

Sachin Karki, Reetu Shrestha

Dr. Wen Xu

CSCI 2493-01

15 November 2020

## Course Project

### Student Management System

#### Introduction

In this project, we developed a console-based record keeping program named as *Student Management System* that simply records and retrieves basic academic information. To begin with, this program allows a user or say a student to keep track of their information such as name, ID, GPA and balance. To elaborate on this, users are given 4 major options to choose from. Beginning with the option Add, users can also View and Delete their account. In addition to this, users can view their due balance and ultimately, they can exit from the program.

#### Functions

The program uses multiple functions and methods to achieve the execution of the program. Here is the breakdown of the functions used in the program.

- 1) **Input from Array:** To achieve this, we proceeded by importing “import.java.util.ArrayList;”. We then declared required ArrayList inside the ‘StudentManagement’ class and stored four different variables (name, id, grade, balance) to be used later on for passing that value into the ArrayList in ‘addAccount’ public method.

```
*StudentManagement.java
1 import java.util.ArrayList; //Provides Resizable-array Implement List Interface
2 import java.util.Scanner; //Scanner Class for input
3
4 public class StudentManagement {
5
6     ArrayList<String> name = new ArrayList<String>();
7     ArrayList<String> id = new ArrayList<String>();
8     ArrayList<Double> grade = new ArrayList<Double>(); // For decimal number
9     ArrayList<Integer> balance = new ArrayList<Integer>(); // For whole number
10
11     public void addAccount() //Public Method
12     {
13         String userName="";
14         String idNum="";
15         Double userGrade=0.0;
16         int userBalance=0;
17         Scanner input= new Scanner(System.in);
18         System.out.println("To Add An Account Please Enter Your Name.");
19         if (input.hasNextLine())
20         {
21             userName=input.nextLine();
22         }
23
24         System.out.println("Enter Your ID Number:");
25         if (input.hasNextInt())
26         {
27             idNum= input.nextLine();
28         }
29         System.out.println("Enter Your GPA:");
30         Scanner input1= new Scanner(System.in);
31         userGrade= input1.nextDouble();
32
33         System.out.println("Enter Your Balance Due:");
34         Scanner input2= new Scanner(System.in);
35         userBalance = input2.nextInt();
36
37         //Adding All the information to the ArrayList.
38         name.add(userName);
39         id.add(idNum);
40         grade.add(userGrade);
41         balance.add(userBalance);
42     }
43 }
```

- 2) **Scanner Class:** To ask user input and pass it to array list. (import java.util.Scanner;)
- 3) **Variables:** A great deal of variables were used but out of all, notable variables like “userName, idNum, userGrade, userBalance” were used to associate name, ID, GPA and balance due respectively. These four variables would also rely on string, string, double and integer datatypes respectively.
- 4) **If/else:** An effective use of if/else was utilized along with the loop inside one of the public method “public method viewAccount()”. Based on the given input, the program would execute the prompt, right input (if), the program would retrieve the account and in case of wrong input (else), the program would print error prompt.
- 5) **Methods:** Throughout the program, 3 methods were utilized mostly to achieve execution. For example, adding an account was done with “public void addAccount()” public method. Similarly viewing an account was done with “public int viewAccount()” public method. Lastly, a main method was used to execute the welcome screen and ultimately rest of program from there.
- 6) **Switch Case:** An excellent application of Switch-Case we used in the program was during the user prompt section where program would ask user to make choices from the option. Numbers 1,2,3,4,5 were used as an input to trigger different options. 1 would trigger add, 2 would trigger view and so on. This allowed program to switch back and forth between different available options.
- 7) **Loop:** One good use of loop we used in this program is ‘for loop’ to prompt and print invalid ID every time user would feed wrong ID that was not in the system.

```

89 //Using Switch-Case To Trigger User Choice
90 switch(userChoice)
91 {
92 case 1:
93     myMain.addAccount();
94     break;
95 case 2:
96     System.out.println("Enter Your ID Number");
97
98     Scanner input2=new Scanner(System.in);
99     myID=input2.nextLine();
100     int a1= myMain.viewAccount(myID);
101     if(a1!=-1)
102     {
103         System.out.println("----Student Information Summary----");
104         System.out.println("NAME: "+ myMain.name.get(a1));
105         System.out.println("ID: "+ myMain.id.get(a1));
106         System.out.println("GRADE: "+ myMain.grade.get(a1));
107         System.out.println("BALANCE: "+ myMain.balance.get(a1));
108     }
109     break;
110 case 3:
111     System.out.println("Enter the ID You Would like to delete:");
112     Scanner input3=new Scanner(System.in);
113     myID=input3.nextLine();
114     int a2= myMain.viewAccount(myID);
115     if(a2!=-1)
116     {
117         myMain.balance.remove(a2);
118         myMain.grade.remove(a2);
119         myMain.name.remove(a2);
120         myMain.id.remove(a2);
121         System.out.println("Succesfully Deleted.");
122     }
123     break;
124 case 4:
125     System.out.println("Enter The ID You Would Like The Balance For:");
126     Scanner input4=new Scanner(System.in);
127     mvID=input4.nextLine();

```

```

public int viewAccount(String Id) |
{
    int number=0;
    for (int i=0;i<id.size();i++)
    {
        if (id.contains(Id))
        {
            number= i;
        }
        else
        {
            System.out.println("There No Such ID In Our List. Try Again!");
            return -1;
        }
    }
    return number;
}

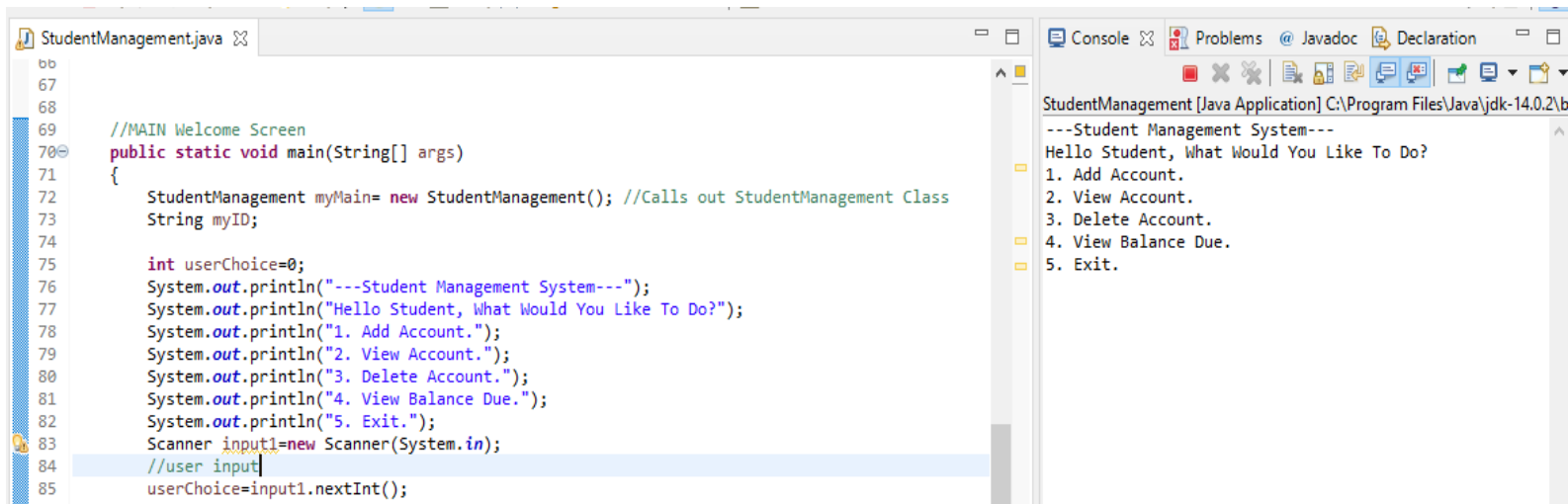
```

If/else and For loop

Switch-Case

## Code Execution & Results

1. **Main Method:** The program will begin with a Welcome screen created by the main method.



```

StudentManagement.java
66
67
68
69 //MAIN Welcome Screen
70 public static void main(String[] args)
71 {
72     StudentManagement myMain= new StudentManagement(); //Calls out StudentManagement Class
73     String myID;
74
75     int userChoice=0;
76     System.out.println("---Student Management System---");
77     System.out.println("Hello Student, What Would You Like To Do?");
78     System.out.println("1. Add Account.");
79     System.out.println("2. View Account.");
80     System.out.println("3. Delete Account.");
81     System.out.println("4. View Balance Due.");
82     System.out.println("5. Exit.");
83     Scanner input1=new Scanner(System.in);
84     //user input
85     userChoice=input1.nextInt();

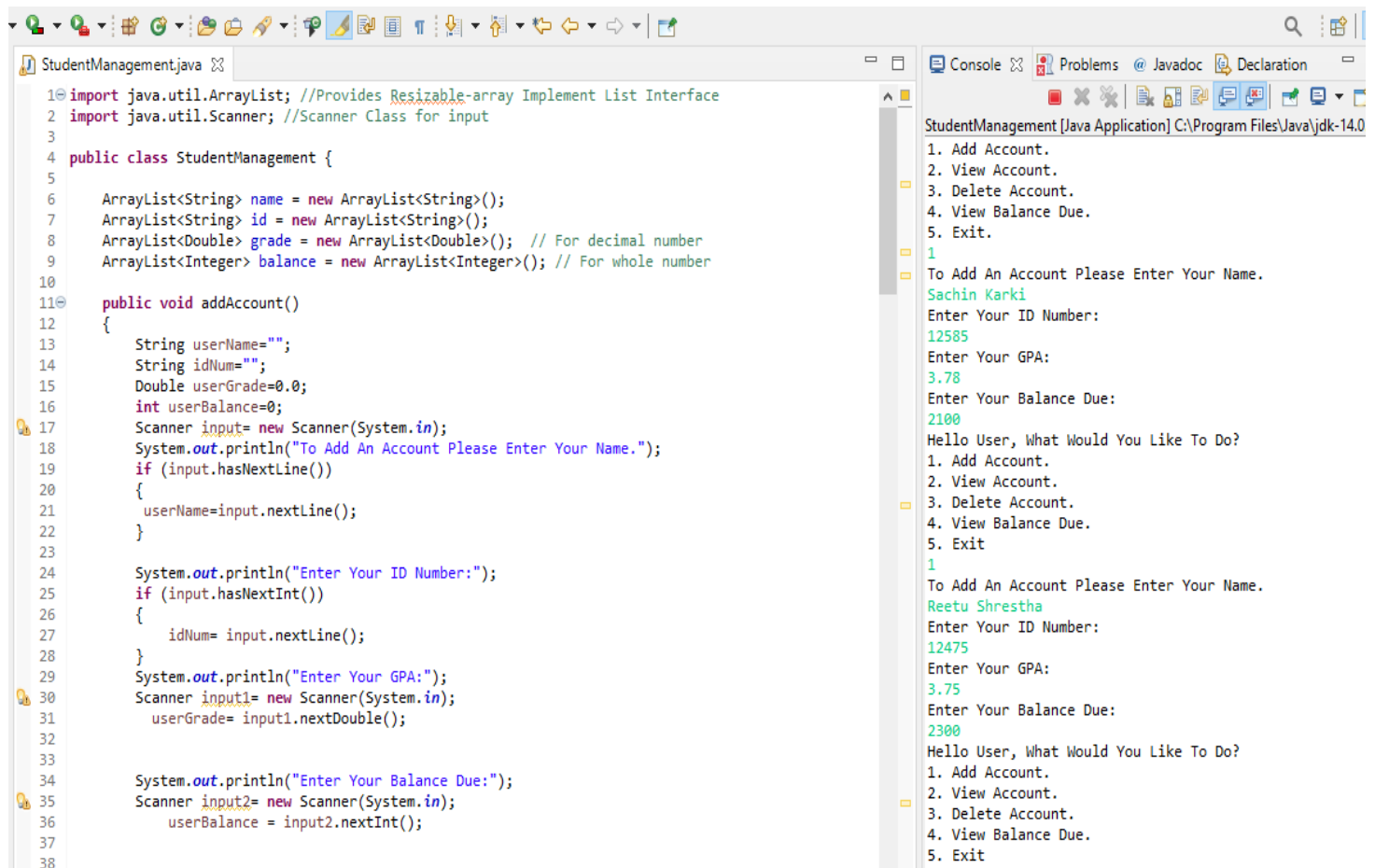
```

```

Console
StudentManagement [Java Application] C:\Program Files\Java\jdk-14.0.2\bin
---Student Management System---
Hello Student, What Would You Like To Do?
1. Add Account.
2. View Account.
3. Delete Account.
4. View Balance Due.
5. Exit.

```

1. **Add Account:** Pressing '1' will trigger the 'add account' option which will prompt user to enter their name, ID number, GPA and their balance. An example has been demonstrated below.



```

StudentManagement.java
1 import java.util.ArrayList; //Provides Resizable-array Implement List Interface
2 import java.util.Scanner; //Scanner Class for input
3
4 public class StudentManagement {
5
6     ArrayList<String> name = new ArrayList<String>();
7     ArrayList<String> id = new ArrayList<String>();
8     ArrayList<Double> grade = new ArrayList<Double>(); // For decimal number
9     ArrayList<Integer> balance = new ArrayList<Integer>(); // For whole number
10
11 public void addAccount()
12 {
13     String userName="";
14     String idNum="";
15     Double userGrade=0.0;
16     int userBalance=0;
17     Scanner input= new Scanner(System.in);
18     System.out.println("To Add An Account Please Enter Your Name.");
19     if (input.hasNextLine())
20     {
21         userName=input.nextLine();
22     }
23
24     System.out.println("Enter Your ID Number:");
25     if (input.hasNextInt())
26     {
27         idNum= input.nextLine();
28     }
29     System.out.println("Enter Your GPA:");
30     Scanner input1= new Scanner(System.in);
31     userGrade= input1.nextDouble();
32
33
34     System.out.println("Enter Your Balance Due:");
35     Scanner input2= new Scanner(System.in);
36     userBalance = input2.nextInt();
37
38

```

```

Console
StudentManagement [Java Application] C:\Program Files\Java\jdk-14.0
1. Add Account.
2. View Account.
3. Delete Account.
4. View Balance Due.
5. Exit.
1
To Add An Account Please Enter Your Name.
Sachin Karki
Enter Your ID Number:
12585
Enter Your GPA:
3.78
Enter Your Balance Due:
2100
Hello User, What Would You Like To Do?
1. Add Account.
2. View Account.
3. Delete Account.
4. View Balance Due.
5. Exit
1
To Add An Account Please Enter Your Name.
Reetu Shrestha
Enter Your ID Number:
12475
Enter Your GPA:
3.75
Enter Your Balance Due:
2300
Hello User, What Would You Like To Do?
1. Add Account.
2. View Account.
3. Delete Account.
4. View Balance Due.
5. Exit

```

2. **View Account:** Pressing '2' will trigger 'view account' option which will prompt user to input ID number. Giving a valid ID number will return student information and an invalid ID will inform the user about the absence of such ID. A demonstration is given below

```
switch(userChoice)
{
case 1:
    myMain.addAccount();
    break;
case 2:
    System.out.println("Enter Your ID Number");

    Scanner input2=new Scanner(System.in);
    myID=input2.nextLine();
    int a1= myMain.viewAccount(myID);
    if(a1!=-1)
    {
        System.out.println("---Student Information Summary---");
        System.out.println("NAME: "+ myMain.name.get(a1));
        System.out.println("ID: "+ myMain.id.get(a1));
        System.out.println("GRADE: "+ myMain.grade.get(a1));
        System.out.println("BALANCE: "+ myMain.balance.get(a1));
    }
}
```

```
2
Enter Your ID Number
12587
There No Such ID In Our List. Try Again!
Hello User, What Would You Like To Do?
1. Add Account.
2. View Account.
3. Delete Account.
4. View Balance Due.
5. Exit
2
Enter Your ID Number
12475
---Student Information Summary---
NAME: Reetu Shrestha
ID: 12475
GRADE: 3.75
BALANCE: 2300
```

3. **Delete Account:** Pressing '3' will trigger 'delete account' option which will ask user for the ID number. Entering a right ID will erase the account associated with that ID; entering a wrong ID will simply inform user about no presence of such ID in the system. A demonstration is given below

```
break;
case 3:
    System.out.println("Enter the ID You Would like to delete:");
    Scanner input3=new Scanner(System.in);
    myID=input3.nextLine();
    int a2= myMain.viewAccount(myID);
    if(a2!=-1)
    {
        myMain.balance.remove(a2);
        myMain.grade.remove(a2);
        myMain.name.remove(a2);
        myMain.id.remove(a2);
        System.out.println("Succesfully Deleted.");
    }
    break;
```

```
5. Exit
3
Enter the ID You Would like to delete:
147
There No Such ID In Our List. Try Again!
Hello User, What Would You Like To Do?
1. Add Account.
2. View Account.
3. Delete Account.
4. View Balance Due.
5. Exit
3
Enter the ID You Would like to delete:
12575
Succesfully Deleted.
```

4. **View Balance:** Pressing '4' will trigger 'view balance' which will again ask user for ID number. Feeding the correct ID will return due balance associated with that ID and feeding a wrong ID will simply inform user about no presence of such ID. A demonstration is given below.

```

        System.out.println( "Successfully Deleted. ");
    }
    break;
case 4:
    System.out.println("Enter The ID You Would Like The Balance For:");
    Scanner input4=new Scanner(System.in);
    myID=input4.nextLine();
    int a3= myMain.viewAccount(myID);
    if(a3!=-1)
    {
        System.out.println("Your Balance Due Is: "+ myMain.balance.get(a3));
    }
    break;
default:
    System.out.println("Please Enter A Valid Number.");
    break;
}
System.out.println("Hello User, What Would You Like To Do?");
System.out.println("1. Add Account.");

```

```

+
To Add An Account Please Enter Your Name.
Sachin Karki
Enter Your ID Number:
12585
Enter Your GPA:
3.45
Enter Your Balance Due:
2100
Hello User, What Would You Like To Do?
1. Add Account.
2. View Account.
3. Delete Account.
4. View Balance Due.
5. Exit
4
Enter The ID You Would Like The Balance For:
12585
Your Balance Due Is: 2100
Hello User, What Would You Like To Do?

```

5. **Exit:** Pressing '5' will trigger 'exit' option, which will simply take user to the end of the program. A demonstration is given below.

```

Scanner newScan= new Scanner(System.in);
if (newScan.hasNextInt())
{
    userChoice = newScan.nextInt();
}
//userChoice=newScan.nextInt();

}while(userChoice!=5);
System.out.println("---EXITED---");
}

```

```

5. Exit
4
Enter The ID You Would Like The Balance For:
12585
Your Balance Due Is: 2100
Hello User, What Would You Like To Do?
1. Add Account.
2. View Account.
3. Delete Account.
4. View Balance Due.
5. Exit
5
---EXITED---

```

## Summary

As mentioned above, this is a console-based program so users are able to dynamically update within the console however the information that are fed as an input will not be stored into the memory meaning the program is not able to remember the last information soon as you terminate or restart the program. There are few restrictions to the program as well. First of all, options like view account, delete account and view balance are not available soon as one starts the program. To be able to use those features, one must add an account first. Its only after adding an account the program is able to retrieve that information for users to view it. By selecting otherwise options before actually adding an account will trigger errors thus terminating the

program as well. In all other case, the sequential follow ups to the prompt of the program should make the program able to run without occurrences of any error. Here's a final look at the program demo.

```

1 import java.util.ArrayList; //Provides Resizable-array Implement List Interface
2 import java.util.Scanner; //Scanner Class for input
3
4 public class StudentManagement {
5
6     ArrayList<String> name = new ArrayList<String>();
7     ArrayList<String> id = new ArrayList<String>();
8     ArrayList<Double> grade = new ArrayList<Double>(); // For decimal number
9     ArrayList<Integer> balance = new ArrayList<Integer>(); // For whole number
10
11     public void addAccount() //Public Method
12     {
13         String userName="";
14         String idNum="";
15         Double userGrade=0.0;
16         int userBalance=0;
17         Scanner input= new Scanner(System.in);
18         System.out.println("To Add An Account Please Enter Your Name.");
19         if (input.hasNextLine())
20         {
21             userName=input.nextLine();
22         }
23
24         System.out.println("Enter Your ID Number:");
25         if (input.hasNextInt())
26         {
27             idNum= input.nextLine();
28         }
29         System.out.println("Enter Your GPA:");
30         Scanner input1= new Scanner(System.in);
31         userGrade= input1.nextDouble();
32
33         System.out.println("Enter Your Balance Due:");
34         Scanner input2= new Scanner(System.in);
35         userBalance = input2.nextInt();
36
37     }
38
39 }

```

```

StudentManagement [Java Application] C:\Program Files\Java\jdk-
Hello Student, What Would You Like To Do?
1. Add Account.
2. View Account.
3. Delete Account.
4. View Balance Due.
5. Exit.
1
To Add An Account Please Enter Your Name.
Sachin Karki
Enter Your ID Number:
12585
Enter Your GPA:
3.78
Enter Your Balance Due:
2100
Hello User, What Would You Like To Do?
1. Add Account.
2. View Account.
3. Delete Account.
4. View Balance Due.
5. Exit
2
Enter Your ID Number
12585
---Student Information Summary---
NAME: Sachin Karki
ID: 12585
GRADE: 3.78
BALANCE: 2100
Hello User, What Would You Like To Do?
1. Add Account.
2. View Account.
3. Delete Account.
4. View Balance Due.
5. Exit

```

### Add & View Account

### Delete Account

```

120         myMain.id.remove(a2);
121         System.out.println("Succesfully Deleted.");
122     }
123     break;
124 case 4:
125     System.out.println("Enter The ID You Would Like The Balance For:");
126     Scanner input4=new Scanner(System.in);
127     myID=input4.nextLine();
128     int a3= myMain.viewAccount(myID);
129     if(a3!=-1)
130     {
131         System.out.println("Your Balance Due Is: " + myMain.balance.get(a3));
132     }
133     break;
134 default:
135     System.out.println("Please Enter A Valid Number.");
136     break;
137
138 }
139 System.out.println("Hello User, What Would You Like To Do?");
140 System.out.println("1. Add Account.");
141 System.out.println("2. View Account.");
142 System.out.println("3. Delete Account.");
143 System.out.println("4. View Balance Due.");
144 System.out.println("5. Exit");
145 Scanner newScan= new Scanner(System.in);
146 if (newScan.hasNextInt())
147 {
148     userChoice = newScan.nextInt();
149 }
150 //userChoice=newScan.nextInt();
151 }while(userChoice!=5);
152 System.out.println("---EXITED---");
153
154 }
155
156 }
157
158 }

```

```

<terminated> StudentManagement [Java Application] C:\Program
Enter Your ID Number
12585
---Student Information Summary---
NAME: Sachin Karki
ID: 12585
GRADE: 3.78
BALANCE: 2100
Hello User, What Would You Like To Do?
1. Add Account.
2. View Account.
3. Delete Account.
4. View Balance Due.
5. Exit
4
Enter The ID You Would Like The Balance For:
12585
Your Balance Due Is: 2100
Hello User, What Would You Like To Do?
1. Add Account.
2. View Account.
3. Delete Account.
4. View Balance Due.
5. Exit
3
Enter the ID You Would like to delete:
12585
Succesfully Deleted.
Hello User, What Would You Like To Do?
1. Add Account.
2. View Account.
3. Delete Account.
4. View Balance Due.
5. Exit
5
---EXITED---

```