

Assignment 1

Submission Date : 14.07.2021

Due Date : 21.07.2021 (23:59)

Programming Languages: Java

Subject : Object Oriented Design, Classes and Objects

1 Introduction

In this experiment, you are supposed to develop a simple Student Information System. The main aim of the experiment is to develop your skill of using Class, Object, Constructor with Java programming language.

2 Experiment

This section contains four subsections. The first part defines the problem definition and the second part defines the content of your reports. Third, there are restrictions that you must adhere to. Otherwise, your experiment will not be evaluated by your advisor. The last one is about submission.

2.1 Problem Definition

In this experiment, you are supposed to develop a simple Student Information System. In this system, there are students, courses and grades taken from those courses. The system will process several data input files and produce the results of commands to be read from a command file. All input files will be error free, both syntactically and semantically.

2.1.1 The Way of Execution

The program will be executed with five command line arguments:

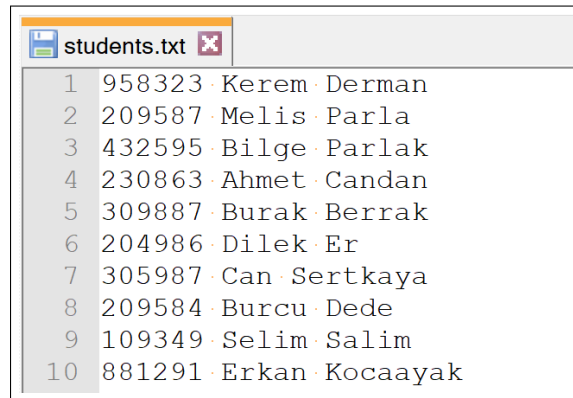
```
<students_file><courses_file><grades_file><commands_file><output_file>
```

There are four types of data input files and one output file. All the file names will be taken as program arguments. The format of each file is given below (WS represents Whitespace).

2.1.2 Students File

```
<STUDENT_ID><WS><STUDENT_NAME><WS><STUDENT_SURNAME>
```

Student's name and surname which are attributes of students class could be at most 30 characters in this system. Student id must be an integer number. According to this definition a sample student file is shown in Figure 1.



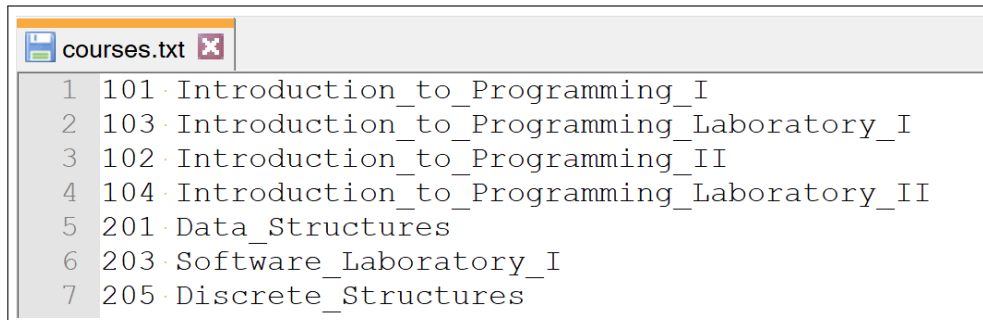
```
1 958323 Kerem Derman
2 209587 Melis Parla
3 432595 Bilge Parlak
4 230863 Ahmet Candan
5 309887 Burak Berrak
6 204986 Dilek Er
7 305987 Can Sertkaya
8 209584 Burcu Dede
9 109349 Selim Salim
10 881291 Erkan Kocaayak
```

Figure 1: Sample students file

2.1.3 Courses File

<COURSE_ID><WS><COURSE_NAME>

Course's name which is the attribute of the courses class could be at most 50 characters length in this system. Course id must be an integer number. According to this definition a sample course file is shown in Figure 2.



```
1 101 Introduction_to_Programming_I
2 103 Introduction_to_Programming_Laboratory_I
3 102 Introduction_to_Programming_II
4 104 Introduction_to_Programming_Laboratory_II
5 201 Data_Structures
6 203 Software_Laboratory_I
7 205 Discrete_Structures
```

Figure 2: Sample courses file

2.1.4 Grades File

There are two types of records in this file. The first one includes courses with one midterm and one final exam, while the other includes courses with two midterms and one final exam.

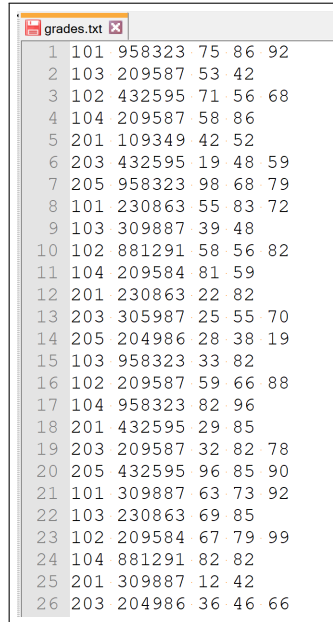
<COURSE_ID><WS><STUDENT_ID><WS><MIDTERM_1_SCORE><WS><MIDTERM_2_SCORE>
<WS><FINAL_EXAM_SCORE>

OR

<COURSEID><WS><STUDENTID><WS><MIDTERMSCORE><WS><FINALEXAMSCORE>

Midterms and final exam points which are attributes of grades class are integer numbers in this system. According to this definition a sample grades file is shown in Figure 3. Grades file includes information which relates a student and a course and scores taken by the student

from that course.



1	101	958323	75	86	92
2	103	209587	53	42	
3	102	432595	71	56	68
4	104	209587	58	86	
5	201	109349	42	52	
6	203	432595	19	48	59
7	205	958323	98	68	79
8	101	230863	55	83	72
9	103	309887	39	48	
10	102	881291	58	56	82
11	104	209584	81	59	
12	201	230863	22	82	
13	203	305987	25	55	70
14	205	204986	28	38	19
15	103	958323	33	82	
16	102	209587	59	66	88
17	104	958323	82	96	
18	201	432595	29	85	
19	203	209587	32	82	78
20	205	432595	96	85	90
21	101	309887	63	73	92
22	103	230863	69	85	
23	102	209584	67	79	99
24	104	881291	82	82	
25	201	309887	12	42	
26	203	204986	36	46	66

Figure 3: Sample grades file

2.1.5 Commands File

All these data input files will be processed according to the commands which will be given in a commands file. The command file can contain nine types of commands whose definitions and formats (in parenthesis) are given below:

1. List all the courses in the system

(LIST COURSES ALL)

2. List the students that take a specified course

(LIST STUDENTS FOR COURSE <COURSE_ID>)

3. List the courses that a student takes

(LIST COURSES FOR STUDENT <STUDENT_ID>)

4. List the grades of a student from a specified course

(LIST GRADES FOR COURSE <COURSEID> AND STUDENT <STUDENTID>)

5. List the grades of a student from all the courses he/she has taken

(LIST GRADES FOR STUDENT <STUDENTID>)

6. Calculate the final grade of a specified student for a specified course

```
(CALCULATE FINALGRADE FOR COURSE <COURSEID> AND STUDENT <STUDENTID>)
```

7. Calculate all the final grade of all the courses which specified student has taken

```
(CALCULATE ALL FINALGRADES FOR STUDENT <STUDENT_ID>)
```

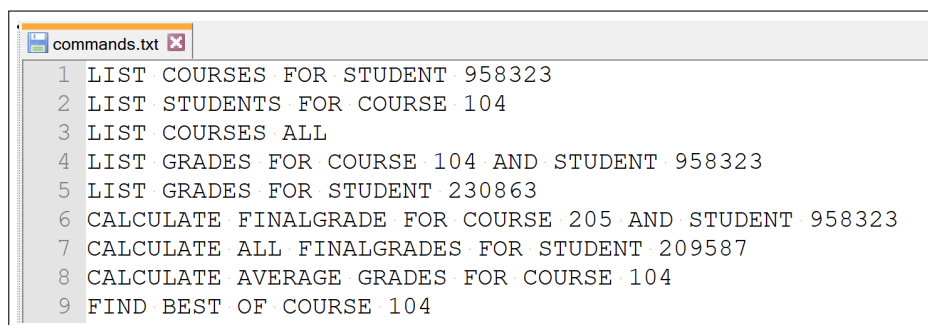
8. Calculate average grades for a specified course

```
(CALCULATE AVERAGE GRADES FOR COURSE <COURSE_ID>)
```

9. Find the student who has the best final grade for a specified course

```
(FIND BEST OF COURSE <COURSE_ID>)
```

A sample commands file is shown in Figure 4.



```
1 LIST COURSES FOR STUDENT 958323
2 LIST STUDENTS FOR COURSE 104
3 LIST COURSES ALL
4 LIST GRADES FOR COURSE 104 AND STUDENT 958323
5 LIST GRADES FOR STUDENT 230863
6 CALCULATE FINALGRADE FOR COURSE 205 AND STUDENT 958323
7 CALCULATE ALL FINALGRADES FOR STUDENT 209587
8 CALCULATE AVERAGE GRADES FOR COURSE 104
9 FIND BEST OF COURSE 104
```

Figure 4: Sample commands file

2.1.6 Output File

The output of the commands will be printed to a output file which is called output.txt. The output of each command should contain each command read from the script itself and the result of its execution (multi line if necessary).

The general format of the output file as shown below:

```
COMMAND:
<COMMAND>
RESULT:
<RESULT_LINE(S)>
```

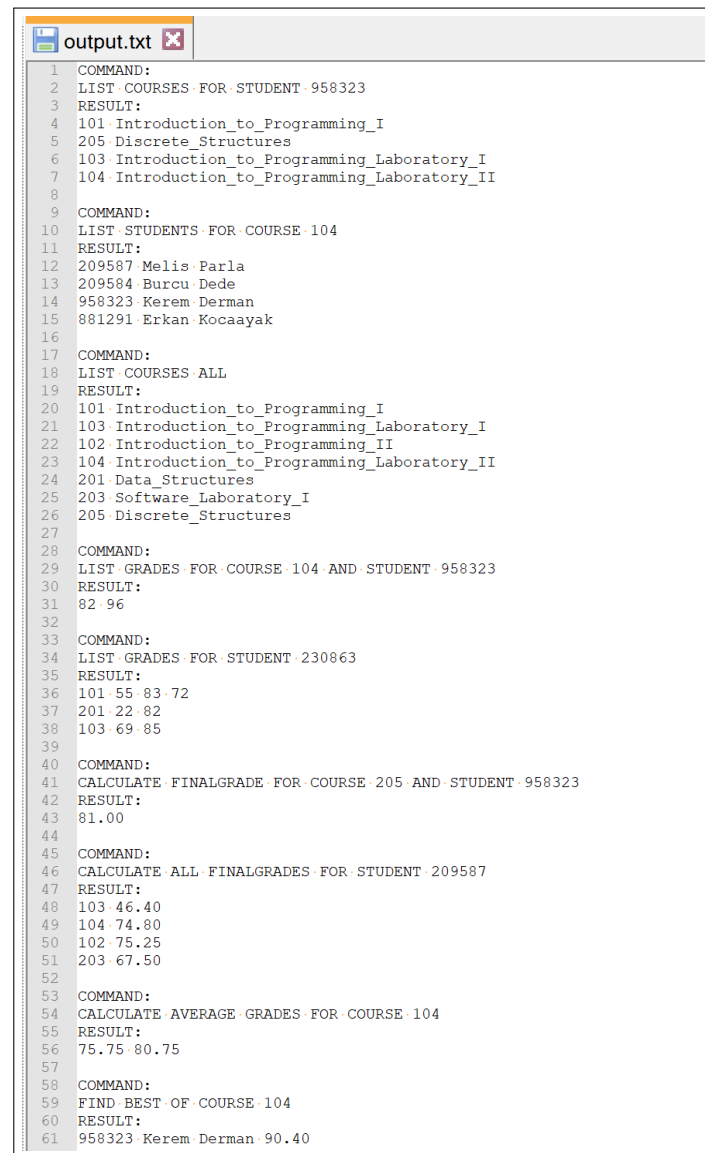
Detailed format of result output for each command type is given below.

1. Command (Multiline if necessary)

```
<COURSEID><WS><COURSENAME>
```

2. Command (Multiline if necessary)

```
<STUDENTID><WS><STUDENTNAME><WS><STUDENTSURNAME>
```



```

1  COMMAND:
2  LIST COURSES FOR STUDENT 958323
3  RESULT:
4  101 Introduction_to_Programming_I
5  205 Discrete_Structures
6  103 Introduction_to_Programming_Laboratory_I
7  104 Introduction_to_Programming_Laboratory_II
8
9  COMMAND:
10 LIST STUDENTS FOR COURSE 104
11 RESULT:
12 209587 Melis Parla
13 209584 Burcu Dede
14 958323 Kerem Derman
15 881291 Erkan Kocaayak
16
17 COMMAND:
18 LIST COURSES ALL
19 RESULT:
20 101 Introduction_to_Programming_I
21 103 Introduction_to_Programming_Laboratory_I
22 102 Introduction_to_Programming_II
23 104 Introduction_to_Programming_Laboratory_II
24 201 Data_Structures
25 203 Software_Laboratory_I
26 205 Discrete_Structures
27
28 COMMAND:
29 LIST GRADES FOR COURSE 104 AND STUDENT 958323
30 RESULT:
31 82 96
32
33 COMMAND:
34 LIST GRADES FOR STUDENT 230863
35 RESULT:
36 101 55 83 72
37 201 22 82
38 103 69 85
39
40 COMMAND:
41 CALCULATE FINALGRADE FOR COURSE 205 AND STUDENT 958323
42 RESULT:
43 81.00
44
45 COMMAND:
46 CALCULATE ALL FINALGRADES FOR STUDENT 209587
47 RESULT:
48 103 46.40
49 104 74.80
50 102 75.25
51 203 67.50
52
53 COMMAND:
54 CALCULATE AVERAGE GRADES FOR COURSE 104
55 RESULT:
56 75.75 80.75
57
58 COMMAND:
59 FIND BEST OF COURSE 104
60 RESULT:
61 958323 Kerem Derman 90.40

```

Figure 5: Sample output file

3. Command (Multiline if necessary)

<COURSEID><WS><COURSENAME>

4. Command (Single line)

<MIDTERM1SCORE><WS><MIDTERM2SCORE><WS><FINALEXAMSCORE>

or

<MIDTERMScore><WS><FINALEXAMSCORE>

5. Command (Multiline if necessary)

<COURSEID><WS><MIDTERM1SCORE><WS><MIDTERM2SCORE><WS><FINALEXAMSCORE>

or/and

<COURSEID><WS><MIDTERMScore><WS><FINALEXAMScore>

6. Command (Single line)

<FINALGRADE>

7. Command (Multiline if necessary)

<COURSEID><WS><FINALGRADE>

8. Command (Single line)

<MIDTERM1AVERAGE><WS><MIDTERM2AVERAGE><WS><FINALEXAMAverage>
or
<MIDTERMAverage><WS><FINALEXAMAverage>

9. Command (Single line)

<STUDENTID><WS><STUDENTNAME><WS><STUDENTSURNAME><WS><FINALGRADE>

Final grade for a course will be calculated as follows:

- If the course has one midterm and one final exam, the final grade is the sum of 40% of the midterm and 60% of the final exam.
- If the course has two midterms and one final exam, the final grade is the sum of %50 of the midterms' average and %50 of the final exam score.

According to these definitions a sample output file is given in Figure 5 .

2.2 Report

The structure of report is described below:

- Cover Page
- Main Data Structures, give your main data structures and explain them.
- Solution, describe details of your program, stating its advantages and disadvantages technically.
- Comments, give feedback about problem, problem description, and solution constraints.
- References, give the references you used throughout your work at the end of your report.

2.3 Constraints

1. The methods' and attributes' names should be satisfied the most common naming conventions in Java.
2. **You should model entities of the system with classes.**
3. Your design will be graded. You are expected to propose a suitable design for the problem.
4. You are responsible for a correct model design. Your design should be accurate.

5. All the input files(student, course, grades and commands file) have at most 500 records in this system. Design and implement your project according to this information.
6. All the input files and output file will be taken as command line arguments.
7. Different inputs and outputs will be used while your work is being evaluated.

2.4 Submit Format

- File hierarchy must be zipped before submitted (Not .rar, only .zip files are supported by the system)

- $\langle studentid \rangle.zip$ (example : 1234567.zip)

- [src]
 - Main.java
 - *.java
- [report]
 - report.pdf
 - *.jpg or *.jpeg

3 Grading Policy

Your experiment results will be evaluated with the following rules. They are fixed. They will never be changed. Hence obey these rules.

- Compilation of program have 10 points.
- The correct output of program have 63 points.
- The submit format have 1 points.
- The report have 26 points: 1 point for Cover Page, 10 points for Object Oriented Design, Classes and Objects, 12 points for Solution, 2 points for Comments and 1 point for References.

4 Notes

- You have to design your application in complete object oriented design approach. You are expected to define a java class for everything that is suitable for a class.
- Class diagram for full program must be given in your report. You have prepare a basic class diagram so you need to do little research about UML Class Diagrams.
- The assignment must be original, individual work. Downloaded or modified source codes will be considered as cheating. Also the students who share their works will be punished in the same way.

- We will be using the Measure of Software Similarity (MOSS) to identify cases of possible plagiarism. Our department takes the act of plagiarism very seriously. Those caught plagiarizing (both originators and copiers) will be sanctioned.
- Please do not miss that the name of the input and output files will be fixed. Use the file names as indicated in the leaflet. File names with different names will not be evaluated.
- You can ask your questions through course's piazza group and you are supposed to be aware of everything discussed in the piazza group. General discussion of the problem is allowed, but DO NOT SHARE answers, algorithms, source codes and reports .
- Ignore the cases which are not stated in this assignment and do not ask questions on Piazza for such extreme cases.
- Don't forget to write comments of your codes when necessary.
- The names of classes', attributes' and methods' should obey to Java naming convention.
- Do not submit your project without first compiling it on dev machine.
- Save all work until the assignment is graded.
- Do not miss the deadline. Submission will be end at 21/07/2021 23:59, the system will be open until 23:59:59. The problem about submission after 23:59 will not be considered.