);  
 if(ageTemp > 0)  
 {  
 age = ageTemp;  
 validateInput = true;  
 }  
 else  
 {  
 JOptionPane.showMessageDialog(null,"Input valid number. Enter number grater then zero");  
 validateInput = false;  
 }  
 }  
 else  
 {  
 JOptionPane.showMessageDialog(null,"Input valid number");  
 validateInput = false;  
 }  
 }while(validateInput == false);  
  
  
 //Array for grades  
 int[] arrayGrade = new int[5];  
  
 //Prompt the user to enter student's assignment Grade using For loop  
 for(int i=0; i<5; i++){  
 do{  
 System.out.println("Enter student's assignment " + (i+1) + " grade");  
 input = keyboard.next();  
 if(isNumber(input))  
 {  
 int gradeTemp = Integer.parseInt(input);  
 if(gradeTemp < 101 && gradeTemp > -1){  
 arrayGrade[i] = gradeTemp;  
 validateInput = true;}  
 else{  
 JOptionPane.showMessageDialog(null,"Input valid number, Enter number between 0 to 100");  
 validateInput = false;  
  
 }  
 }  
 else  
 {  
 JOptionPane.showMessageDialog(null,"Input valid number");  
 validateInput = false;  
 }  
 }while(validateInput == false);  
  
 }  
  
 //Declare and instantiate a Student object  
 Student newStudent = new Student(studentID, firstName, lastName, emailAddress, age,arrayGrade);  
  
  
 //Inserting newStudent object data in StudentArray  
 studentArray [j] = newStudent;  
  
 }  
  
 //Prompt the user to indicate if they want to display data of all students.  
 System.out.println("\n\nEnter number 1 to display the entire roster");  
 System.out.println("\n\nEnter Student's ID to display to display a specific student information");  
 System.out.println("\n\nEnter number 3 to exit");  
 int temp = keyboard.nextInt();  
  
  
 while(temp!= 3){  
  
 //Prompt the user to indicate if they want to display data of all students.  
  
  
 if (temp == 1){  
 //For loop to display all student information.  
 for(int a = 0; a<numberOfStudents; a++ ){  
 studentArray [a].PrintStudentRecord();  
 }  
  
 }  
 // else if statement to find the student ID  
 else if (temp != 1){  
 for(int b = 0; b<numberOfStudents; b++ ){  
 int tempStudentID = studentArray [b].getStudentID();  
 //if statement to validate if student id was found  
 if(temp == tempStudentID){ //if statement to validate if student id was found  
 studentArray [b].PrintStudentRecord();  
  
 }  
 }  
  
 }  
  
 //else print error message  
 else{  
 System.out.println("\n Invalid entry: Please enter correct ID number or enter 1 to display all students information");  
  
 }  
 System.out.println("\n\nEnter number 1 to display the entire roster");  
 System.out.println("\n\nEnter Student's ID to display to display a specific student information");  
 System.out.println("\n\nEnter number 3 to exit");  
 temp = keyboard.nextInt();  
  
 }  
 keyboard.close();  
  
}  
 public static boolean isNumber(String in)  
 {  
 try  
 {  
 Integer.parseInt(in);  
 return true;  
 }  
 catch (Exception E)  
 {  
 return false;  
 }  
 }  
 public static boolean isOnlyAlphabet(String al)  
 {  
 return ((!al.equals("")) && (al.matches("^[a-zA-Z]\*$"))&& (al != null));  
 }  
  
 public static boolean isValidEmail(String email)  
 {  
 String regex = "^[a-zA-Z0-9\_+&\*-]+(?:\\."+  
 "[a-zA-Z0-9\_+&\*-]+)\*@" +  
 "(?:[a-zA-Z0-9-]+\\.)+[a-z" +  
 "A-Z]{2,7}$";  
  
 Pattern p1 = Pattern.compile(regex);  
 if (email == null)  
 return false;  
 return p1.matcher(email).matches();  
 }  
  
}

//===================================================  
// Name : Donark Patel  
// SID : 31327058  
// Course : IT114  
// Section : 451  
// Instructor : Maura Deek  
// Assignment # : Programming Assignment 1  
// Date : 9/20/2019  
// Class : This class created student object.   
//===================================================  
  
  
public class Student{  
  
 private int studentID; //studentID stores student's ID  
 private String firstName; //firstName stores student's first name  
 private String lastName; //lastName stores student's last name  
 private String emailAddress; //emailAddress stores student's email address  
 private int age; //age stores student's age.  
 private int[] grades; //grades array stores students grade  
  
  
 /\*\*  
 Constructor with parameters  
 sID Student's ID  
 sFirst Studetnt's First Name  
 sLast Student's Last Name  
 sEmail Student's Email Address  
 sAge Student's Age  
 sGrade array of Student's Grade  
 \*/  
  
 public Student(int sID, String sFirst, String sLast, String sEmail, int sAge, int [] sGrade)  
 {  
 studentID = sID;  
 firstName = sFirst;  
 lastName = sLast;  
 emailAddress = sEmail;  
 age = sAge;  
 grades = sGrade;  
 }  
 //Print method to print student information.  
 public void PrintStudentRecord()  
 {  
  
 System.out.println("\n\nID:" + studentID);  
 System.out.println("First Name:" + firstName);  
 System.out.println("Last Name:" + lastName);  
 System.out.println("Email:" + emailAddress);  
 System.out.println("Age:" + age);  
 System.out.println("Assignment 1 grade: " + grades[0]);  
 System.out.println("Assignment 2 grade: " + grades[1]);  
 System.out.println("Assignment 3 grade: " + grades[2]);  
 System.out.println("Assignment 4 grade: " + grades[3]);  
 System.out.println("Assignment 5 grade: " + grades[4]);  
 System.out.println("Average: " + average());  
 System.out.println("Letter Grade: " + letterGrade());  
 }  
 //Method to return Student ID  
 public int getStudentID()  
 {  
 return studentID;  
 }  
 //Method to calculate average  
 public double average()  
 {  
  
 double sum = 0;  
 for(int i=0; i<5; i++)  
 sum += grades[i];  
  
 double average = sum / 5;  
  
 return average;  
 }  
 // Method to calculate letter grade  
 public char letterGrade()  
 {  
 double score = average();  
 if (score >=90 ){  
 return 'A';}  
 else if (score >= 80){  
 return 'B';}  
 else if (score >= 70){  
 return 'C';}  
 else if (score >= 60){  
 return 'D';}  
 else return 'F';  
  
 }  
  
} **return p1.matcher(email).matches();**

**}**

**}**

**)4.2 Program and Module Description**

PrintStudentRecord

This function prints out student’s information.

getStudentID

This function returns the current student object’s student ID variable.

average

calculate and return the average of grades.

letterGrade

calculate and return a letter grade.

isNumber

Validate if the input is a number or not. Returns boolean.

isOnlyAlphabets

Validate if the input is only alphabets or not. Returns boolean.

isValidEmail

Validate if the input is in proper email format. Returns boolean.

Main

The main function asks under to input data of student’s information. Then the function creates a student object. For each object, it creates it also store in an array of object. Then the program asks the user to choose what they want to display. If a user wants to display an entire roster. A loop goes through an object array and calls the print method (each time it loops). In the print method, the program calls an average() and letterGrad() method. Which calculate the current student grads and print the information. If a user wants to display particular students information by using a Student ID. Loops go through an object array and get the StudentID (Using getStudentID method). Then the program matches the Student ID with the user input. If it matches, then it prints the student information of a current object.

**5. Solution Testing**

Test Case: 1 – Data validation.

|  |  |  |  |
| --- | --- | --- | --- |
| **Question** | **Entry** | **Invalid Entry** | **Reason** |
| How many student’s information you want to enter? | 15 | Invalid input | More then 10 |
| Enter student's ID | AB123 | Invalid input | Not a number |
| Enter student's first name | 123 | Invalid input | Not an alphabet |
| Enter student's last name | 123 | Invalid input | Not an alphabet |
| Enter student's email address | [DonarkPatelatgmail.com](mailto:DonarkPatel@gmail.com) | Invalid input | Not a valid email format |
| Enter Student’s age | AB | Invalid input | Not a number |
| Enter student's assignment 1 grade | AB | Invalid input | Not a number |
| Enter student's assignment 2 grade | -1 | Invalid input | Less than zero |
| Enter student's assignment 3 grade | 101 | Invalid input | Grater then 100 |
| Enter student's assignment 4 grade | 50 | valid input |  |
| Enter student's assignment 5 grade | 95 | valid input |  |

Test Case: 1

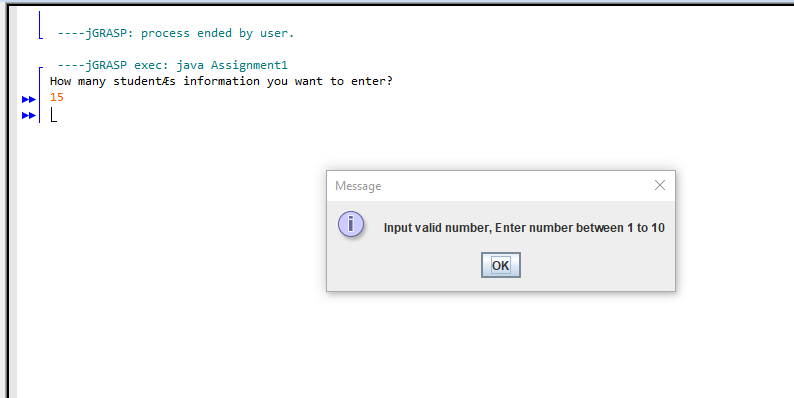
|  |  |
| --- | --- |
| Question | Valid entry |
| How many student’s information you want to enter? | 1 |
| Enter student's ID | 123456 |
| Enter student's first name | Donark |
| Enter student's last name | Patel |
| Enter student's email address | [DonarkPatel@gmail.com](mailto:DonarkPatel@gmail.com) |
| Enter student's assignment 1 grade | 100 |
| Enter student's assignment 2 grade | 90 |
| Enter student's assignment 3 grade | 80 |
| Enter student's assignment 4 grade | 70 |
| Enter student's assignment 5 grade | 60 |

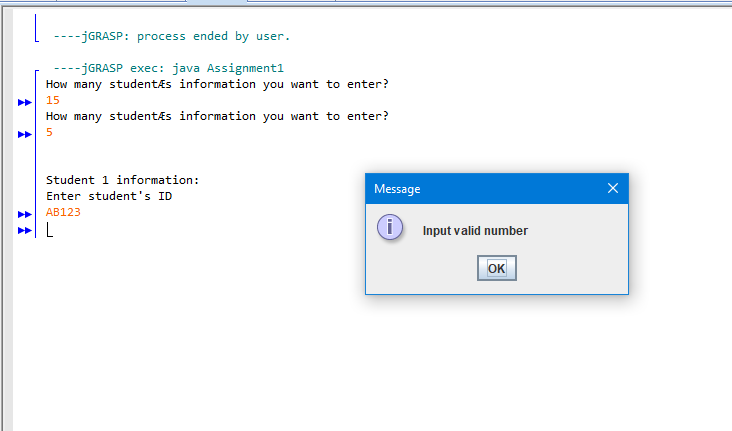
Test Case: 2

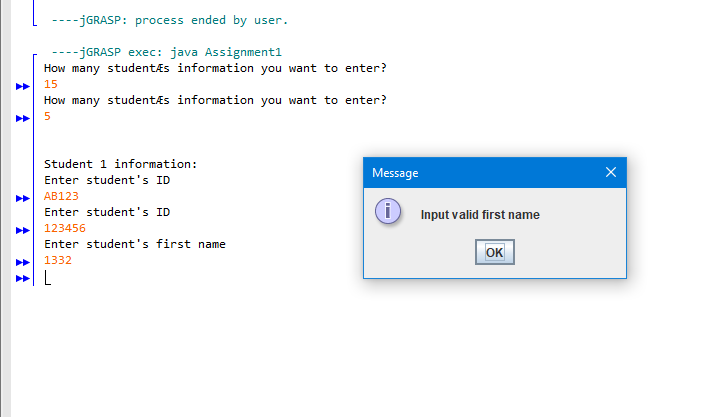
|  |  |
| --- | --- |
| Question | Valid entry |
| How many student’s information you want to enter? | 3 |
| Enter student's ID | 123456, 234567, 345678 |
| Enter student's first name | Donark, John, David |
| Enter student's last name | Patel, Smith, Williams |
| Enter student's email address | [DonarkPatel@gmail.com, nichoj@gmail.com, jaxweb@yahoo.com](mailto:DonarkPatel@gmail.com) |
| Enter student's assignment 1 grade | 100, 80, 51 |
| Enter student's assignment 2 grade | 95, 85, 55 |
| Enter student's assignment 3 grade | 96, 87, 52 |
| Enter student's assignment 4 grade | 93, 82, 53 |
| Enter student's assignment 5 grade | 99, 89, 56 |

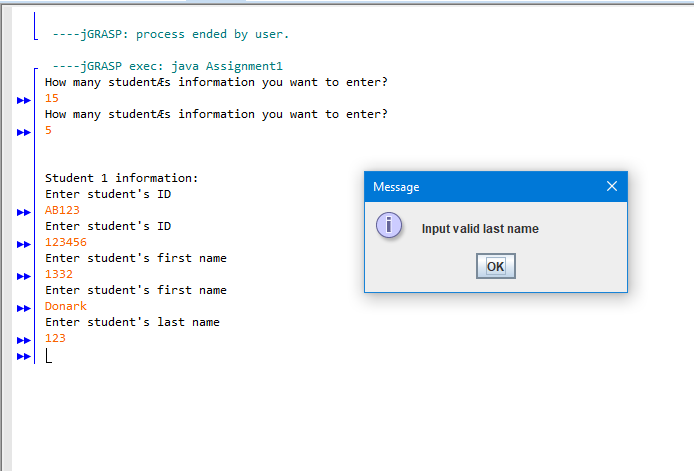
**(This is just a sample what you need to submit are actual screenshots of I/O or files)**

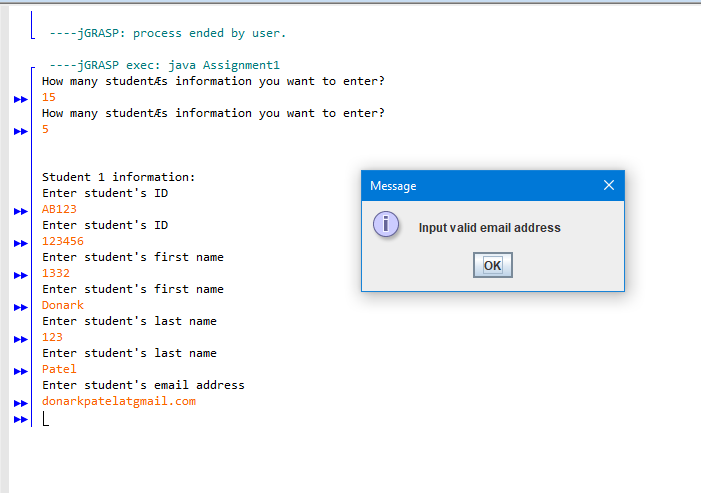
Test Case 1:

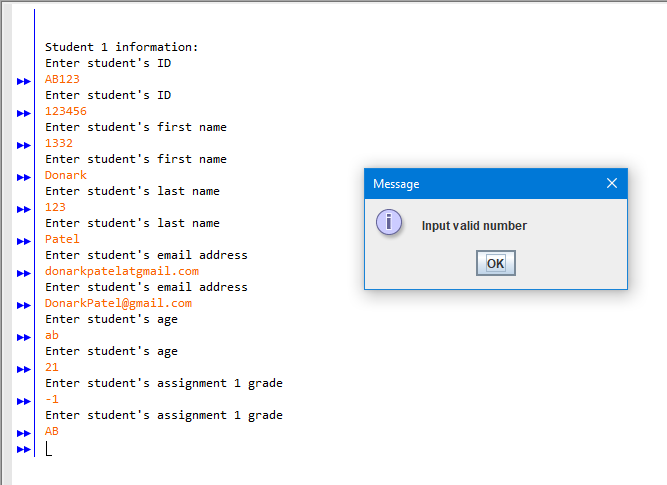
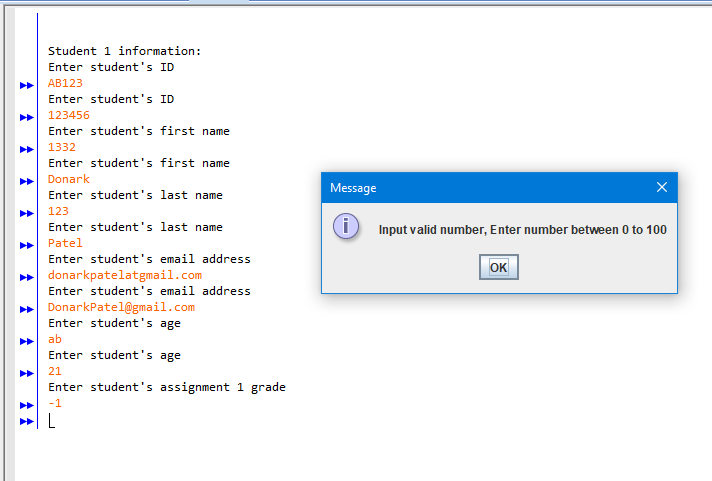
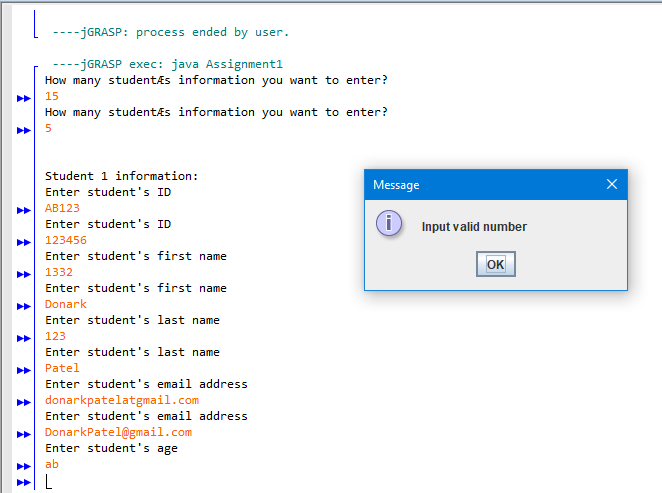


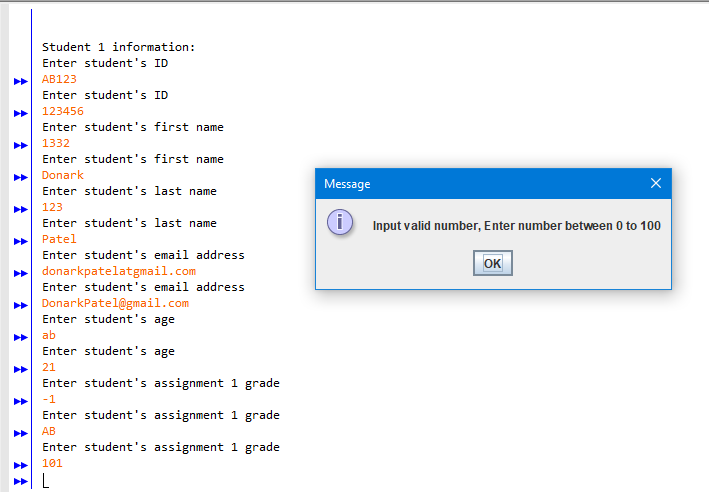


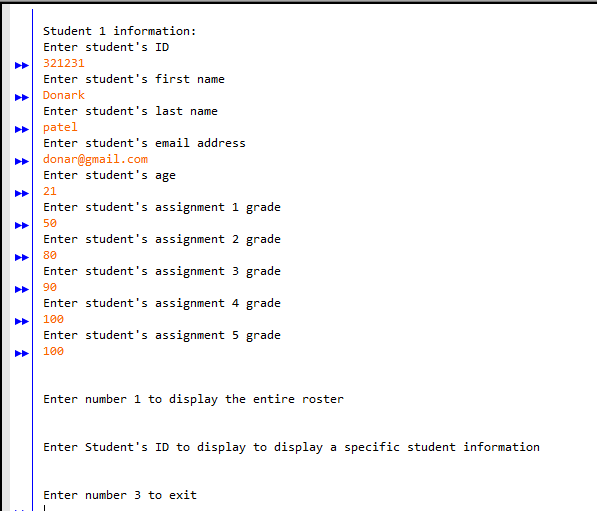












Test case 2:

