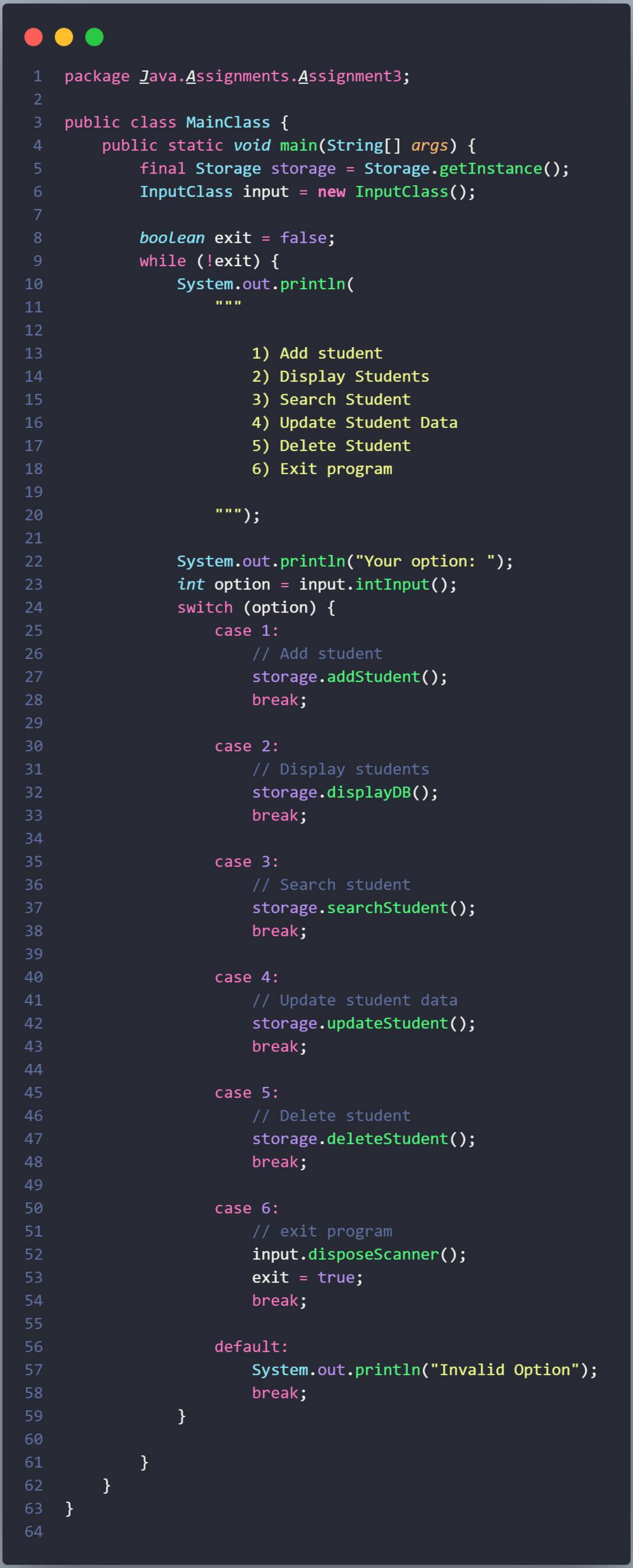
Name: Dev Kamlesh Bhanushali

PRN: 22070126032

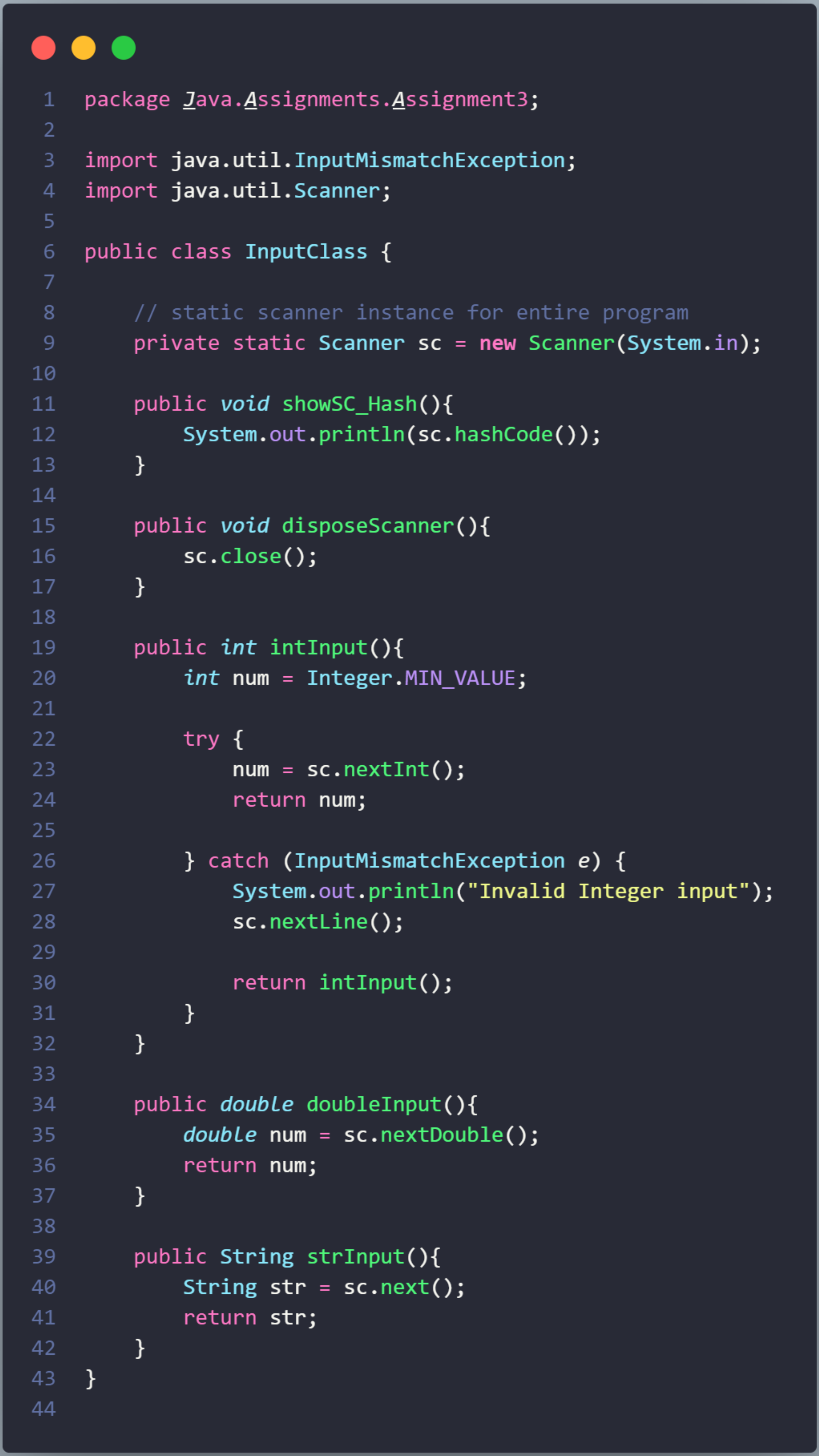
Java Assignment 3

Student Management System

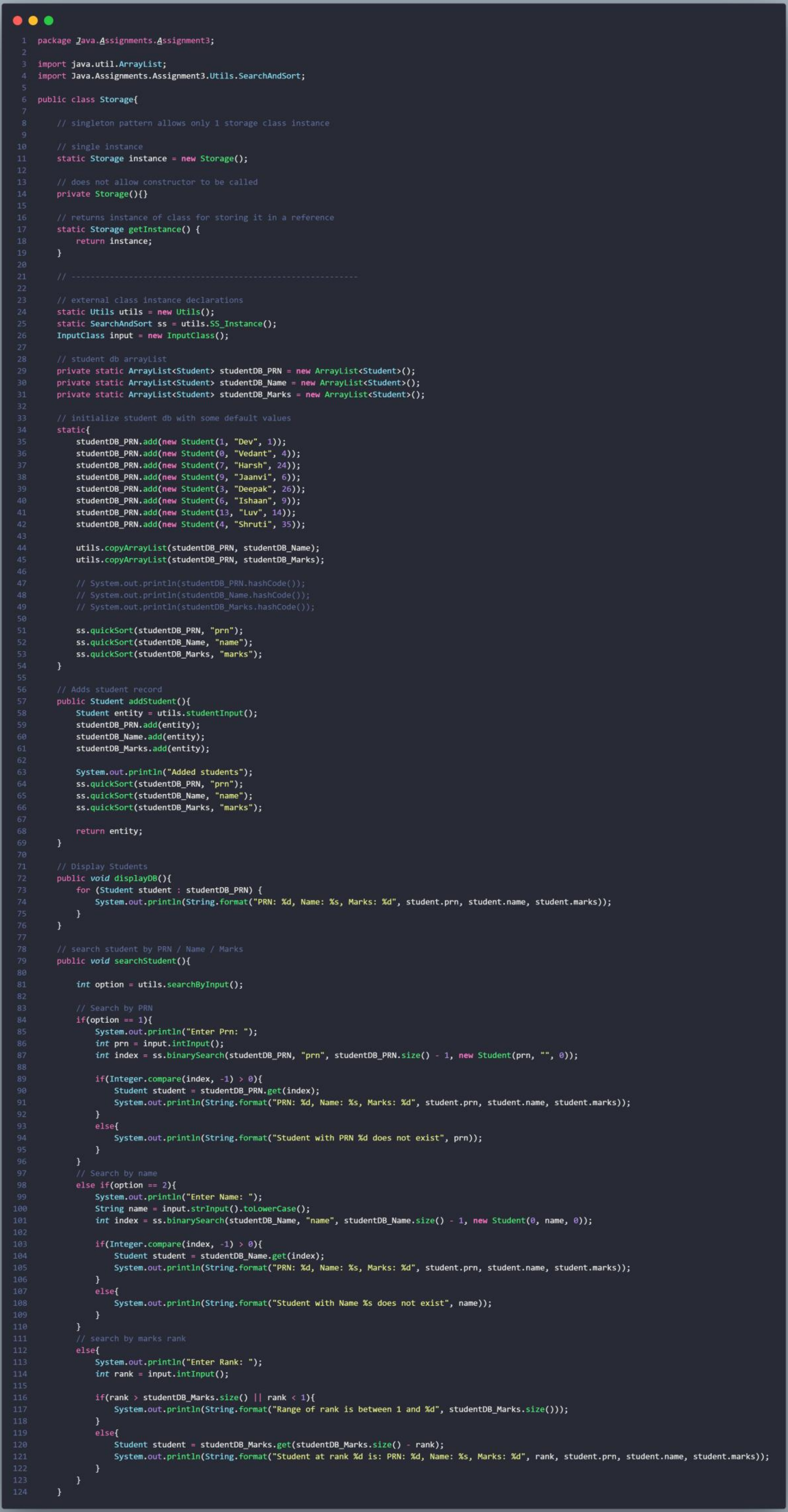
Github: https://github.com/devilb2103/Sem-4/tree/main/Java/Assignments/Assignment3



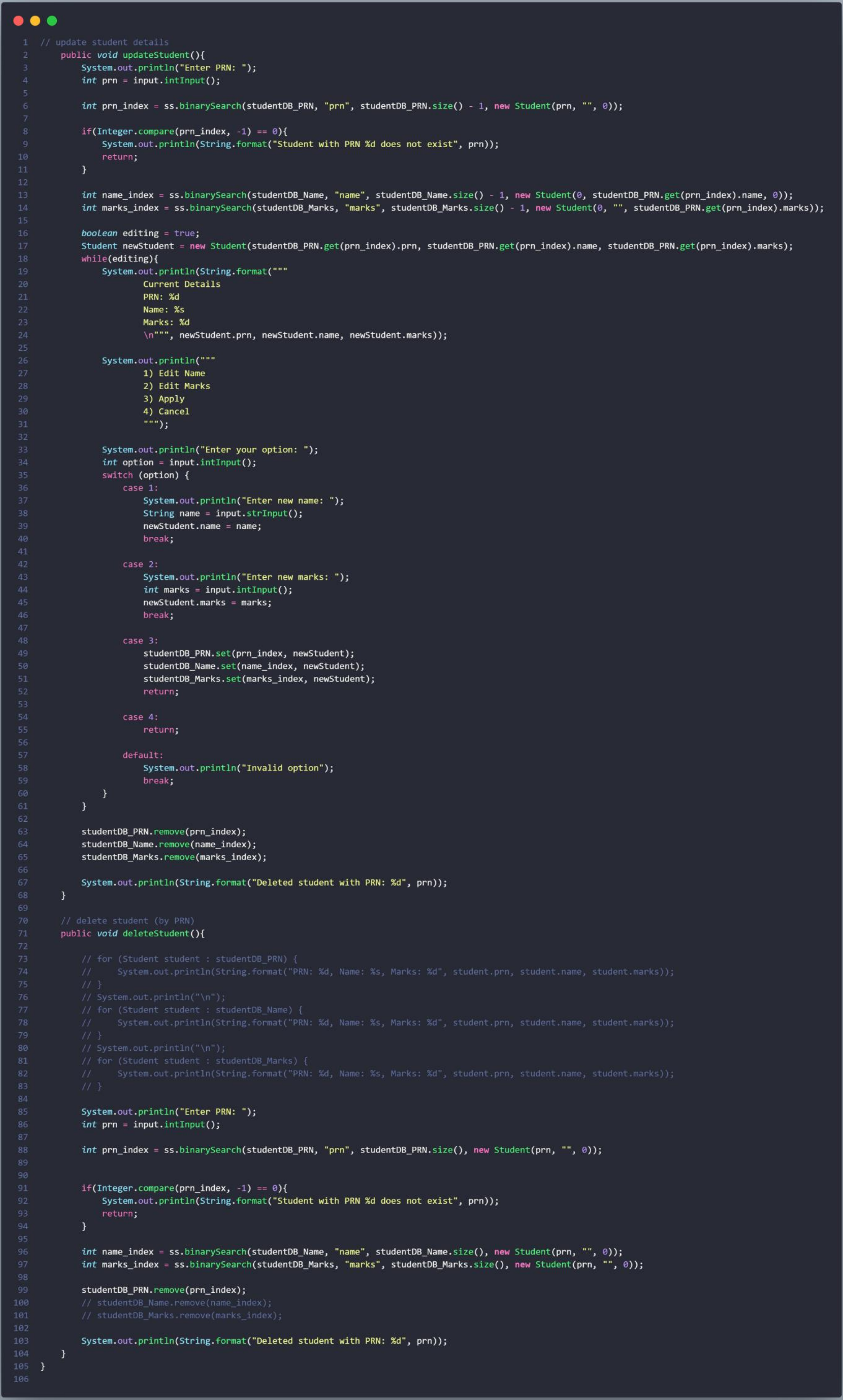
```java
package Java.Assignments.Assignment3;

public class MainClass {
    public static void main(String[] args) {
        final Storage storage = Storage.getInstance();
        InputClass input = new InputClass();

        boolean exit = false;
        while (!exit) {
            System.out.println(
                """

                    1) Add student
                    2) Display Students
                    3) Search Student
                    4) Update Student Data
                    5) Delete Student
                    6) Exit program

                """);

            System.out.println("Your option: ");
            int option = input.intInput();
            switch (option) {
                case 1:
                    // Add student
                    storage.addStudent();
                    break;

                case 2:
                    // Display students
                    storage.displayDB();
                    break;

                case 3:
                    // Search student
                    storage.searchStudent();
                    break;

                case 4:
                    // Update student data
                    storage.updateStudent();
                    break;

                case 5:
                    // Delete student
                    storage.deleteStudent();
                    break;

                case 6:
                    // exit program
                    input.disposeScanner();
                    exit = true;
                    break;

                default:
                    System.out.println("Invalid Option");
                    break;
            }

        }
    }
}

```



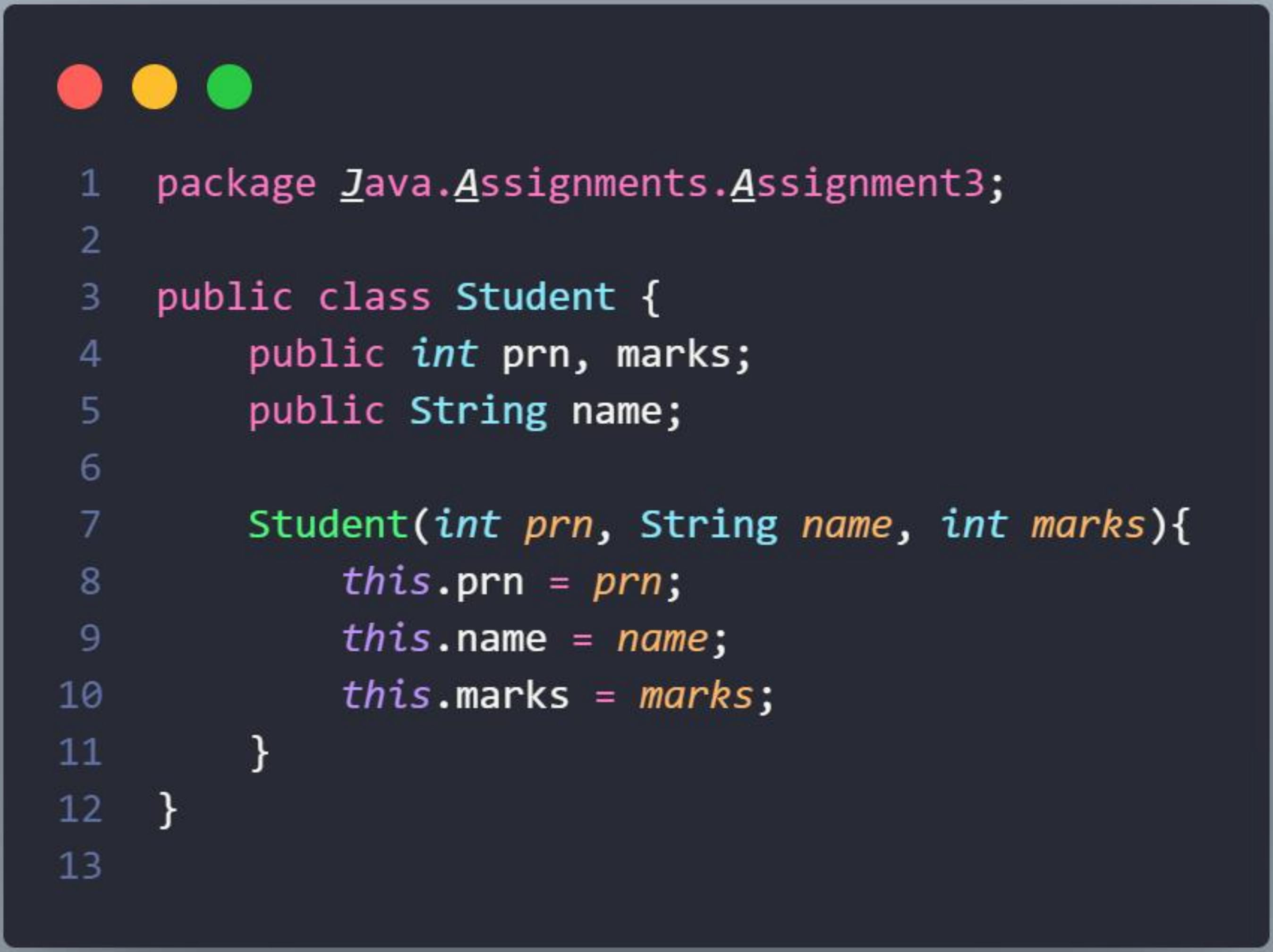
```java
package Java.Assignments.Assignment3;

import java.util.InputMismatchException;
import java.util.Scanner;

public class InputClass {

    // static scanner instance for entire program
    private static Scanner sc = new Scanner(System.in);

    public void showSC_Hash(){
        System.out.println(sc.hashCode());
    }

    public void disposeScanner(){
        sc.close();
    }

    public int intInput(){
        int num = Integer.MIN_VALUE;

        try {
            num = sc.nextInt();
            return num;

        } catch (InputMismatchException e) {
            System.out.println("Invalid Integer input");
            sc.nextLine();

            return intInput();
        }
    }

    public double doubleInput(){
        double num = sc.nextDouble();
        return num;
    }

    public String strInput(){
        String str = sc.next();
        return str;
    }
}

```



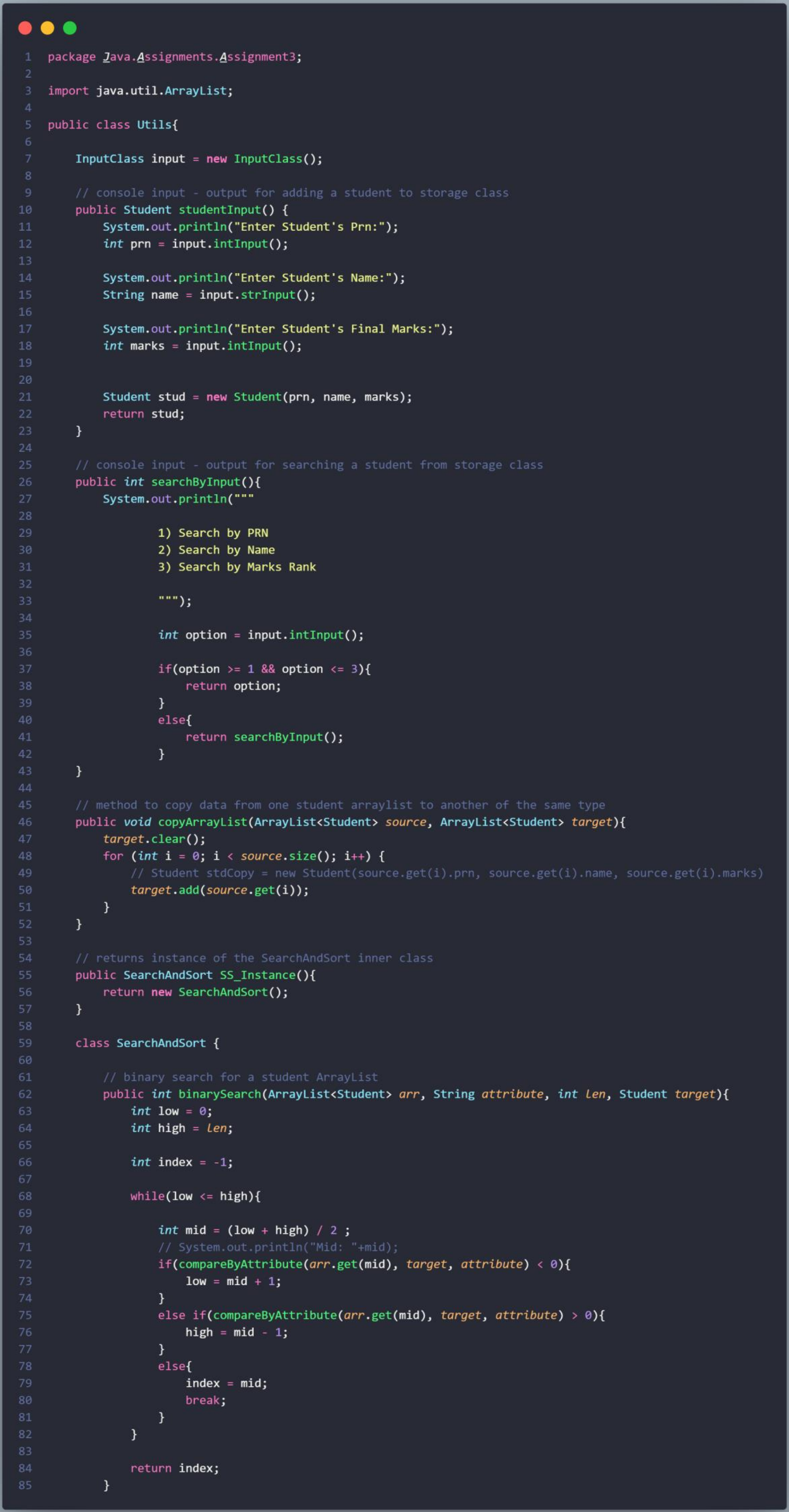
```java
package Java.Assignments.Assignment3;

import java.util.ArrayList;
import Java.Assignments.Assignment3.Utils.SearchAndSort;

public class Storage{

    // singleton pattern allows only 1 storage class instance

    // single instance
    static Storage instance = new Storage();

    // does not allow constructor to be called
    private Storage(){}

    // returns instance of class for storing it in a reference
    static Storage getInstance() {
        return instance;
    }

    // ------------------------------------------------------------

    // external class instance declarations
    static Utils utils = new Utils();
    static SearchAndSort ss = utils.SS_Instance();
    InputClass input = new InputClass();

    // student db arrayList
    private static ArrayList<Student> studentDB_PRN = new ArrayList<Student>();
    private static ArrayList<Student> studentDB_Name = new ArrayList<Student>();
    private static ArrayList<Student> studentDB_Marks = new ArrayList<Student>();

    // initialize student db with some default values
    static{
        studentDB_PRN.add(new Student(1, "Dev", 1));
        studentDB_PRN.add(new Student(0, "Vedant", 4));
        studentDB_PRN.add(new Student(7, "Harsh", 24));
        studentDB_PRN.add(new Student(9, "Jaanvi", 6));
        studentDB_PRN.add(new Student(3, "Deepak", 26));
        studentDB_PRN.add(new Student(6, "Ishaan", 9));
        studentDB_PRN.add(new Student(13, "Luv", 14));
        studentDB_PRN.add(new Student(4, "Shruti", 35));

        utils.copyArrayList(studentDB_PRN, studentDB_Name);
        utils.copyArrayList(studentDB_PRN, studentDB_Marks);

        // System.out.println(studentDB_PRN.hashCode());
        // System.out.println(studentDB_Name.hashCode());
        // System.out.println(studentDB_Marks.hashCode());

        ss.quickSort(studentDB_PRN, "prn");
        ss.quickSort(studentDB_Name, "name");
        ss.quickSort(studentDB_Marks, "marks");
    }

    // Adds student record
    public Student addStudent(){
        Student entity = utils.studentInput();
        studentDB_PRN.add(entity);
        studentDB_Name.add(entity);
        studentDB_Marks.add(entity);

        System.out.println("Added students");
        ss.quickSort(studentDB_PRN, "prn");
        ss.quickSort(studentDB_Name, "name");
        ss.quickSort(studentDB_Marks, "marks");

        return entity;
    }

    // Display Students
    public void displayDB(){
        for (Student student : studentDB_PRN) {
            System.out.println(String.format("PRN: %d, Name: %s, Marks: %d", student.prn, student.name, student.marks));
        }
    }

    // search student by PRN / Name / Marks
    public void searchStudent(){

        int option = utils.searchByInput();

        // Search by PRN
        if(option == 1){
            System.out.println("Enter Prn: ");
            int prn = input.intInput();
            int index = ss.binarySearch(studentDB_PRN, "prn", studentDB_PRN.size() - 1, new Student(prn, "", 0));

            if(Integer.compare(index, -1) > 0){
                Student student = studentDB_PRN.get(index);
                System.out.println(String.format("PRN: %d, Name: %s, Marks: %d", student.prn, student.name, student.marks));
            }
            else{
                System.out.println(String.format("Student with PRN %d does not exist", prn));
            }
        }
        // Search by name
        else if(option == 2){
            System.out.println("Enter Name: ");
            String name = input.strInput().toLowerCase();
            int index = ss.binarySearch(studentDB_Name, "name", studentDB_Name.size() - 1, new Student(0, name, 0));

            if(Integer.compare(index, -1) > 0){
                Student student = studentDB_Name.get(index);
                System.out.println(String.format("PRN: %d, Name: %s, Marks: %d", student.prn, student.name, student.marks));
            }
            else{
                System.out.println(String.format("Student with Name %s does not exist", name));
            }
        }
        // search by marks rank
        else{
            System.out.println("Enter Rank: ");
            int rank = input.intInput();

            if(rank > studentDB_Marks.size() || rank < 1){
                System.out.println(String.format("Range of rank is between 1 and %d", studentDB_Marks.size()));
            }
            else{
                Student student = studentDB_Marks.get(studentDB_Marks.size() - rank);
                System.out.println(String.format("Student at rank %d is: PRN: %d, Name: %s, Marks: %d", rank, student.prn, student.name, student.marks));
            }
        }
    }
```



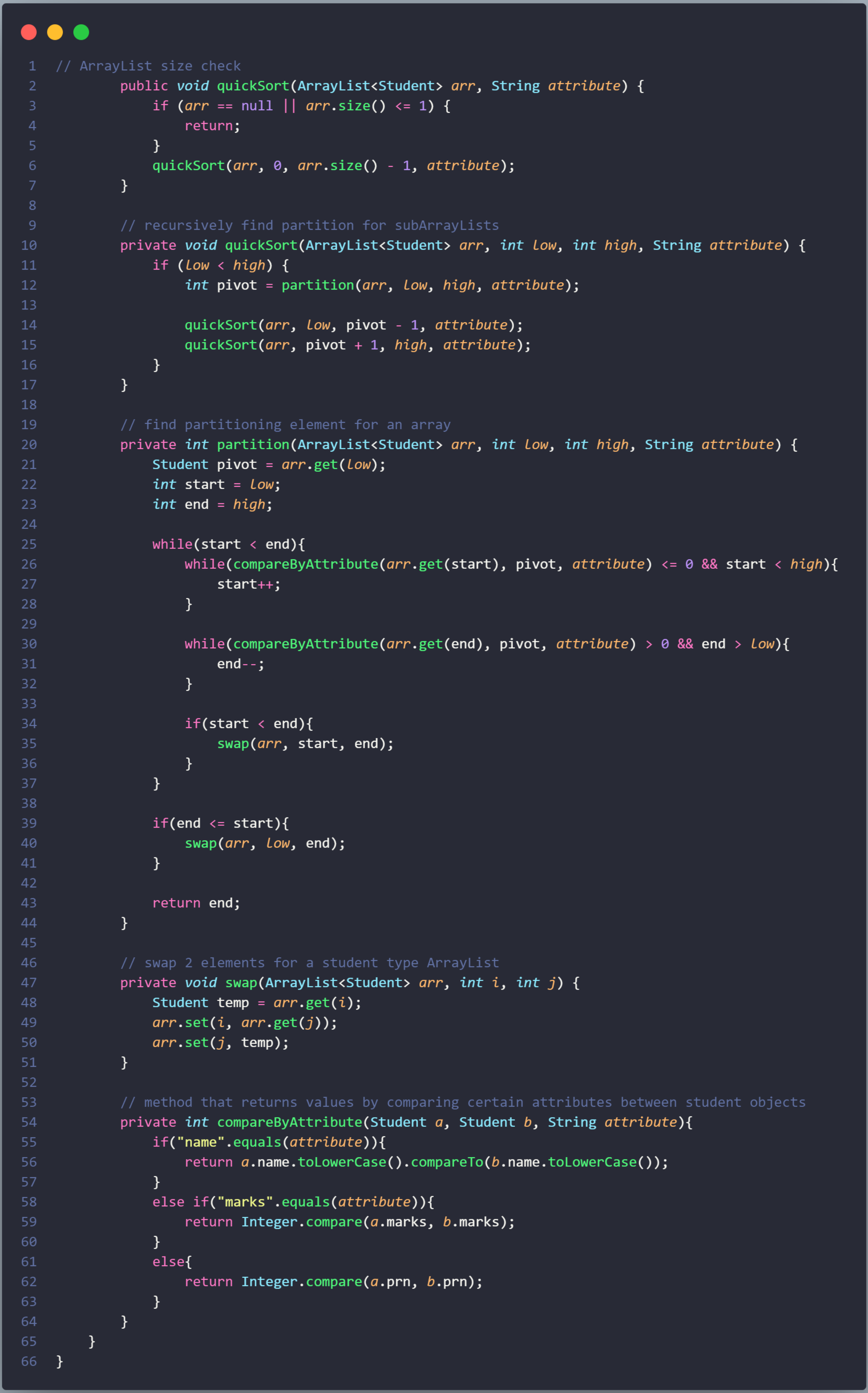
```java
// update student details
    public void updateStudent(){
        System.out.println("Enter PRN: ");
        int prn = input.intInput();

        int prn_index = ss.binarySearch(studentDB_PRN, "prn", studentDB_PRN.size() - 1, new Student(prn, "", 0));

        if(Integer.compare(prn_index, -1) == 0){
            System.out.println(String.format("Student with PRN %d does not exist", prn));
            return;
        }

        int name_index = ss.binarySearch(studentDB_Name, "name", studentDB_Name.size() - 1, new Student(0, studentDB_PRN.get(prn_index).name, 0));
        int marks_index = ss.binarySearch(studentDB_Marks, "marks", studentDB_Marks.size() - 1, new Student(0, "", studentDB_PRN.get(prn_index).marks));

        boolean editing = true;
        Student newStudent = new Student(studentDB_PRN.get(prn_index).prn, studentDB_PRN.get(prn_index).name, studentDB_PRN.get(prn_index).marks);
        while(editing){
            System.out.println(String.format("""
                    Current Details
                    PRN: %d
                    Name: %s
                    Marks: %d
                    \n""", newStudent.prn, newStudent.name, newStudent.marks));

            System.out.println("""
                    1) Edit Name
                    2) Edit Marks
                    3) Apply
                    4) Cancel
                    """);

            System.out.println("Enter your option: ");
            int option = input.intInput();
            switch (option) {
                case 1:
                    System.out.println("Enter new name: ");
                    String name = input.strInput();
                    newStudent.name = name;
                    break;

                case 2:
                    System.out.println("Enter new marks: ");
                    int marks = input.intInput();
                    newStudent.marks = marks;
                    break;

                case 3:
                    studentDB_PRN.set(prn_index, newStudent);
                    studentDB_Name.set(name_index, newStudent);
                    studentDB_Marks.set(marks_index, newStudent);
                    return;

                case 4:
                    return;

                default:
                    System.out.println("Invalid option");
                    break;
            }
        }

        studentDB_PRN.remove(prn_index);
        studentDB_Name.remove(name_index);
        studentDB_Marks.remove(marks_index);

        System.out.println(String.format("Deleted student with PRN: %d", prn));
    }

    // delete student (by PRN)
    public void deleteStudent(){

        // for (Student student : studentDB_PRN) {
        //     System.out.println(String.format("PRN: %d, Name: %s, Marks: %d", student.prn, student.name, student.marks));
        // }
        // System.out.println("\n");
        // for (Student student : studentDB_Name) {
        //     System.out.println(String.format("PRN: %d, Name: %s, Marks: %d", student.prn, student.name, student.marks));
        // }
        // System.out.println("\n");
        // for (Student student : studentDB_Marks) {
        //     System.out.println(String.format("PRN: %d, Name: %s, Marks: %d", student.prn, student.name, student.marks));
        // }

        System.out.println("Enter PRN: ");
        int prn = input.intInput();

        int prn_index = ss.binarySearch(studentDB_PRN, "prn", studentDB_PRN.size(), new Student(prn, "", 0));


        if(Integer.compare(prn_index, -1) == 0){
            System.out.println(String.format("Student with PRN %d does not exist", prn));
            return;
        }

        int name_index = ss.binarySearch(studentDB_Name, "name", studentDB_Name.size(), new Student(prn, "", 0));
        int marks_index = ss.binarySearch(studentDB_Marks, "marks", studentDB_Marks.size(), new Student(prn, "", 0));

        studentDB_PRN.remove(prn_index);
        // studentDB_Name.remove(name_index);
        // studentDB_Marks.remove(marks_index);

        System.out.println(String.format("Deleted student with PRN: %d", prn));
    }
}
```



```java
package Java.Assignments.Assignment3;

public class Student {
    public int prn, marks;
    public String name;

    Student(int prn, String name, int marks){
        this.prn = prn;
        this.name = name;
        this.marks = marks;
    }
}

```

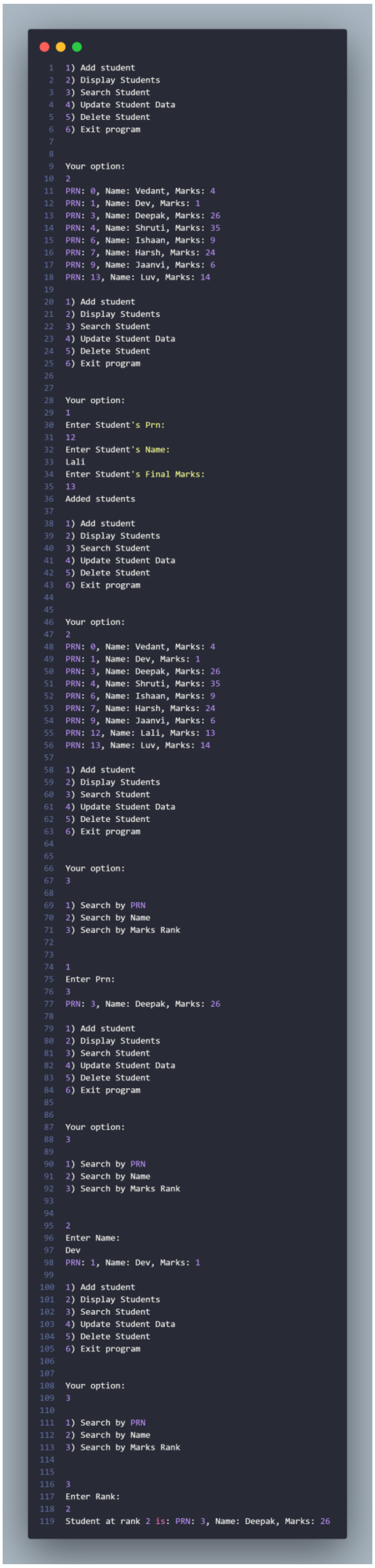


```java
package Java.Assignments.Assignment3;

import java.util.ArrayList;

public class Utils{

    InputClass input = new InputClass();

    // console input - output for adding a student to storage class
    public Student studentInput() {
        System.out.println("Enter Student's Prn:");
        int prn = input.intInput();

        System.out.println("Enter Student's Name:");
        String name = input.strInput();

        System.out.println("Enter Student's Final Marks:");
        int marks = input.intInput();


        Student stud = new Student(prn, name, marks);
        return stud;
    }

    // console input - output for searching a student from storage class
    public int searchByInput(){
        System.out.println("""

                1) Search by PRN
                2) Search by Name
                3) Search by Marks Rank

                """);

                int option = input.intInput();

                if(option >= 1 && option <= 3){
                    return option;
                }
                else{
                    return searchByInput();
                }
    }

    // method to copy data from one student arraylist to another of the same type
    public void copyArrayList(ArrayList<Student> source, ArrayList<Student> target){
        target.clear();
        for (int i = 0; i < source.size(); i++) {
            // Student stdCopy = new Student(source.get(i).prn, source.get(i).name, source.get(i).marks)
            target.add(source.get(i));
        }
    }

    // returns instance of the SearchAndSort inner class
    public SearchAndSort SS_Instance(){
        return new SearchAndSort();
    }

    class SearchAndSort {

        // binary search for a student ArrayList
        public int binarySearch(ArrayList<Student> arr, String attribute, int len, Student target){
            int low = 0;
            int high = len;

            int index = -1;

            while(low <= high){

                int mid = (low + high) / 2 ;
                // System.out.println("Mid: "+mid);
                if(compareByAttribute(arr.get(mid), target, attribute) < 0){
                    low = mid + 1;
                }
                else if(compareByAttribute(arr.get(mid), target, attribute) > 0){
                    high = mid - 1;
                }
                else{
                    index = mid;
                    break;
                }
            }

            return index;
        }
```

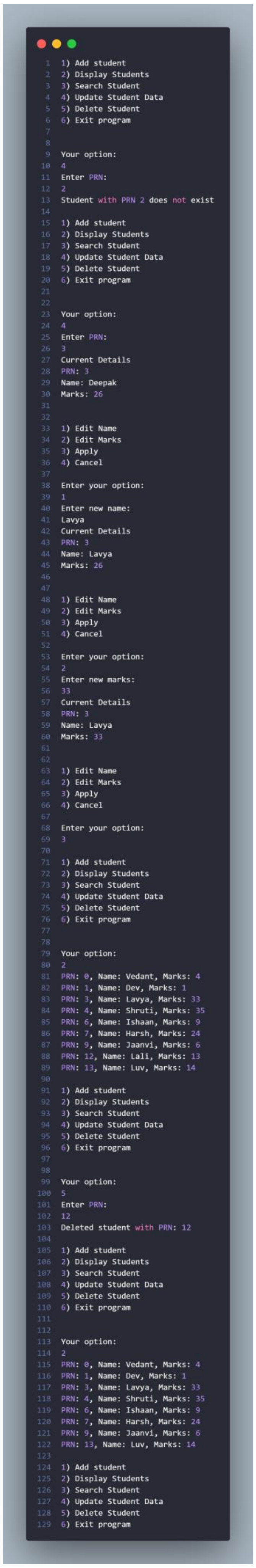


```java
// ArrayList size check
        public void quickSort(ArrayList<Student> arr, String attribute) {
            if (arr == null || arr.size() <= 1) {
                return;
            }
            quickSort(arr, 0, arr.size() - 1, attribute);
        }

        // recursively find partition for subArrayLists
        private void quickSort(ArrayList<Student> arr, int low, int high, String attribute) {
            if (low < high) {
                int pivot = partition(arr, low, high, attribute);

                quickSort(arr, low, pivot - 1, attribute);
                quickSort(arr, pivot + 1, high, attribute);
            }
        }

        // find partitioning element for an array
        private int partition(ArrayList<Student> arr, int low, int high, String attribute) {
            Student pivot = arr.get(low);
            int start = low;
            int end = high;

            while(start < end){
                while(compareByAttribute(arr.get(start), pivot, attribute) <= 0 && start < high){
                    start++;
                }

                while(compareByAttribute(arr.get(end), pivot, attribute) > 0 && end > low){
                    end--;
                }

                if(start < end){
                    swap(arr, start, end);
                }
            }

            if(end <= start){
                swap(arr, low, end);
            }

            return end;
        }

        // swap 2 elements for a student type ArrayList
        private void swap(ArrayList<Student> arr, int i, int j) {
            Student temp = arr.get(i);
            arr.set(i, arr.get(j));
            arr.set(j, temp);
        }

        // method that returns values by comparing certain attributes between student objects
        private int compareByAttribute(Student a, Student b, String attribute){
            if("name".equals(attribute)){
                return a.name.toLowerCase().compareTo(b.name.toLowerCase());
            }
            else if("marks".equals(attribute)){
                return Integer.compare(a.marks, b.marks);
            }
            else{
                return Integer.compare(a.prn, b.prn);
            }
        }
    }
}
```

Output



```
1) Add student
2) Display Students
3) Search Student
4) Update Student Data
5) Delete Student
6) Exit program


Your option:
2
PRN: 0, Name: Vedant, Marks: 4
PRN: 1, Name: Dev, Marks: 1
PRN: 3, Name: Deepak, Marks: 26
PRN: 4, Name: Shruti, Marks: 35
PRN: 6, Name: Ishaan, Marks: 9
PRN: 7, Name: Harsh, Marks: 24
PRN: 9, Name: Jaanvi, Marks: 6
PRN: 13, Name: Luv, Marks: 14

1) Add student
2) Display Students
3) Search Student
4) Update Student Data
5) Delete Student
6) Exit program


Your option:
1
Enter Student's Prn:
12
Enter Student's Name:
Lali
Enter Student's Final Marks:
13
Added students

1) Add student
2) Display Students
3) Search Student
4) Update Student Data
5) Delete Student
6) Exit program


Your option:
2
PRN: 0, Name: Vedant, Marks: 4
PRN: 1, Name: Dev, Marks: 1
PRN: 3, Name: Deepak, Marks: 26
PRN: 4, Name: Shruti, Marks: 35
PRN: 6, Name: Ishaan, Marks: 9
PRN: 7, Name: Harsh, Marks: 24
PRN: 9, Name: Jaanvi, Marks: 6
PRN: 12, Name: Lali, Marks: 13
PRN: 13, Name: Luv, Marks: 14

1) Add student
2) Display Students
3) Search Student
4) Update Student Data
5) Delete Student
6) Exit program


Your option:
3

1) Search by PRN
2) Search by Name
3) Search by Marks Rank


1
Enter Prn:
3
PRN: 3, Name: Deepak, Marks: 26

1) Add student
2) Display Students
3) Search Student
4) Update Student Data
5) Delete Student
6) Exit program


Your option:
3

1) Search by PRN
2) Search by Name
3) Search by Marks Rank


2
Enter Name:
Dev
PRN: 1, Name: Dev, Marks: 1

1) Add student
2) Display Students
3) Search Student
4) Update Student Data
5) Delete Student
6) Exit program


Your option:
3

1) Search by PRN
2) Search by Name
3) Search by Marks Rank


3
Enter Rank:
2
Student at rank 2 is: PRN: 3, Name: Deepak, Marks: 26
```



```
1) Add student
2) Display Students
3) Search Student
4) Update Student Data
5) Delete Student
6) Exit program


Your option:
4
Enter PRN:
2
Student with PRN 2 does not exist

1) Add student
2) Display Students
3) Search Student
4) Update Student Data
5) Delete Student
6) Exit program


Your option:
4
Enter PRN:
3
Current Details
PRN: 3
Name: Deepak
Marks: 26


1) Edit Name
2) Edit Marks
3) Apply
4) Cancel

Enter your option:
1
Enter new name:
Lavya
Current Details
PRN: 3
Name: Lavya
Marks: 26


1) Edit Name
2) Edit Marks
3) Apply
4) Cancel

Enter your option:
2
Enter new marks:
33
Current Details
PRN: 3
Name: Lavya
Marks: 33


1) Edit Name
2) Edit Marks
3) Apply
4) Cancel

Enter your option:
3

1) Add student
2) Display Students
3) Search Student
4) Update Student Data
5) Delete Student
6) Exit program


Your option:
2
PRN: 0, Name: Vedant, Marks: 4
PRN: 1, Name: Dev, Marks: 1
PRN: 3, Name: Lavya, Marks: 33
PRN: 4, Name: Shruti, Marks: 35
PRN: 6, Name: Ishaan, Marks: 9
PRN: 7, Name: Harsh, Marks: 24
PRN: 9, Name: Jaanvi, Marks: 6
PRN: 12, Name: Lali, Marks: 13
PRN: 13, Name: Luv, Marks: 14

1) Add student
2) Display Students
3) Search Student
4) Update Student Data
5) Delete Student
6) Exit program


Your option:
5
Enter PRN:
12
Deleted student with PRN: 12

1) Add student
2) Display Students
3) Search Student
4) Update Student Data
5) Delete Student
6) Exit program


Your option:
2
PRN: 0, Name: Vedant, Marks: 4
PRN: 1, Name: Dev, Marks: 1
PRN: 3, Name: Lavya, Marks: 33
PRN: 4, Name: Shruti, Marks: 35
PRN: 6, Name: Ishaan, Marks: 9
PRN: 7, Name: Harsh, Marks: 24
PRN: 9, Name: Jaanvi, Marks: 6
PRN: 13, Name: Luv, Marks: 14

1) Add student
2) Display Students
3) Search Student
4) Update Student Data
5) Delete Student
6) Exit program
```

Github: https://github.com/devilb2103/Sem-4/tree/main/Java/Assignments/Assignment3