Student Information System

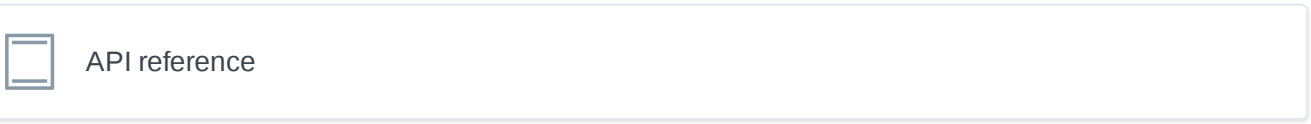
😇 Welcome!

Welcome to Student Information System

Welcome to Student Information System! Here you'll find all the documentation you need to get up and running with the Student Information System API.

Project Instruction

As a humanized student information management system, our program contains add, delete, modify, check and print student information, calculate average points output and a series of reasonable functional requirements. During the use of this program, users can experience basic information processing functions.



Details about this API Doc can be found on the website below

https://cho-2.gitbook.io/student-information-system/

API reference

Dive into the specifics of each API endpoint by checking out our complete documentation.

StudentInfo

Student information encapsulation class

StudentTool

A collection of CRUD abstract methods for this program

SystemTool

SystemToolImpl

Implementation class of abstract CRUD methods

SystemToolImpl

SystemMenu

Class for designing the program menu

SystemMenu

Main

the main class for running Student Information System

Main

> **Good to know**: Using the 'Page Link' block lets you link directly to a page. If this page's name, URL or parent location changes, the reference will be kept up to date. You can also mention a page – like Pets – if you don't want a block-level link.

StudentInfo

Student information encapsulation class

This class contains all the attributes required by the entire program, such as student ID, name, grade

and other information, and encapsulates it.

```
/**
 * @Author: Cho
 * @Date: 2022/05/12/0:30
 * @Description: Student information encapsulation class
 */
public class StudentInfo {
    private String studentId;
    private String studentName;
    private double studentMark;
    public  static StudentInfo[] studentInfos = new StudentInfo[999];
    public  static Object[][] Marks = new Object[999][2];

    public StudentInfo() {
    }

    public StudentInfo(String studentId, String studentName, double studentMark) {
        this.studentId = studentId;
        this.studentName = studentName;
        this.studentMark = studentMark;
    }

    public String getStudentId() {
        return studentId;
    }

    public void setStudentId(String studentId) {
        this.studentId = studentId;
    }

    public String getStudentName() {
        return studentName;
    }

    public void setStudentName(String studentName) {
        this.studentName = studentName;
    }

    public double getStudentMark() {
        return studentMark;
    }

    public void setStudentMark(double studentMark) {
        this.studentMark = studentMark;
    }

    @Override
    public String toString() {
        return "StudentInfo{" +
                "studentId='" + studentId + '\'' +
                ", studentName='" + studentName + '\'' +
```

```
                ", studentMark=" + studentMark +
                '}';
    }
}
```

SystemTool

A collection of CRUD abstract methods for this program

This is an interface that contains multiple abstract methods for CRUD operations.

```
import java.io.IOException;
/**
 * @Author: Cho
 * @Date: 2022/05/12/0:30
 * @Description: System tool interface
 */
public interface SystemTool {
    public void insertData();
    public void deleteData();
    public void findData();
    public void modifyData();
    public void displayData();
    public void calculateData();
    public void printData() throws IOException;
}

```

SystemToolImpl

Implementation class of abstract CRUD methods

This is an implementation class based on all the abstract methods in the SystemTool interface.

insertData

Get the keyboard input and store it in the StudentInfo object.

```
/**
     *  insert Data from user to student information array
     */
    @Override
    public void insertData() {
        StudentInfo student = new StudentInfo();
        Scanner sc = new Scanner(System.in);
        System.out.println("What is the id of student?");
```

```
        String id = sc.next();

        student.setStudentId(id);
        System.out.println("What is the name of student?");
        student.setStudentName(sc.next());
        System.out.println("What is student's mark in the final test?");
        double score = sc.nextDouble();
        student.setStudentMark(score);

        // Check the number of elements in array
        int count = checkNumberOfElement();
        // insert element to array
        studentInfos[count] = student;
        Marks[count][0] = id;
        Marks[count][1] = score;
        System.out.println("Inserting successful!");
    }
```

deleteData

Obtain the element to be deleted through the user's keyboard input, and use an algorithm to delete its data from the array.

```
/**
     *  delete Data from student information array
     */
    @Override
    public void deleteData() {
        Scanner sc = new Scanner(System.in);
        System.out.println("What is the id of student you want to delete data?");
        String id = sc.next();

        boolean check = false;
        int count = checkNumberOfElement();
        // delete the element of studentInfos
        for(int i=0; i<count; i++){
            if (studentInfos[i].getStudentId().equals(id)){
                for(int j=i; j<count; j++) {
                    studentInfos[j] = studentInfos[j + 1];
                    check = true;
                }
            }
        }
        // exception check
        if(!check)
            System.out.println("Cannot find specified id");
        else
            System.out.println("Deleting successful!");
    }
```

findData

Obtain the element to be searched through the user's keyboard input, and traverse the array to find the location of the index and the information it stores.

```
/**
     *  search Data from student information array
     */
    @Override
    public void findData() {
        Scanner sc = new Scanner(System.in);
        System.out.println("What is the id of student that the data you want to search?");
        String id = sc.next();
        int count = checkNumberOfElement();
        boolean check = false;
        // iterate array and find out the specified element
        System.out.printf("%s %8s %8s","ID","Name","Mark");
        for(int i=0; i<count; i++){
            if(studentInfos[i].getStudentId().equals(id)){
                System.out.printf("%s %8s %8s%n",studentInfos[i].getStudentId(),
                        studentInfos[i].getStudentName(),
                        String.valueOf(studentInfos[i].getStudentMark()));
            }
        }
        // exception check
        if(check){
            System.out.println("Cannot find the id");
        }
    }
```

modifyData

Obtain the element that needs to be modified through the input of the user's keyboard, and traverse the array to find the location of the index and modify the stored student information.

```
/**
     *  modify Data from student information array
     */
    @Override
    public void modifyData() {
        Scanner sc = new Scanner(System.in);
        System.out.println("What is the id of student that data you want to modify?");
        String id = sc.next();
        int count = checkNumberOfElement();
        // iterate the array and modify the element
        for(int i=0; i<count; i++){
            if (studentInfos[i].getStudentId().equals(id)){
                System.out.println("What is the id of student?");
                studentInfos[i].setStudentId(sc.next());
                System.out.println("What is the name of student?");
                studentInfos[i].setStudentName(sc.next());
```

```
                System.out.println("What is the mark of student?");
                studentInfos[i].setStudentMark(sc.nextDouble());
            }
        }
        System.out.println("Modifying successful!");
    }
```

displayData

Iterates through the array and displays the student information stored in each element.

```
/**
     *  display Data from student information array
     */
    @Override
    public void displayData() {
        int count = checkNumberOfElement();
        //using bubble sort to implement
        for(int i=0; i<count-1; i++){
            for(int j=i+1; j<count; j++){
                if(studentInfos[i].getStudentMark()>studentInfos[j].getStudentMark()){
                    StudentInfo temp = studentInfos[i];
                    studentInfos[i] = studentInfos[j];
                    studentInfos[j] = temp;
                }
            }
        }
        System.out.printf("%s %8s %8s%n","ID","Name","Mark");
        for (int k=0; k<count; k++){
            System.out.printf("%s %8s %8s%n",studentInfos[k].getStudentId(),
                    studentInfos[k].getStudentName(),
                    String.valueOf(studentInfos[k].getStudentMark()));
        }
    }
```

calculateData

Iterate through the two-dimensional array and find each student's score and calculate the average score.

```
 /**
     * calculate the average mark of students
     */
    @Override
    public void calculateData() {
        int count = checkNumberOfElement();
        double sum = 0;
        for(int i=0; i<count; i++){
            sum = sum+ Double.parseDouble(Marks[i][1].toString());
        }
```

```
        System.out.println("The average mark of students is "+Math.round(sum/count*100)/10
    }
```

printData

Iterate over a number group and output it to an external file.

```
/**
     * print the data to another file
     */
    @Override
    public void printData() throws IOException {
        int count = checkNumberOfElement();
        PrintWriter printWriter = new PrintWriter(new FileWriter("StudentInfo.txt"));
        printWriter.printf("%s %8s %8s%n","ID","Name","Mark");
        for(int i=0; i<count; i++){
            printWriter.printf("%s %8s %8s%n",studentInfos[i].getStudentId(),
                    studentInfos[i].getStudentName(),
                    String.valueOf(studentInfos[i].getStudentMark()));
        }
        printWriter.close();
        System.out.println("printing successful!");
    }
```

checkNumberOfElement

Find the number of element in array.

```
/**
     * a method for checking the number of element in array
     * @return
     */
    public int checkNumberOfElement(){
        int count = 0;
        for(int i=0; i<studentInfos.length; i++){
            if(studentInfos[i] != null)
                count++;
            else
                break;
        }
        return count;
    }
```

SystemMenu

Class for designing the program menu

This class includes the page design of the program and the connection and fusion of various functions.

welcomMenu

major method for designing the logic of interface interaction for user.

```
public void welcomeMenu() throws InterruptedException, IOException {
        SystemToolImpl tool = new SystemToolImpl();
        boolean check = false;
        String decision = "";
        Scanner sc = new Scanner(System.in);

        //Use the concept of circulation to design interactive programs
        while(!check){
            mainMenuWord();
            switch (sc.nextInt()){
                case 1:
                    tool.insertData();
                    while (true){
                        System.out.println("Press Y to continue inserting; Press N to go ba
                        decision = sc.next();
                        if(decision.equals("Y")){
                            tool.insertData();
                        } else if (decision.equals("N")) {
                            welcomeMenu();
                        }
                    }

                case 2:
                    if(studentInfos[0] == null) {
                        noParameterException();
                    }else {
                        tool.deleteData();
                        while (true){
                            System.out.println("Press Y to continue deleting; Press N to go
                            decision = sc.next();
                            if(decision.equals("Y")){
                                tool.deleteData();
                            } else if (decision.equals("N")) {
                                welcomeMenu();
                            }
                        }
                    }

                case 3:
                    if(studentInfos[0] == null) {
                        noParameterException();
                    }else {
                        tool.findData();
                        while (true){
```

```
                            System.out.println("Press Y to search another data; Press N to
                            decision = sc.next();
                            if(decision.equals("Y")){
                                tool.findData();
                            } else if (decision.equals("N")) {
                                welcomeMenu();
                            }
                        }
                    }
                case 4:
                    if(studentInfos[0] == null) {
                       noParameterException();
                    }else {
                        tool.modifyData();
                        while (true){
                            System.out.println("Press Y to modify another data; Press N to
                            decision = sc.next();
                            if(decision.equals("Y")){
                                tool.modifyData();
                            } else if (decision.equals("N")) {
                                welcomeMenu();
                            }
                        }
                    }
                case 5:
                    if(studentInfos[0] == null) {
                       noParameterException();
                    }else {
                        tool.displayData();
                        System.out.println("Press N to go back main menu.");
                        decision = sc.next();
                        if (decision.equals("N"))
                            welcomeMenu();
                    }
                case 6:
                    if(studentInfos[0] == null) {
                        noParameterException();
                    }else {
                        tool.calculateData();
                        System.out.println("Press N to go back main menu.");
                        decision = sc.next();
                        if (decision.equals("N"))
                            welcomeMenu();
                    }
                case 7:
                    if(studentInfos[0] == null) {
                        noParameterException();
                    }else {
                        tool.printData();
                        System.out.println("Press N to go back main menu.");
                        decision = sc.next();
                        if (decision.equals("N"))
                            welcomeMenu();
```

```
                    }
                case 8:
                    System.out.println("Thank you for using Student Information System");
                    System.exit(0);
                default:
                    System.out.println("Value error, please enter again");
                    Thread.sleep(1000);
            }
        }
    }
```

mainMenuWord

method for designing the interface.

```
public void mainMenuWord(){
        System.out.println("*****************************************************************
        System.out.println("Welcome to use the Student Information System.");
        System.out.println("Please follow the instructions below to hit the number on your
        System.out.println("1. insert a student's information");
        System.out.println("2. delete a student's information");
        System.out.println("3. search a student's information");
        System.out.println("4. modify a student's information");
        System.out.println("5. display all information of students(Ascending Order By Mark)
        System.out.println("6. find the average mark of students");
        System.out.println("7. print all information of students to another file");
        System.out.println("8. exit");
        System.out.println("*****************************************************************
        System.out.println();
    }
```

noParameterException

This method is used to check and determine null data exceptions.

```
public void noParameterException() throws InterruptedException, IOException {
        Scanner sc = new Scanner(System.in);
        String decision = "";
            System.out.println("There is not any recording in system, please insert the in
            System.out.println("Press N to go back main menu.");
            decision = sc.next();
            if(decision.equals("N"))
                welcomeMenu();
    }
```

Main

the main class for running Student Information System

This class includes the running of the entire program.

```
1  import java.io.IOException;
2
3  /**
4   * @Author: Cho
5   * @Date: 2022/05/12/0:30
6   * @Description: the main class of Student Information System
7   */
8  public class Main {
9      public static void main(String[] args) throws InterruptedException, IOException {
10         SystemMenu test = new SystemMenu();
11         test.welcomeMenu();
12     }
13 }
14
```