Main class

```
import java.awt.*;
import java.awt.GridLayout;
import java.awt.event.*;
import java.io.*;
import java.util.*;
import java.util.stream.Collectors;

import javax.swing.*;
import javax.swing.table.*;

public class main extends JFrame implements ActionListener {
	static ArrayList<String> list1 =new ArrayList<>();
	JMenuBar menubar;
	JMenu studentMenu,sortMenu,filterMenu;
	JMenuItem
addStudent,removeStudent,removeAll,sortLastName,sortID,sortAverage,filterGrade,fil
terRemove;
	JPanel p1,tablist,tabchart,p2,p4;
	JTabbedPane list;
	static JTable table;
	JScrollPane scrollPane;
	static String arrays[][];
	static int
counter=0,counter2=0,counter3=0,counter4=0,avgQuiz1,avgQuiz2,avgProject,avgMidterm
,avgFinal,avgAverage;
	static String []column={"NAME", "LAST NAME", "ID", "QUIZ1", "QUIZ2",
"PROJECT", "MIDTERM", "FINAL", "AVERAGE", "LETTER GRADE"};
	static String
textFilter,textAddStudent,quiz1,quiz2,Final,midterm,project,average,textRemoveById
;
	static Course  t3[][];
	static int sum1 = 0,sum2 = 0,sum3 = 0,sum4 = 0,sum5 = 0,sum6 =
0,arrayslength;
	String arrays2[][]=new String[800][10];
	public main() throws IOException{
		//we give layout to frame
		setLayout(new BorderLayout());
		//create panel,menubar and 3 menus.
		p1=new JPanel();
		menubar=new JMenuBar();
		studentMenu=new JMenu("Student");//studentMenu  text is Student.
		sortMenu=new JMenu("Sort");//sortMenu text is Sort.
		filterMenu=new JMenu("Filter");//filterMenu text is Filter.
		//I am adding menu into menubar.
		menubar.add(studentMenu);
		menubar.add(sortMenu);
		menubar.add(filterMenu);
		// i am creating  items for studentMenu.
		addStudent=new JMenuItem("Add Student");
		removeStudent=new JMenuItem("Remove Student");
		removeAll=new JMenuItem("Remove all Student");
		//I am adding item into studentMenu.
		studentMenu.add(addStudent);
		studentMenu.add(removeStudent);
		studentMenu.add(removeAll);
		// i am creating  items for sortMenu.
		sortLastName=new JMenuItem("Sort by last name");
```

```
            sortID=new JMenuItem("Sort by ID");
            sortAverage=new JMenuItem("Sort by avarage");
            //I am adding item into sortMenu.
            sortMenu.add(sortLastName);
            sortMenu.add(sortID);
            sortMenu.add(sortAverage);
            // i am creating  items for filterMenu.
            filterGrade=new JMenuItem("Filter by letter grade");
            filterRemove=new JMenuItem("Remove filter");
            //I am adding item into filterMenu.
            filterMenu.add(filterGrade);
            filterMenu.add(filterRemove);

            p1.add(menubar);//I'm adding Menubar to the panel.

            p1.setLayout(new GridLayout(0, 1));//panel row is 0,column is 1.
            add(p1,BorderLayout.NORTH);//i am adding panel to the north of frame
            //i am creating 2 panels.
            tablist=new JPanel();//first tab's panel.
            tabchart=new JPanel();//second tab's panel.

            tabchart.setLayout(new BorderLayout());//tabchart  panel layuot is
borderlayout.
            tabchart.add(new DrawChart());//I'm adding the panel to DrawChart
class. In DrawChart class, a bar chart is shown.
            p2=new JPanel();//I'm creating a new panel to add tabs.
            list=new JTabbedPane();//i am creating Tab
            tablist.setLayout(new GridLayout(0, 1));//tablist panel column is 1
and row 0
            //I add the panels to the tab and i give name.
            list.add("List", tablist);
            list.add("Chart", tabchart);

            p2.setLayout(new GridLayout(0, 1));//p2 panel columns is 1  and row
0.
            p2.add(list);// i am adding tab to panel.

            //I activate the events of the menus.
            sortLastName.addActionListener(this);
            sortID.addActionListener(this);
            sortAverage.addActionListener(this);
            filterGrade.addActionListener(this);
            filterRemove.addActionListener(this);
            addStudent.addActionListener(this);
            removeAll.addActionListener(this);
            removeStudent.addActionListener(this);

            p4=new JPanel();//i am creating new panel
            scrollPane = new JScrollPane();// i am creating scrollpane for table.
            scrollPane.setBounds(40, 11, 825, 427);

            table = new JTable();// i am creating Table.
            table.setModel(new DefaultTableModel(new String[][] {},new String[]
{}));

            getData();//I get the data in the notepad.


            scrollPane.setViewportView(table);
```

```
            //I add to the table of data I have received.
            TableModel tablemodel=new DefaultTableModel(arrays, column);
            table.setModel(tablemodel);

            tablist.add(scrollPane);//i am adding scrollpane to tablist.
            add(p2);// i am adding p2 panel to frame.
            setSize(1000,500);
            setDefaultCloseOperation(EXIT_ON_CLOSE);
            setVisible(true);
      }
      public static void getData() throws IOException {
            arrays=new String[800][10];
            Scanner scanner = new Scanner(new FileReader("student.txt"));
            String[] arr = new String[8000];
            while(scanner.hasNext()) {
                  arr=scanner.nextLine().split(" ");
            }
            scanner.close();

            for (int i = 0; i < 800; i++) {
                  for (int j = 0; j < 10; j++) {
                        list1.add(arr[j+10*i]);
                        arrays[i][j]=list1.get(j+10*i);
                  }
            }
            classAverageExams(arrays);
      }

      public static void sortLastName(){
            if(counter==0){
                  Comparator<String[]> arrayComparator = new
Comparator<String[]>() {
                        @Override
                        public int compare(String[] o1, String[] o2) {
                              return o1[1].compareTo(o2[1]);
                        }
                  };
                  Arrays.sort(arrays, arrayComparator);
                  TableModel tablemodel=new DefaultTableModel(arrays, column);
                  table.setModel(tablemodel);
                  counter=1;
            }
            else if(counter==1){
                  Comparator<String[]> arrayComparator = new
Comparator<String[]>() {
                        @Override
                        public int compare(String[] o1, String[] o2) {
                              return o1[1].compareTo(o2[1]);
                        }
                  };
                  Arrays.sort(arrays, Collections.reverseOrder(arrayComparator));
                  TableModel tablemodel=new DefaultTableModel(arrays, column);
                  table.setModel(tablemodel);
                  counter=0;
            }
      }
      public void sortID(){
            if(counter2==0){//first click is sort smallest to largest.
```

```
                    Comparator<String[]> arrayComparator = new
Comparator<String[]>() {
                          @Override
                          public int compare(String[] o1, String[] o2) {
                                return o1[2].compareTo(o2[2]);// column 2 is ID.
                          }
                    };
                    Arrays.sort(arrays, arrayComparator);//Sort smalest to largest.
                    TableModel tablemodel=new DefaultTableModel(arrays,
column);//update table
                    table.setModel(tablemodel);//update table
                    counter2=1;
              }
              else if(counter2==1){//second click is sort largest to smalest.
                    Comparator<String[]> arrayComparator = new
Comparator<String[]>() {
                          @Override
                          public int compare(String[] o1, String[] o2) {
                                return o1[2].compareTo(o2[2]);// column 2 is ID.
                          }
                    };
                    Arrays.sort(arrays,
Collections.reverseOrder(arrayComparator));// sort largest to smalest
                    TableModel tablemodel=new DefaultTableModel(arrays,
column);//update table.
                    table.setModel(tablemodel);//update table.
                    counter2=0;//the loop returns to the beginning.
              }
        }
        public void sortAverage2(){
              Double avg[][]=new Double[arrays.length][10];
              //converting string  to int .
              for (int i = 0; i < arrays.length; i++)
                    avg[i][8]=Double.parseDouble(arrays[i][8]);

              double tmp;
              String tmp2;
              boolean swapped = true;
              int g = 0;
              if(counter3==0){
                    //I sort rows by average from smallset to largest.
                    while (swapped) {
                          swapped = false;
                          g++;
                          for (int i = 0; i < arrays.length-g; i++) {
                                if(avg[i][8]>avg[i+1][8]){
                                      tmp = avg[i][8];
                                      avg[i][8] = avg[i + 1][8];
                                      avg[i + 1][8] = tmp;
                                      for (int j = 0; j < 10; j++) {
                                            tmp2 = arrays[i][j];
                                            arrays[i][j] = arrays[i + 1][j];
                                            arrays[i + 1][j] = tmp2;
                                      }
                                      swapped = true;
                                }
                          }
                    }
```

```
                    TableModel tablemodel=new DefaultTableModel(arrays,
column);//updated table
                    table.setModel(tablemodel);//updated table
                    counter3=1;
              }
              else if(counter3==1){
                    g = 0;
                    swapped = true;
                    //I sort rows by average from largest to smalest.
                    while (swapped) {
                          swapped = false;
                          g++;
                          for (int i = 0; i < arrays.length-g; i++) {
                                if(avg[i][8]<avg[i+1][8]){

                                      tmp = avg[i+1][8];
                                      avg[i+1][8] = avg[i][8];
                                      avg[i][8] = tmp;
                                      for (int j = 0; j < 10; j++) {
                                            tmp2 = arrays[i+1][j];
                                            arrays[i+1][j] = arrays[i][j];
                                            arrays[i][j] = tmp2;
                                      }
                                      swapped = true;
                                }
                          }
                    }
                    TableModel tablemodel=new DefaultTableModel(arrays,
column);//update table
                    table.setModel(tablemodel);//update table
                    counter3=0;//the loop returns to the beginning.
              }
        }
        public void filterByGrade(){
              arrayslength=arrays.length;//We keep the length of the array. We will
use it to removeFilter.
              arrays2=new String[arrayslength][10];
              //we keep the arrays.We will use it to removeFilter.
              for (int i = 0; i < arrayslength; i++) {
                    for (int j = 0; j < 10; j++)
                          arrays2[i][j]=arrays[i][j];
              }
              //s[9] is Grade Column and textFilter
        //filter array by text filter
              arrays = (String[][]) Arrays.stream(arrays).filter(s ->
s[9].equals(textFilter)).toArray(String[][]::new);
              TableModel tablemodel=new DefaultTableModel(arrays, column);//Uptade
table
              table.setModel(tablemodel);//Uptade table
              classAverageExams(arrays);//we calculate the changing class average.
        }
        public void filterRemove(){
              arrays=new String[arrayslength][10];//we write the length of the
array before filtering.
              //I'm adding an array of arrays, which we maintain the array
previously.
              for (int i = 0; i < arrayslength; i++) {
                    for (int j = 0; j < 10; j++)
                          arrays[i][j]=arrays2[i][j];
```

```
            }
            TableModel tablemodel=new DefaultTableModel(arrays, column);//I am
updating the table.
            table.setModel(tablemodel);//I am updating the table.
            classAverageExams(arrays);//we calculate class Average
      }
      public void removeStudentByID(){
            //arrays column two is ID.I want to delete the ID is
textRemoveById.if there is a student I want to delete, counter 4  equal two..
            for (int i = 0; i < arrays.length; i++)
                  if(textRemoveById.equals(arrays[i][2]))
                        counter4=2;

            if(counter4==2)//if counter4 equal two.Student is deleted.
                  JOptionPane.showMessageDialog(null, "Student is deleted.");
            if(counter4!=2)//if counter4 doesnt equal two.No such student is
found.
                  JOptionPane.showMessageDialog(null, "No such student is
found.");
            //filtering except for the identity to be deleted.So the student with
that identity is deleted.
            arrays=(String[][]) Arrays.stream(arrays).filter(s ->
!s[2].equals(textRemoveById)).toArray(String[][]::new);
            TableModel tablemodel=new DefaultTableModel(arrays, column);//i am
uptading the table
            table.setModel(tablemodel);//i am uptading the table
            classAverageExams(arrays);//calculate class average.
      }
      public void removeAllStudent(){
            list1.clear();//list is cleared.
            for (int i = 0; i < arrays.length; i++) //arrays column and row
equals empty.
                  for (int j = 0; j < 10; j++)
                        arrays[i][j]="";

            arrays=new String[0][0];//arrays column and row length is 0
            TableModel tablemodel=new DefaultTableModel(arrays, column);//update
table
            table.setModel(tablemodel);//update table
            sum1=0;sum2=0;sum3=0;sum4=0;sum5=0;sum6=0;// all average is 0
      }
      public void addStudent(){
            int length=1;
            int quiz12,quiz22,project11,final1,midterm11;
            String s[]=textAddStudent.split(" ");
            t3=new Course[length][10];
            for (int i = 0; i < length; i++) {
                  for (int j = 0; j < 10; j++) {
                        t3[i][j]=new Course();
                        if(j==0){
                              list1.add(s[0]);//name
                              t3[i][j].setName(s[j]);
                        }
                        if(j==1){
                              list1.add(s[1]);//lastname
                              t3[i][j].setLastName(s[j]);
                        }
                        if(j==2){
                              list1.add(s[2]);//Id
```

```
                        t3[i][j].setID(s[j]);
                    }
                    if(j==3){
                        quiz1=String.valueOf(s[j]);//quiz1
                         quiz12=Integer.parseInt(quiz1);
                        list1.add(quiz1);
                        t3[i][j].setQuiz1(quiz12);
                    }
                    if(j==4){
                        quiz2=String.valueOf(s[4]);//quiz2
                         quiz12=Integer.parseInt(quiz2);
                        list1.add(quiz2);
                        t3[i][j].setQuiz2(quiz12);
                    }
                    if(j==5){
                        project=String.valueOf(s[5]);//project
                         project11=Integer.parseInt(project);
                        list1.add(project);
                        t3[i][j].setProject(project11);
                    }
                    if(j==6){
                        midterm=String.valueOf(s[6]);//midterm
                         midterm11=Integer.parseInt(midterm);
                        list1.add(midterm);
                        t3[i][j].setMidterm(midterm11);
                    }
                    if(j==7){
                        Final=String.valueOf(s[7]);//final
                        final1=Integer.parseInt(Final);
                        list1.add(Final);
                        t3[i][j].setFinal(final1);
                    }
                    if(j==8){//average

        average=String.valueOf(t3[i][j].calculateAvg(t3[i][3].getQuiz1(),t3[i][4].ge
tQuiz2(),t3[i][5].getProject(),

        t3[i][6].getMidterm(),t3[i][7].getFinal()));

                        list1.add(average);
                    }
                    if(j==9){

        list1.add(t3[i][j].calculateAvgGrade(t3[i][8].getAvarage()));//Letter grade
                    }
                }
            }
            int x=arrays.length;//keep to arrays lenght
            arrays=new String[length+x][10];//(current student number+number of
students to add) is new length row.
            for (int i = 0; i < arrays.length; i++) //Add the students in list1
to arrays.
                for (int j = 0; j < 10; j++)
                    arrays[i][j]=list1.get(j+10*i);
            classAverageExams(arrays);//calculate class average
            TableModel tablemodel=new DefaultTableModel(arrays, column);//update
table
            table.setModel(tablemodel);//update table
        }
```

```
	public static void classAverageExams(String a[][]){
		Double q1[]= new Double[a.length];
		Double q2[]= new Double[a.length],p[]= new Double[a.length],
				m[]= new Double[a.length],f[]= new
Double[a.length],averg[]= new Double[a.length];
		sum1=0;sum2=0;sum3=0;sum4=0;sum5=0;sum6=0;//
		q1	q2		p		m		f	averg
		for (int i = 0; i <a.length; i++) {//Converting String to Double
(quiz1,quiz2,project,midterm,final,average)
			for (int j = 0; j < 10; j++) {
				if(j==3)
					q1[i]=Double.parseDouble(a[i][j]);//quiz1
				if(j==4)
					q2[i]=Double.parseDouble(a[i][j]);//quiz2
				if(j==5)
					p[i]=Double.parseDouble(a[i][j]);//project
				if(j==6)
					m[i]=Double.parseDouble(a[i][j]);//midterm
				if(j==7)
					f[i]=Double.parseDouble(a[i][j]);//final
				if(j==8)
					averg[i]=Double.parseDouble(a[i][j]);//average
			}
		}
		for (int i = 0; i < a.length; i++) {
			sum1+=q1[i];sum2+=q2[i];sum3+=p[i];sum4+=m[i];sum5+=f[i];
			sum6+=averg[i];//sum of
quiz1,quiz2,project,midterm,final,average
		}
		if(a.length!=0){

	sum1=sum1/a.length;sum2=sum2/a.length;sum3=sum3/a.length;sum4=sum4/a.length;
			sum5=sum5/a.length;sum6=sum6/a.length;//averages of
quiz1,quiz2,project,midterm,final,average
		}
		else if(a.length==0){
			sum1=0;sum2=0;sum3=0;sum4=0;sum5=0;sum6=0;//averages of
quiz1,quiz2,project,midterm,final,average
		}
	}
	@Override
	public void actionPerformed(ActionEvent e) {
		if(e.getSource().equals(sortLastName)){//click menuitem Sort by last
name menuitem.
			sortLastName();
			counter2=0;counter3=0;
		}
		if(e.getSource().equals(sortID)){//click  Sort by ID menuitem.
			sortID();
			counter=0;counter3=0;
		}
		if(e.getSource().equals(sortAverage)){//click  Sort by Average
menuitem.
			sortAverage2();
			counter=0;counter2=0;
		}
		if(e.getSource().equals(filterGrade)){//click  Filter by letter grade
menuitem.
```

```
                    textFilter=JOptionPane.showInputDialog(null,"Which grade do you
want to filter?");
                    if(textFilter!=null)
                          filterByGrade();
             }
             if(e.getSource().equals(filterRemove))//click  Remove filter
menuitem.
                    filterRemove();
             if(e.getSource().equals(removeAll)){//click  Remove all Student
menuitem.
                    removeAllStudent();
                    JOptionPane.showMessageDialog(null,"Student list is cleared.");
             }
             if(e.getSource().equals(addStudent)){//click  Add Student menuitem.
                    textAddStudent=JOptionPane.showInputDialog(null,"Enter Student
information with one space i.e (name lastname id quiz1 quiz2 project midterm final
average lettergrade)");
                    if(textAddStudent!=null){
                          addStudent();
                          JOptionPane.showMessageDialog(null," students are added.
Total number of students is: "+arrays.length);
                    }
             }
             if(e.getSource().equals(removeStudent)){//click  Remove Student
menuitem.
                    counter4=0;
                    textRemoveById=JOptionPane.showInputDialog(null,"Enter the ID
of the student you want to remove from the list.");
                    if(textRemoveById!=null)
                          removeStudentByID();
             }
      }
      public static void main(String[] args) throws IOException {
             new main();

      }
}
```

DrawChart class

```
import java.awt.Color;
import java.awt.Dimension;
import java.awt.Font;
import java.awt.FontMetrics;
import java.awt.Graphics;
import java.lang.reflect.Array;
import javax.swing.JComponent;
import java.awt.Graphics2D;
import java.awt.geom.Rectangle2D;

import javax.swing.JPanel;

public class DrawChart extends JPanel{
      main m;
      String
labels[]={"QUÄ°Z1","QUÄ°Z2","PROJECT","MIDTERM","FÄ°NAL","AVERAGE"};//text of bars
      int n;
      int value[];

      public void init()
```

```
	{
		n = 6;
		value = new int[n];
		value[0] = m.sum1;//class average of quiz1
		value[1] = m.sum2;//class average of quiz2
		value[2] = m.sum3;//class average of project
		value[3] = m.sum4;//class average of midterm
		value[4] =m.sum5;//class average of final
		value[5] =m.sum6;//class average of average
	}
	@Override
	public void paint(Graphics g) {
		super.paint(g);
		Graphics2D g2 = (Graphics2D) g;
		init();
		Font font = new Font("Arial",Font.BOLD,15);//font and size
		g.setFont(font);
		for(int i  = 0; i < 6; i ++)
		{
			g.drawString(labels[i], i * 160 + 30, 400);//text of bars.
			g.drawLine(10, 10, 10, 380);
			g.drawLine(10, 380, 950, 380);
			Rectangle2D rect = new Rectangle2D.Double(i*160+30, 380-
value[i]*3.8, 40,  value[i]*3.8);//creating bars
			g2.fill(rect);//draw bars
			g.drawString(String.valueOf(value[i]) + "%", i*160+30, 30);
//it shows the percentages to understand the graph.
		}
	}

}
```

Course Class

```
import java.util.*;
public class Course {
	private  String Name; private String lastName;private String ID;private int
Quiz1;private int Quiz2;private int Project;private int Midterm;private int Final;
	private double Avarage;private String grade;Random r=new Random();main ma;
	public Course() {}
	public Course(String name, String lastName, String iD, int quiz1, int quiz2,
int project, int midterm, int final1,int avarage, String grade) {
		Name = name;
		this.lastName = lastName;
		ID = iD;
		Quiz1 = quiz1;
		Quiz2 = quiz2;
		Project = project;
		Midterm = midterm;
		Final = final1;
		Avarage = avarage;
		this.grade = grade;
	}
	public String getName() {
		return Name;
	}
	public String setName(String name) {
		return Name = name;
	}
	public String getLastName() {
```

```
		return lastName;
	}
	public void setLastName(String lastName) {
		this.lastName = lastName;
	}
	public String getID() {
		return ID;
	}
	public void setID(String iD) {
		ID = iD;
	}
	public int getQuiz1() {
		return Quiz1;
	}
	public void setQuiz1(int quiz1) {
		Quiz1 = quiz1;
	}
	public int getQuiz2() {
		return Quiz2;
	}
	public void setQuiz2(int quiz2) {
		Quiz2 = quiz2;
	}
	public int getProject() {
		return Project;
	}
	public void setProject(int project) {
		Project = project;
	}
	public int getMidterm() {
		return Midterm;
	}
	public void setMidterm(int midterm) {
		Midterm = midterm;
	}
	public int getFinal() {
		return Final;
	}
	public void setFinal(int final1) {
		Final = final1;
	}
	public void setAvarage(double avarage) {
		Avarage = avarage;
	}
	public double getAvarage(){
		return Avarage;
	}
	public String getGrade() {
		return grade;
	}
	public void setGrade(String grade) {
		this.grade = grade;
	}
	@Override
	public String toString() {
		return  "\t"+Name + "\t" + lastName + "\t\t" + ID + "\t" + Quiz1 +
"\t"+ Quiz2 + "\t" + Project + "\t" + Midterm + "\t" + Final + "\t" + "\t" + grade
+ "\n";
	}
```

```
	public  double calculateAvg(int q1,int q2,int p,int m,int f){//gets the
average of all exams.
		Avarage=(q1+q2+p+m+f)/5.0;
		return Avarage;
	}
	public  String calculateAvgGrade(double avg){//gives a letter grade
according to the average class.
		if(avg<=30 && avg>=0)
			grade="F";
		else if(avg>30 && avg<=50)
			grade="D";
		else if(avg>50 && avg<=60)
			grade="D+";
		else if(avg>60 && avg<=65)
			grade="C-";
		else if(avg>65 && avg<=70)
			grade="C";
		else if(avg>70 && avg<=75)
			grade="C+";
		else if(avg>75 && avg<=80)
			grade="B-";
		else if(avg>80 && avg<=85)
			grade="B";
		else if(avg>85 && avg<=90)
			grade="B+";
		else if(avg>90 && avg<=95)
			grade="A-";
		else if(avg>95 && avg<=100)
			grade="A";
		return grade;
	}
}
```