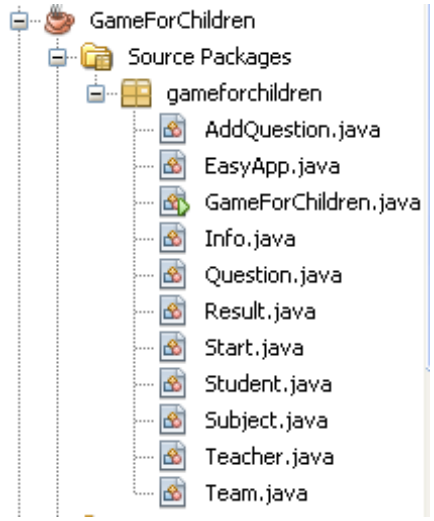
Criterion C: Development

All classes:



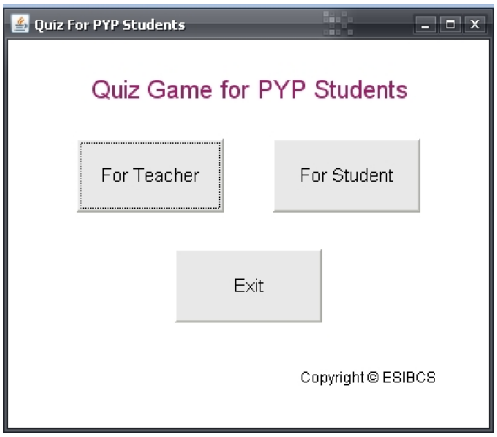
List of the techniques:

1. Each class extends EasyApp[1] to create controls and place them in the window;
2. Different methods are used in the program: methods that are defined in EasyApp.java and methods created by me;
3. String array
4. RandomAccessFiles: Math.txt, Art.txt, Music.txt, SC.txt, UOI.txt, team.txt, Results.txt
5. Static methods and variables

Program file:

[1]Mulkey, Dave. "Free and Easy Java." 2004-2007. http://ibcomp.fis.edu/easyJava/ (accessed xx).

Main window:



Use of controls:

```
Label Title = addLabel("Quiz Game for PYP Students", 70, 40, 500, 60, this);
Button Teacher = addButton("For Teacher", 60, 110, 120, 60, this);
Button Student = addButton("For Student", 220, 110, 120, 60, this);
Button Exit = addButton("Exit", 140, 200, 120, 60, this);
Label CopyRight = addLabel("Copyright © ESIBCS", 240, 280, 200, 50, this);
```

Use of constructor:

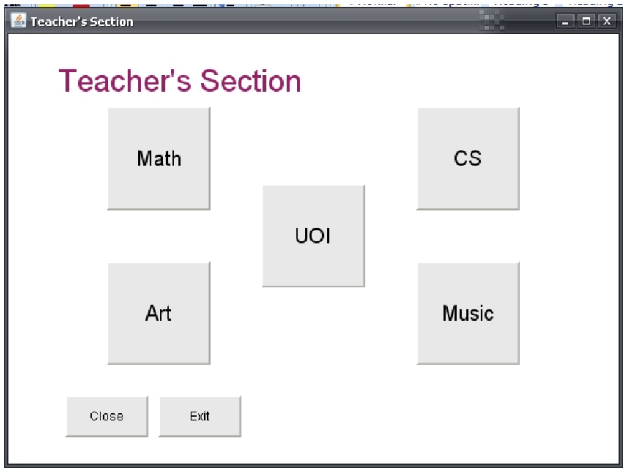
```
public GameForChildren() {

    setTitle("Quiz For PYP Students");
    Title.setForeground(Color.getHSBColor(0.9f, 0.8f, 0.6f));
    Title.setFont(new Font("Arial", 0, 20));
    Teacher.setFont(new Font("Arial", 0, 15));
    Student.setFont(new Font("Arial", 0, 15));
    Exit.setFont(new Font("Arial", 0, 15));
    setBounds(50, 50, 400, 350);
}
```

When a button is clicked appropriate method decides what to do:

```
public void actions(Object source, String command) {
    if (source == Teacher) {
        new Teacher();
    }
    if (source == Student) {
        new Student();
    }
    if (source == Exit) {
        System.exit(0);
    }
}
```

Exit button ends the program.

After clicking button "**Teacher**" – new object "Teacher" will be created according the class Teacher.java



When teacher presses any of them a RandomAccessFile will be opened for riding/writing;

If file length = 0, it will be created and title line will be written in it:

```
void SubjectChoise() {

    // .txt file should be created for each subject
    try {
        RandomAccessFile subjectFile = new RandomAccessFile(subject + ".txt", "rw");
        if (subjectFile.length() == 0) {
            subjectFile.writeBytes(subject + " Questions:\n");
        }
    } catch (IOException e) {
        e.getMessage();
    }

    new Subject();
}
```

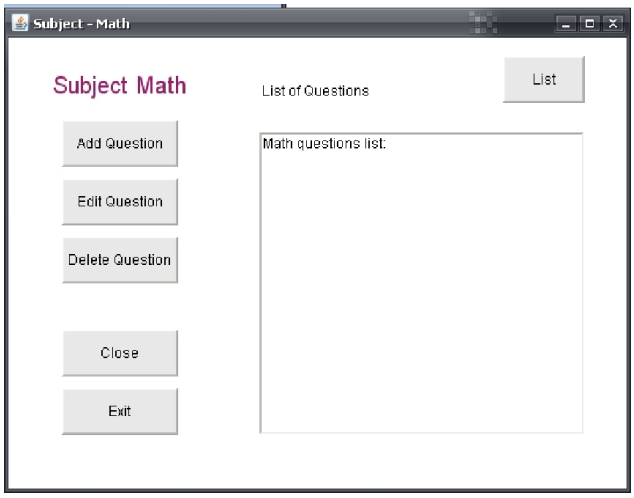
In this class

```
static String subject;

public void actions(Object source, String command) {

    if (source == Math) {
        subject = "Math";
        SubjectChoise();
    }
```

Static variable **subject** "brings" the value – subject name, that will be used in subject class.

```
Label Title = addLabel("Subject " + Teacher.subject, 40, 50, 150, 40, this);
```

Subject class creates new object:



Used **Methods**: arrayWord(),ListButton(),Delete() and EditQuestion()

ListButton() is used for refreshing the list already created questions in that subject. It removes all records from the list then reads records from the Random Access File line-by-line.

```
void ListButton() {
    List.removeAll();
    List.add(Teacher.subject + " questions list:\n");
    //read questions/lines from the appropriate file
    try {
        RandomAccessFile subjectFile = new RandomAccessFile(Teacher.subject + ".txt", "rw");
        subjectFile.seek(0);
        subjectFile.readLine();
        while (subjectFile.getFilePointer() != subjectFile.length()) {
            List.add(subjectFile.readLine());
        }
    } catch (IOException e) {
        e.getMessage();
    }
}
```

(subjectFile.getFilePointer() != subjectFile.length()) will be false after the filepointer reaches end of the file.

List.add(subjectFile.readLine()); - reads line from file and adds in List;

For **Delete()** method we need to create array and putt all records from file in the array:

```
String[] arrayQuestions = new String[25];
 // maximum 25 tasks will be written in one subject file
int k;  // number of elements in array

public void arrayWord() {
    k = 0;
    try {
        RandomAccessFile subjectFile = new RandomAccessFile(Teacher.subject + ".txt", "rw");
        subjectFile.readLine(); //to read title line

        while (subjectFile.length() != subjectFile.getFilePointer()) {
            arrayQuestions[k] = subjectFile.readLine();
            k++;
        }
```

We should fined selected element/record in array and delete it, then write array's elements in file newly:

```
String questionDel = List.getSelectedItem();
      //question, we need to delete from list
arrayWord();

for (int j = 0; j < k; j++) {      //delete word from array
    if (arrayQuestions[j].equals(questionDel)) {
        for (int i = j; i < k - 1; i++) {
            arrayQuestions[i] = arrayQuestions[i + 1];
        }
        arrayQuestions[k - 1] = "";
    }
}
```

I used liner search algorithm here;

```
RandomAccessFile subjectFile =
            new RandomAccessFile(Teacher.subject + ".txt", "rw");
subjectFile.setLength(0);
subjectFile.writeBytes(Teacher.subject + " Questions:\n");
for (int z = 0; z < k - 1; z++) {
    subjectFile.writeBytes(arrayQuestions[z] + "\n");
}
```

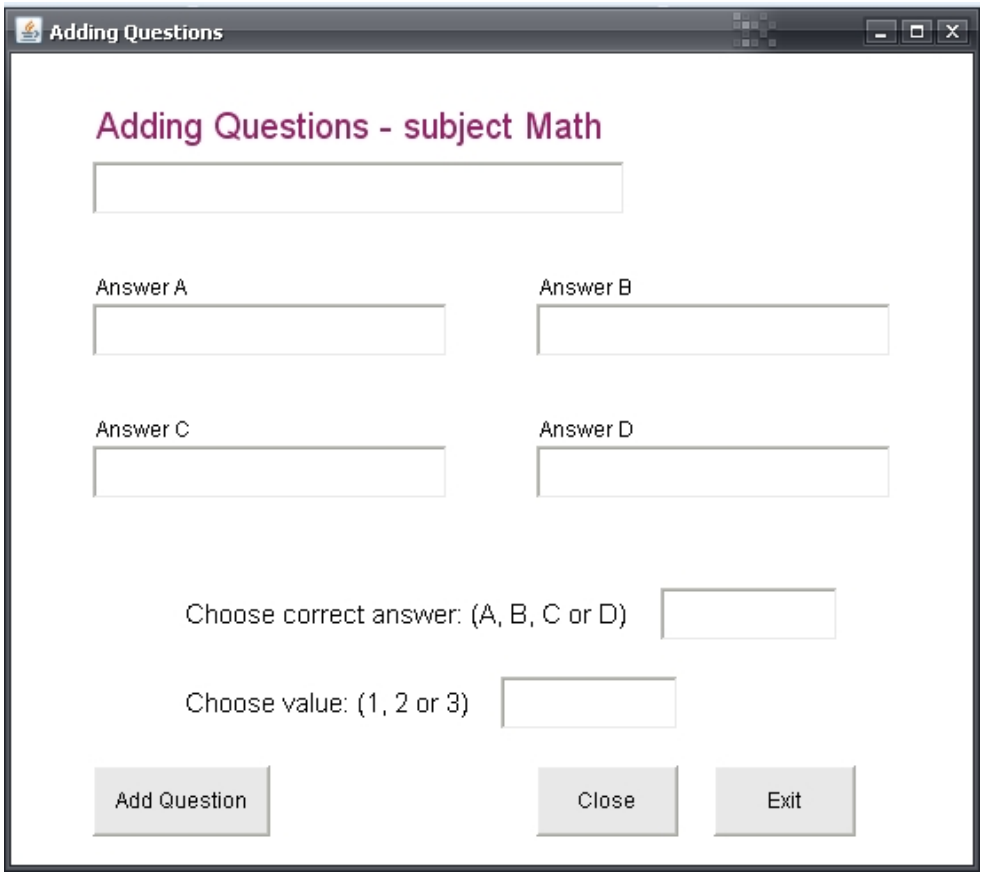
setLength(0) deletes everything from file;

Edit=Delete+add

```
void EditQuestion() {
    String questionEdit = List.getSelectedItem();
                    //question, we need to Edit
    Delete();  // first we delete the sellected question
    // and then edit and save in the file again with AddQuestion method
    new AddQuestion(questionEdit);
}
```

Add question button is used to create and add new questions in Math.txt file.

Pressing button will cause a new window to open.



Method **Add():**

Teacher writes question and four answers in provided fields.

String question = Question.getText() – method helps to take texts from fields.

To check data entry, I created the code:

Correct answers should be: A or B or C or D

Point for question: 1 or 2 or 3

```
String correctA = CorrectAnswer.getText();
String value = Value.getText();
if ((correctA.equals("A") || correctA.equals("B") || correctA.equals("C")
      || correctA.equals("D")) && (value.equals("1") || value.equals("2")
      || value.equals("3"))) {
    if (question.equals("") || answerA.equals("") || answerB.equals("")
      || answerC.equals("") || answerD.equals("") || correctA.equals("")
           || value.equals("")) {
       outputString("Worning!\nFill in all fileds!");
```

Then question will be added in file:

```
//write question in appropriate file directly
try {
    RandomAccessFile subjectFile = new RandomAccessFile(Teacher.subject + ".txt", "rw");
    subjectFile.seek(subjectFile.length());
    subjectFile.writeBytes(question + "~" + answerA + "~" + answerB + "~" + answerC
            + "~" + answerD + "~" + correctA + "~" + value + "\n");
```

In case of Edit – the value of questionEdit variable will be transferred in to AddQuestion class and the selected question will be “devided”into parts and written in textfields.

```
// get all fields initial values
// then Edit data and add in the file with Add() method
int index1 = 0;
int index2 = questionEdit.indexOf("~");
Question.setText(questionEdit.substring(0, index2));
index1 = index2 + 1;
index2 = questionEdit.indexOf("~", index1);
AnswerA.setText(questionEdit.substring(index1, index2));
index1 = index2 + 1;
index2 = questionEdit.indexOf("~", index1);
AnswerB.setText(questionEdit.substring(index1, index2));
```

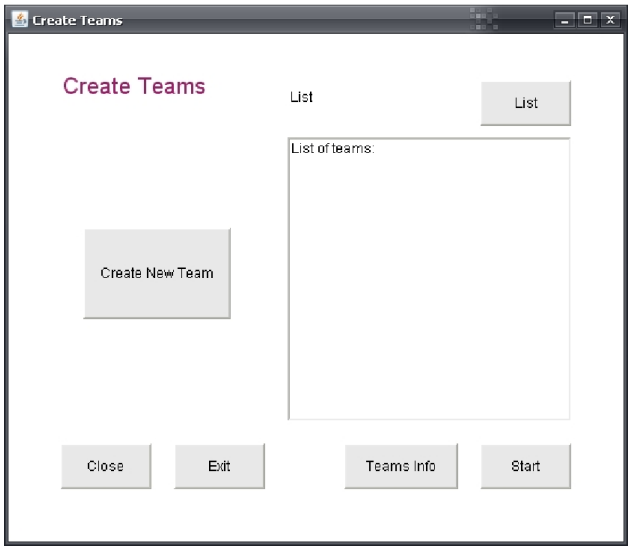
After editing the data – it will be saved again in the txt file.

```
Question.setText("");
AnswerA.setText("");
AnswerB.setText("");
AnswerC.setText("");
AnswerD.setText("");
CorrectAnswer.setText("");
Value.setText("");
```

we use setText(“”) to cleane all data from textfields.

End_of_Teacher_section.

Student's section



there are 4 conrol buttons.

Just like a **List** button in Teacher's subject section, it removes all records, then writes them again from the RandomAccessFile "team.txt" File.

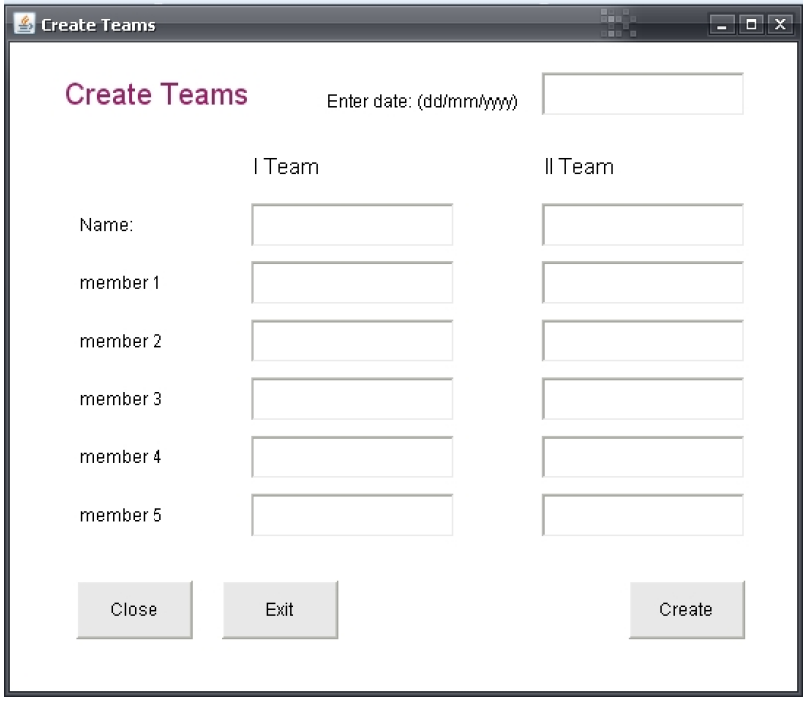
Information about teams consists from 3 lines:

```
5~Spring~Sofo~Lika~Levan~David~Toko
5~Winter~Lasha~Elene~Maka~Anna~Roma
```

```
void ListButton() {
    String Line1;
    String Line2;
    String date;
    List1.removeAll();
    List1.add("List of teams:\n");
    List1.add("\n");
    //read questions/lines from the appropriate file
    try {
        RandomAccessFile teamFile = new RandomAccessFile("team.txt", "rw");

        while (teamFile.getFilePointer() != teamFile.length()) {
            date = teamFile.readLine();
            Line1 = teamFile.readLine();
            Line2 = teamFile.readLine();
            int index1 = Line1.indexOf("~", 2);
            String str1 = Line1.substring(2, index1);
            int index2 = Line2.indexOf("~", 2);
            String str2 = Line2.substring(2, index2);
            List1.add(date + " | " + str1 + " - " + str2);
        }
    } catch (IOException e) {
        e.getMessage();
    }
}
```

Variables date, line1 anline2 are needed to read information about one competition: date and two teams. Then with indexOf() and substring() methods will be written in List window.

Create_New_Teams button opens new window to create teams that will participate in the competition. Here teacher should enter all information and with Create action button all information will be written in team.txt file.



Team.java class conteins methods: Create() to orginaiz teams crating process and Clean() to remove all data from textfields in order client wants to register one more team;

Not obligatory to register all five members; thus, number of members should be counted for each team:

```
line1 = count1 + "~" + line1;
line2 = count2 + "~" + line2;

try {
     RandomAccessFile teamFile = new RandomAccessFile("team.txt", "rw");
     teamFile.seek(teamFile.length());
     teamFile.writeBytes(Date1.getText() + "\n");
     teamFile.writeBytes(line1 + "\n");
     teamFile.writeBytes(line2 + "\n");

void clean() {
     Team1.setText("");
     member11.setText("");
     member12.setText("");
```

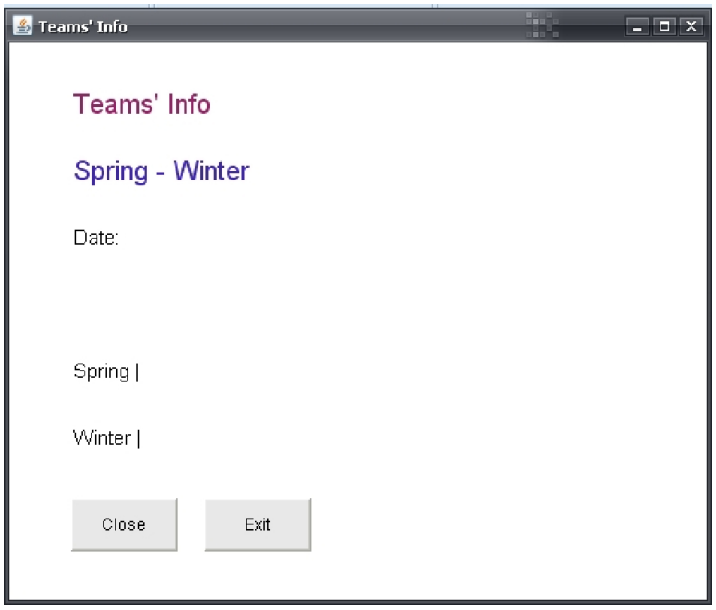
etc. for all textfields.

Teams Info button invokes Info() method to prevent dataEntryErrors and call Info class.

```
void Info() {
    Teams = Listl.getSelectedItem();
    if (Teams == null || Teams.equals("\n") || Teams.equals("List of teams:\n")) {
        outputString("Error! choose Teams");
    } else {
        new Info();
    }
}
```

Teams is static variable that holds selected string, thus we will use this value in Info class;

Info class shows selected teams' info: Names of the Teams, Date, Score and their members; if teams played the game, their scores will be presented in this window.



Teams names and data are taken from "Teams" variable:

```
String line = Student.Teams;  //selected line
InfoDate1 = line.substring(0, line.indexOf("|") - 1); // competition/Game date
InfoNames1 = line.substring(line.indexOf("|") + 2);   //partisipant teams names
Label InfoNames = addLabel(InfoNames1, 50, 100, 200, 50, this);
Label InfoDate = addLabel("Date: " + InfoDate1, 50, 150, 200, 50, this);
```

Members are taken from team.txt file and results, from Results.txt file:

```
try {
    RandomAccessFile teamFile = new RandomAccessFile("team.txt", "rw");
    // all registered teams information is written in this file
```

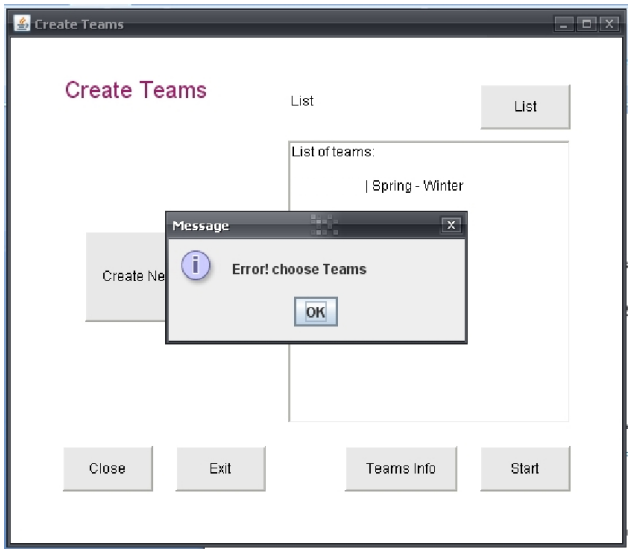
Scores are read from the RandomAccessFile resultFile.

```
try {
    RandomAccessFile resultFile = new RandomAccessFile("Results.txt", "rw");
```

Again, to fined record in file, I used liner search algorith

```
void points() {
    String line1, line2, line3;
          // in result file competition result is written in three lines
    try {
        RandomAccessFile resultFile = new RandomAccessFile("Results.txt", "rw");
                                          //all results are saved in file
        while (resultFile.getFilePointer() != resultFile.length()) {
            line1 = resultFile.readLine();
            line2 = resultFile.readLine();
            line3 = resultFile.readLine();

            if (line1.equals(InfoDate1) && line2.equals(InfoNames1)) {
                Label InfoPoints = addLabel(line3, 50, 190, 200, 50, this);
                InfoPoints.setFont(new Font("Arial", 0, 25));
                InfoPoints.setForeground(Color.getHSBColor(0.68f, 0.9f, 0.9f));
            }
```

Info window also has **Close** and **Exit** buttons for the same uses as anywhere else. If teams are not chosen from the list error message will appear.



Last button is the **Start;** it is used to begin the competition.

Start uses two clasess: **Question** and **Result**;

Start

CreateArray() – is located in constructor

```
//create array with all questions
static String[][] questionArray = new String[90][9];
// subject, question, answer 1, answer 2, answer 3,
// answer 4, correct answer, value, yes/no
static int countQuestions;
```

Max number we need for competition is 90=(9+9)*5 – 9 question for each team from one subject.

9 question=3(1_point_questions) + 3(2_point_questions)+ 3(3_point_questions)

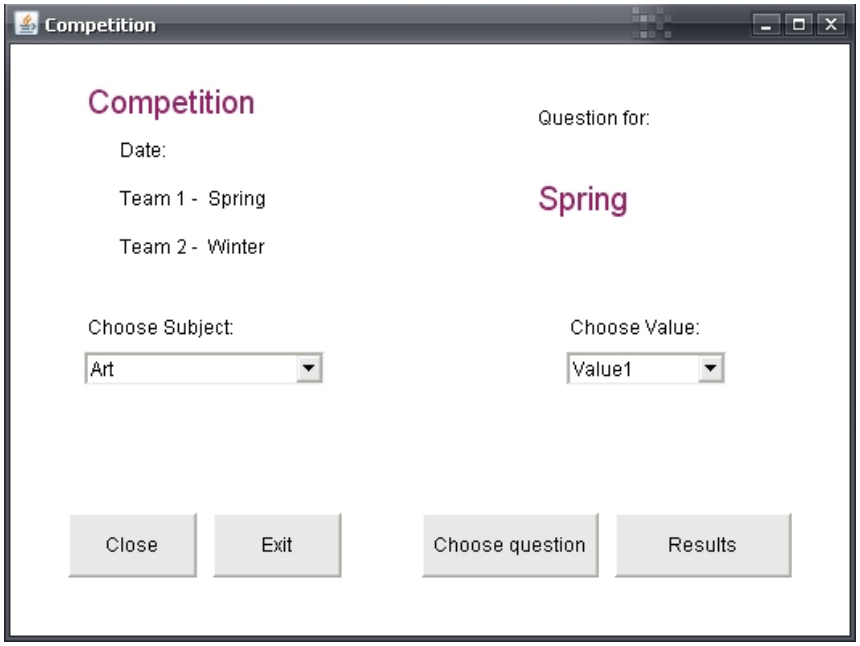
All information will be extracted from files, crated by the teacher according to subjects.

Question() – that calculates which team's turn is and crates new object from **Question class**.

Question To choose question from queationArray acording subject and point_value, display it and after the answer will be entered, calculate how many points are gaind by the team. Also, add points to team result.

Result Is called after pressing button **"Results"** to show teams results in new window.

Start class window:



Start window has Date, name of teams and whose turn is it to answer question. The last is made with the help of the method Question .

```
void Question() {
    subject = Subject.getSelectedItem();  // subject the question belongs to
    value = Value.getSelectedItem();  // question value 1,2,3
    count++;
    if (count <= 30) {  //max questions = 30 - 15 for each teams
        // 15 = 5 subject * 3 questions from each subject
        if (count % 2 == 0) {  // even number question - for team II
            teamName.setText(team2);
            teamsName = team2;
            new Question();
        } else {
            teamName.setText(team1);  // odd number question - for team I
            teamsName = team1;
            new Question();
        }
    } else {
        outputString("The Competition is Finished!\nCheck the results!");
    }
}
```

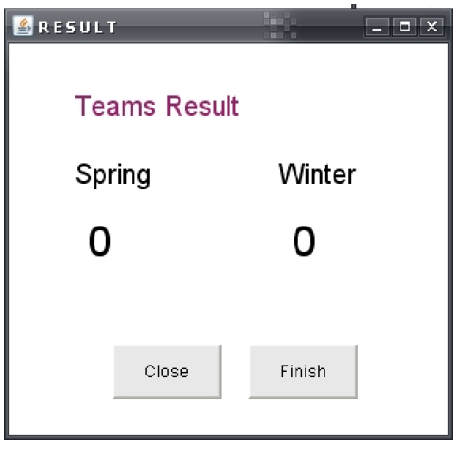
In addition there are 2 choices.

```
Choice Subject = addChoice("Art|Math|Music|CS|UOI", 50, 220, 150, 50, this);
Choice Value = addChoice("Value1|Value2|Value3", 350, 220, 100, 50, this);
```

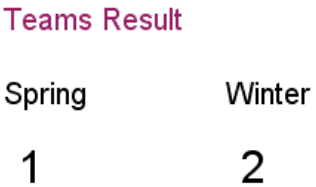
One to chose a subject and another to choose a value.

Result class

Iit opens a new window where scores are displayed; after the teams answer questons they will gain points and the result will be ex. Like this:



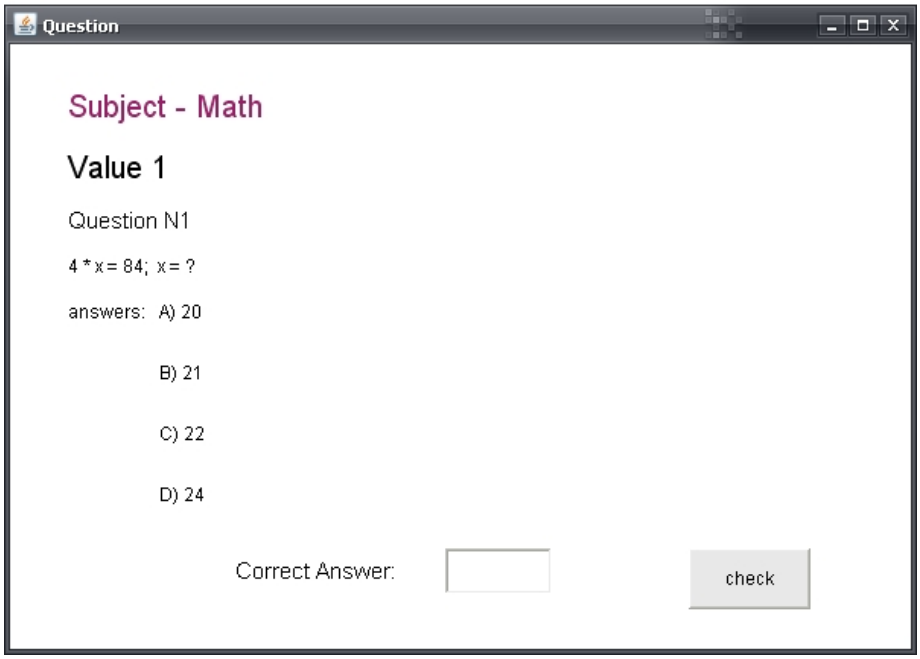
after first questions →



Button **Finish** actes as **Exit** but at first it writes the results in the RandomAccessFile Results.txt.

```
void Finish() {
    String competition = Student.Teams;
    int index1 = competition.indexOf(" ");
    String date = competition.substring(0, index1);
    String team1 = competition.substring(index1 + 3, competition.indexOf(" ", index1 + 4));
    String team2 = competition.substring(competition.lastIndexOf(" ") + 1);

    try {
        RandomAccessFile resultFile = new RandomAccessFile("Results.txt", "rw");
        resultFile.seek(resultFile.length());
        resultFile.writeBytes(date + "\n");
        resultFile.writeBytes(team1 + " - " + team2 + "\n");
        resultFile.writeBytes(team1result + " : " + team2result + "\n");

    } catch (IOException e) {
        e.getMessage();
    }
    System.exit(0);
}
```

The last button of the Start window is **Choose question.**



Choose question opens a new window which shows the Question and provides with the possible answers. All you have to do is write A,B,C or D and then press the **Check** button.

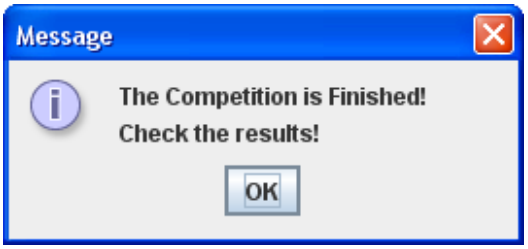
This button compares if the answer you wrote is correct by retrieving the same question from RandomAccessFile, which has the name of the subject you chose, and compares your answer to that writen by the Teacher when she created that question. If correct, you receive the value that the question has, if not you get zero.

```
static String point;  // how many point gain the team
void check() {   //is the answer correct or incorrect; if yes, hom many point the team gained
    String answer = Answer.getText();
    if (answer == null || !answer.equals("A") && !answer.equals("B") && !answer.equals("C")
            && !answer.equals("D")) {
        outputString("Enter Correct Answer!\nTry again...");
    } else {
        if (answer.equals(Start.questionArray[i][6])) {
            point = Start.questionArray[i][7];
        } else {
            point = "0";
        }
        outputString("You gained " + point + " point");
```

After checking the answer – seqond teams turn is to choose question of the particular subject, value and answer it. I used following way to distinguish between teams and add correct values.

```
if (count % 2 == 0) {
    Start.team2array[count / 2 - 1][0] = subject;
    Start.team2array[count / 2 - 1][1] = val;
    Start.team2array[count / 2 - 1][2] = point;
} else {
    Start.team1array[count / 2][0] = subject;
    Start.team1array[count / 2][1] = val;
    Start.team1array[count / 2][2] = point;
}
```

After answering 30 questions, 15 from each side game will automatically finish by the message:




```
void Question() {
    subject = Subject.getSelectedItem();  // subject the question belongs to
    value = Value.getSelectedItem();  // question value 1,2,3
    count++;
    if (count <= 30) {  //max questions = 30 - 15 for each teams
        // 15 = 5 subject * 3 questions from each subject
        if (count % 2 == 0) {  // even number question - for team II
            teamName.setText(team2);
            teamsName = team2;
            new Question();
        } else {
            teamName.setText(team1);  // odd number question - for team I
            teamsName = team1;
            new Question();
        }
    } else {
        outputString("The Competition is Finished!\nCheck the results!");
    }
```

See results by pressing button **Results,** then by pressing the button **Finish** results will be saved in RandomAccessFile resultFile and the program will end.

words - 1015