

Name: Dev Kamlesh Bhanushali

PRN: 22070126032

Java Assignment 3

Student Management System

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

```
1 package Java.Assignments.Assignment3;
2
3 public class MainClass {
4     public static void main(String[] args) {
5         final Storage storage = Storage.getInstance();
6         InputClass input = new InputClass();
7
8         boolean exit = false;
9         while (!exit) {
10             System.out.println(
11                 ""
12
13                 1) Add student
14                 2) Display Students
15                 3) Search Student
16                 4) Update Student Data
17                 5) Delete Student
18                 6) Exit program
19
20                 "");
21
22             System.out.println("Your option: ");
23             int option = input.intInput();
24             switch (option) {
25                 case 1:
26                     // Add student
27                     storage.addStudent();
28                     break;
29
30                 case 2:
31                     // Display students
32                     storage.displayDB();
33                     break;
34
35                 case 3:
36                     // Search student
37                     storage.searchStudent();
38                     break;
39
40                 case 4:
41                     // Update student data
42                     storage.updateStudent();
43                     break;
44
45                 case 5:
46                     // Delete student
47                     storage.deleteStudent();
48                     break;
49
50                 case 6:
51                     // exit program
52                     input.disposeScanner();
53                     exit = true;
54                     break;
55
56                 default:
57                     System.out.println("Invalid Option");
58                     break;
59             }
60
61         }
62     }
63 }
64
```



```
1  package Java.Assignments.Assignment3;
2
3  import java.util.InputMismatchException;
4  import java.util.Scanner;
5
6  public class InputClass {
7
8      // static scanner instance for entire program
9      private static Scanner sc = new Scanner(System.in);
10
11     public void showSC_Hash(){
12         System.out.println(sc.hashCode());
13     }
14
15     public void disposeScanner(){
16         sc.close();
17     }
18
19     public int intInput(){
20         int num = Integer.MIN_VALUE;
21
22         try {
23             num = sc.nextInt();
24             return num;
25
26         } catch (InputMismatchException e) {
27             System.out.println("Invalid Integer input");
28             sc.nextLine();
29
30             return intInput();
31         }
32     }
33
34     public double doubleInput(){
35         double num = sc.nextDouble();
36         return num;
37     }
38
39     public String strInput(){
40         String str = sc.next();
41         return str;
42     }
43 }
44
```

```

1 package Java.Assignments.Assignment3;
2
3 import java.util.ArrayList;
4 import Java.Assignments.Assignment3.Utils.SearchAndSort;
5
6 public class Storage{
7
8     // singleton pattern allows only 1 storage class instance
9
10    // single instance
11    static Storage instance = new Storage();
12
13    // does not allow constructor to be called
14    private Storage(){}
15
16    // returns instance of class for storing it in a reference
17    static Storage getInstance() {
18        return instance;
19    }
20
21    // -----
22
23    // external class instance declarations
24    static Utils utils = new Utils();
25    static SearchAndSort ss = utils.ss_Instance();
26    InputClass input = new InputClass();
27
28    // student db arrayList
29    private static ArrayList<Student> studentDB_PRN = new ArrayList<Student>();
30    private static ArrayList<Student> studentDB_Name = new ArrayList<Student>();
31    private static ArrayList<Student> studentDB_Marks = new ArrayList<Student>();
32
33    // initialize student db with some default values
34    static{
35        studentDB_PRN.add(new Student(1, "Dev", 1));
36        studentDB_PRN.add(new Student(0, "Vedant", 4));
37        studentDB_PRN.add(new Student(7, "Harsh", 24));
38        studentDB_PRN.add(new Student(9, "Jaanvi", 6));
39        studentDB_PRN.add(new Student(3, "Deepak", 26));
40        studentDB_PRN.add(new Student(6, "Ishaan", 9));
41        studentDB_PRN.add(new Student(13, "Luv", 14));
42        studentDB_PRN.add(new Student(4, "Shruti", 35));
43
44        utils.copyArrayList(studentDB_PRN, studentDB_Name);
45        utils.copyArrayList(studentDB_PRN, studentDB_Marks);
46
47        // System.out.println(studentDB_PRN.hashCode());
48        // System.out.println(studentDB_Name.hashCode());
49        // System.out.println(studentDB_Marks.hashCode());
50
51        ss.quickSort(studentDB_PRN, "prn");
52        ss.quickSort(studentDB_Name, "name");
53        ss.quickSort(studentDB_Marks, "marks");
54    }
55
56    // Adds student record
57    public Student addStudent(){
58        Student entity = utils.studentInput();
59        studentDB_PRN.add(entity);
60        studentDB_Name.add(entity);
61        studentDB_Marks.add(entity);
62
63        System.out.println("Added students");
64        ss.quickSort(studentDB_PRN, "prn");
65        ss.quickSort(studentDB_Name, "name");
66        ss.quickSort(studentDB_Marks, "marks");
67
68        return entity;
69    }
70
71    // Display Students
72    public void displayDB(){
73        for (Student student : studentDB_PRN) {
74            System.out.println(String.format("PRN: %d, Name: %s, Marks: %d", student.prn, student.name, student.marks));
75        }
76    }
77
78    // search student by PRN / Name / Marks
79    public void searchStudent(){
80
81        int option = utils.searchByInput();
82
83        // Search by PRN
84        if(option == 1){
85            System.out.println("Enter Prn: ");
86            int prn = input.intInput();
87            int index = ss.binarySearch(studentDB_PRN, "prn", studentDB_PRN.size() - 1, new Student(prn, "", 0));
88
89            if(Integer.compare(index, -1) > 0){
90                Student student = studentDB_PRN.get(index);
91                System.out.println(String.format("PRN: %d, Name: %s, Marks: %d", student.prn, student.name, student.marks));
92            }
93            else{
94                System.out.println(String.format("Student with PRN %d does not exist", prn));
95            }
96        }
97        // Search by name
98        else if(option == 2){
99            System.out.println("Enter Name: ");
100            String name = input.strInput().toLowerCase();
101            int index = ss.binarySearch(studentDB_Name, "name", studentDB_Name.size() - 1, new Student(0, name, 0));
102
103            if(Integer.compare(index, -1) > 0){
104                Student student = studentDB_Name.get(index);
105                System.out.println(String.format("PRN: %d, Name: %s, Marks: %d", student.prn, student.name, student.marks));
106            }
107            else{
108                System.out.println(String.format("Student with Name %s does not exist", name));
109            }
110        }
111        // search by marks rank
112        else{
113            System.out.println("Enter Rank: ");
114            int rank = input.intInput();
115
116            if(rank > studentDB_Marks.size() || rank < 1){
117                System.out.println(String.format("Range of rank is between 1 and %d", studentDB_Marks.size()));
118            }
119            else{
120                Student student = studentDB_Marks.get(studentDB_Marks.size() - rank);
121                System.out.println(String.format("Student at rank %d is: PRN: %d, Name: %s, Marks: %d", rank, student.prn, student.name, student.marks));
122            }
123        }
124    }

```

```

1 // update student details
2 public void updateStudent(){
3     System.out.println("Enter PRN: ");
4     int prn = input.intInput();
5
6     int prn_index = ss.binarySearch(studentDB_PRN, "prn", studentDB_PRN.size() - 1, new Student(prn, "", 0));
7
8     if(Integer.compare(prn_index, -1) == 0){
9         System.out.println(String.format("Student with PRN %d does not exist", prn));
10        return;
11    }
12
13    int name_index = ss.binarySearch(studentDB_Name, "name", studentDB_Name.size() - 1, new Student(0, studentDB_PRN.get(prn_index).name, 0));
14    int marks_index = ss.binarySearch(studentDB_Marks, "marks", studentDB_Marks.size() - 1, new Student(0, "", studentDB_PRN.get(prn_index).marks));
15
16    boolean editing = true;
17    Student newStudent = new Student(studentDB_PRN.get(prn_index).prn, studentDB_PRN.get(prn_index).name, studentDB_PRN.get(prn_index).marks);
18    while(editing){
19        System.out.println(String.format("""
20            Current Details
21            PRN: %d
22            Name: %s
23            Marks: %d
24            \n""", newStudent.prn, newStudent.name, newStudent.marks));
25
26        System.out.println("""
27            1) Edit Name
28            2) Edit Marks
29            3) Apply
30            4) Cancel
31            """);
32
33        System.out.println("Enter your option: ");
34        int option = input.intInput();
35        switch (option) {
36            case 1:
37                System.out.println("Enter new name: ");
38                String name = input.strInput();
39                newStudent.name = name;
40                break;
41
42            case 2:
43                System.out.println("Enter new marks: ");
44                int marks = input.intInput();
45                newStudent.marks = marks;
46                break;
47
48            case 3:
49                studentDB_PRN.set(prn_index, newStudent);
50                studentDB_Name.set(name_index, newStudent);
51                studentDB_Marks.set(marks_index, newStudent);
52                return;
53
54            case 4:
55                return;
56
57            default:
58                System.out.println("Invalid option");
59                break;
60        }
61    }
62
63    studentDB_PRN.remove(prn_index);
64    studentDB_Name.remove(name_index);
65    studentDB_Marks.remove(marks_index);
66
67    System.out.println(String.format("Deleted student with PRN: %d", prn));
68 }
69
70 // delete student (by PRN)
71 public void deleteStudent(){
72
73     // for (Student student : studentDB_PRN) {
74     //     System.out.println(String.format("PRN: %d, Name: %s, Marks: %d", student.prn, student.name, student.marks));
75     // }
76     // System.out.println("\n");
77     // for (Student student : studentDB_Name) {
78     //     System.out.println(String.format("PRN: %d, Name: %s, Marks: %d", student.prn, student.name, student.marks));
79     // }
80     // System.out.println("\n");
81     // for (Student student : studentDB_Marks) {
82     //     System.out.println(String.format("PRN: %d, Name: %s, Marks: %d", student.prn, student.name, student.marks));
83     // }
84
85     System.out.println("Enter PRN: ");
86     int prn = input.intInput();
87
88     int prn_index = ss.binarySearch(studentDB_PRN, "prn", studentDB_PRN.size(), new Student(prn, "", 0));
89
90
91     if(Integer.compare(prn_index, -1) == 0){
92         System.out.println(String.format("Student with PRN %d does not exist", prn));
93         return;
94     }
95
96     int name_index = ss.binarySearch(studentDB_Name, "name", studentDB_Name.size(), new Student(prn, "", 0));
97     int marks_index = ss.binarySearch(studentDB_Marks, "marks", studentDB_Marks.size(), new Student(prn, "", 0));
98
99     studentDB_PRN.remove(prn_index);
100    // studentDB_Name.remove(name_index);
101    // studentDB_Marks.remove(marks_index);
102
103    System.out.println(String.format("Deleted student with PRN: %d", prn));
104 }
105 }
106

```



```
1  package Java.Assignments.Assignment3;  
2  
3  public class Student {  
4      public int prn, marks;  
5      public String name;  
6  
7      Student(int prn, String name, int marks){  
8          this.prn = prn;  
9          this.name = name;  
10         this.marks = marks;  
11     }  
12 }  
13
```



```

1 package Java.Assignments.Assignment3;
2
3 import java.util.ArrayList;
4
5 public class Utils{
6
7     InputClass input = new InputClass();
8
9     // console input - output for adding a student to storage class
10    public Student studentInput() {
11        System.out.println("Enter Student's Prn:");
12        int prn = input.intInput();
13
14        System.out.println("Enter Student's Name:");
15        String name = input.strInput();
16
17        System.out.println("Enter Student's Final Marks:");
18        int marks = input.intInput();
19
20
21        Student stud = new Student(prn, name, marks);
22        return stud;
23    }
24
25    // console input - output for searching a student from storage class
26    public int searchByInput(){
27        System.out.println("""
28
29            1) Search by PRN
30            2) Search by Name
31            3) Search by Marks Rank
32
33            """);
34
35        int option = input.intInput();
36
37        if(option >= 1 && option <= 3){
38            return option;
39        }
40        else{
41            return searchByInput();
42        }
43    }
44
45    // method to copy data from one student arraylist to another of the same type
46    public void copyArrayList(ArrayList<Student> source, ArrayList<Student> target){
47        target.clear();
48        for (int i = 0; i < source.size(); i++) {
49            // Student stdCopy = new Student(source.get(i).prn, source.get(i).name, source.get(i).marks)
50            target.add(source.get(i));
51        }
52    }
53
54    // returns instance of the SearchAndSort inner class
55    public SearchAndSort SS_Instance(){
56        return new SearchAndSort();
57    }
58
59    class SearchAndSort {
60
61        // binary search for a student ArrayList
62        public int binarySearch(ArrayList<Student> arr, String attribute, int Len, Student target){
63            int low = 0;
64            int high = Len;
65
66            int index = -1;
67
68            while(low <= high){
69
70                int mid = (low + high) / 2 ;
71                // System.out.println("Mid: "+mid);
72                if(compareByAttribute(arr.get(mid), target, attribute) < 0){
73                    low = mid + 1;
74                }
75                else if(compareByAttribute(arr.get(mid), target, attribute) > 0){
76                    high = mid - 1;
77                }
78                else{
79                    index = mid;
80                    break;
81                }
82            }
83
84            return index;
85        }

```

```

1 // ArrayList size check
2 public void quickSort(ArrayList<Student> arr, String attribute) {
3     if (arr == null || arr.size() <= 1) {
4         return;
5     }
6     quickSort(arr, 0, arr.size() - 1, attribute);
7 }
8
9 // recursively find partition for subArrayLists
10 private void quickSort(ArrayList<Student> arr, int low, int high, String attribute) {
11     if (low < high) {
12         int pivot = partition(arr, low, high, attribute);
13
14         quickSort(arr, low, pivot - 1, attribute);
15         quickSort(arr, pivot + 1, high, attribute);
16     }
17 }
18
19 // find partitioning element for an array
20 private int partition(ArrayList<Student> arr, int low, int high, String attribute) {
21     Student pivot = arr.get(low);
22     int start = low;
23     int end = high;
24
25     while(start < end){
26         while(compareByAttribute(arr.get(start), pivot, attribute) <= 0 && start < high){
27             start++;
28         }
29
30         while(compareByAttribute(arr.get(end), pivot, attribute) > 0 && end > low){
31             end--;
32         }
33
34         if(start < end){
35             swap(arr, start, end);
36         }
37     }
38
39     if(end <= start){
40         swap(arr, low, end);
41     }
42
43     return end;
44 }
45
46 // swap 2 elements for a student type ArrayList
47 private void swap(ArrayList<Student> arr, int i, int j) {
48     Student temp = arr.get(i);
49     arr.set(i, arr.get(j));
50     arr.set(j, temp);
51 }
52
53 // method that returns values by comparing certain attributes between student objects
54 private int compareByAttribute(Student a, Student b, String attribute){
55     if("name".equals(attribute)){
56         return a.name.toLowerCase().compareTo(b.name.toLowerCase());
57     }
58     else if("marks".equals(attribute)){
59         return Integer.compare(a.marks, b.marks);
60     }
61     else{
62         return Integer.compare(a.prn, b.prn);
63     }
64 }
65 }
66 }

```


Output

```
1 1) Add student
2 2) Display Students
3 3) Search Student
4 4) Update Student Data
5 5) Delete Student
6 6) Exit program
7
8
9 Your option:
10 2
11 PRN: 0, Name: Vedant, Marks: 4
12 PRN: 1, Name: Dev, Marks: 1
13 PRN: 3, Name: Deepak, Marks: 26
14 PRN: 4, Name: Shruti, Marks: 35
15 PRN: 6, Name: Ishaan, Marks: 9
16 PRN: 7, Name: Harsh, Marks: 24
17 PRN: 9, Name: Jaanvi, Marks: 6
18 PRN: 13, Name: Luv, Marks: 14
19
20 1) Add student
21 2) Display Students
22 3) Search Student
23 4) Update Student Data
24 5) Delete Student
25 6) Exit program
26
27
28 Your option:
29 1
30 Enter Student's Prn:
31 12
32 Enter Student's Name:
33 Lali
34 Enter Student's Final Marks:
35 13
36 Added students
37
38 1) Add student
39 2) Display Students
40 3) Search Student
41 4) Update Student Data
42 5) Delete Student
43 6) Exit program
44
45
46 Your option:
47 2
48 PRN: 0, Name: Vedant, Marks: 4
49 PRN: 1, Name: Dev, Marks: 1
50 PRN: 3, Name: Deepak, Marks: 26
51 PRN: 4, Name: Shruti, Marks: 35
52 PRN: 6, Name: Ishaan, Marks: 9
53 PRN: 7, Name: Harsh, Marks: 24
54 PRN: 9, Name: Jaanvi, Marks: 6
55 PRN: 12, Name: Lali, Marks: 13
56 PRN: 13, Name: Luv, Marks: 14
57
58 1) Add student
59 2) Display Students
60 3) Search Student
61 4) Update Student Data
62 5) Delete Student
63 6) Exit program
64
65
66 Your option:
67 3
68
69 1) Search by PRN
70 2) Search by Name
71 3) Search by Marks Rank
72
73
74 1
75 Enter Prn:
76 3
77 PRN: 3, Name: Deepak, Marks: 26
78
79 1) Add student
80 2) Display Students
81 3) Search Student
82 4) Update Student Data
83 5) Delete Student
84 6) Exit program
85
86
87 Your option:
88 3
89
90 1) Search by PRN
91 2) Search by Name
92 3) Search by Marks Rank
93
94
95 2
96 Enter Name:
97 Dev
98 PRN: 1, Name: Dev, Marks: 1
99
100 1) Add student
101 2) Display Students
102 3) Search Student
103 4) Update Student Data
104 5) Delete Student
105 6) Exit program
106
107
108 Your option:
109 3
110
111 1) Search by PRN
112 2) Search by Name
113 3) Search by Marks Rank
114
115
116 3
117 Enter Rank:
118 2
119 Student at rank 2 is: PRN: 3, Name: Deepak, Marks: 26
```

```
1 1) Add student
2 2) Display Students
3 3) Search Student
4 4) Update Student Data
5 5) Delete Student
6 6) Exit program
7
8
9 Your option:
10 4
11 Enter PRN:
12 2
13 Student with PRN 2 does not exist
14
15 1) Add student
16 2) Display Students
17 3) Search Student
18 4) Update Student Data
19 5) Delete Student
20 6) Exit program
21
22
23 Your option:
24 4
25 Enter PRN:
26 3
27 Current Details
28 PRN: 3
29 Name: Deepak
30 Marks: 26
31
32
33 1) Edit Name
34 2) Edit Marks
35 3) Apply
36 4) Cancel
37
38 Enter your option:
39 1
40 Enter new name:
41 Lavya
42 Current Details
43 PRN: 3
44 Name: Lavya
45 Marks: 26
46
47
48 1) Edit Name
49 2) Edit Marks
50 3) Apply
51 4) Cancel
52
53 Enter your option:
54 2
55 Enter new marks:
56 33
57 Current Details
58 PRN: 3
59 Name: Lavya
60 Marks: 33
61
62
63 1) Edit Name
64 2) Edit Marks
65 3) Apply
66 4) Cancel
67
68 Enter your option:
69 3
70
71 1) Add student
72 2) Display Students
73 3) Search Student
74 4) Update Student Data
75 5) Delete Student
76 6) Exit program
77
78
79 Your option:
80 2
81 PRN: 0, Name: Vedant, Marks: 4
82 PRN: 1, Name: Dev, Marks: 1
83 PRN: 3, Name: Lavya, Marks: 33
84 PRN: 4, Name: Shruti, Marks: 35
85 PRN: 6, Name: Ishaan, Marks: 9
86 PRN: 7, Name: Harsh, Marks: 24
87 PRN: 9, Name: Jaanvi, Marks: 6
88 PRN: 12, Name: Lali, Marks: 13
89 PRN: 13, Name: Luv, Marks: 14
90
91 1) Add student
92 2) Display Students
93 3) Search Student
94 4) Update Student Data
95 5) Delete Student
96 6) Exit program
97
98
99 Your option:
100 5
101 Enter PRN:
102 12
103 Deleted student with PRN: 12
104
105 1) Add student
106 2) Display Students
107 3) Search Student
108 4) Update Student Data
109 5) Delete Student
110 6) Exit program
111
112
113 Your option:
114 2
115 PRN: 0, Name: Vedant, Marks: 4
116 PRN: 1, Name: Dev, Marks: 1
117 PRN: 3, Name: Lavya, Marks: 33
118 PRN: 4, Name: Shruti, Marks: 35
119 PRN: 6, Name: Ishaan, Marks: 9
120 PRN: 7, Name: Harsh, Marks: 24
121 PRN: 9, Name: Jaanvi, Marks: 6
122 PRN: 13, Name: Luv, Marks: 14
123
124 1) Add student
125 2) Display Students
126 3) Search Student
127 4) Update Student Data
128 5) Delete Student
129 6) Exit program
```

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