# Technical Specification

­­­

Author : Juyeon Lee , Moeun Son

Date : 19.02.2019

1. Introduction ………………………………………………………………………………………………...2
2. Main features……………………………………………………………………………………………….2
3. UML of Classes……………………………………………………………………………………………...2
4. Methods in Launcher……………………………………………………………………………………3
5. Methods in Services……………………………………………………………………………………..6
6. Console Application……………………………………………………………………………………..7
7. Schemas………………………………………………………………………………………………………9
8. Sample data…………………………………………………………………………………………………9

## Introduction

We’ve had a project in OOA&UML&JAVA lecture for 1 semester. The project is for evaluation of overall concepts of our lecture.

We used database ‘H2’ for store and manage data. Plus, we made a console application to make quiz manager. We uploaded our source code at github.com , so you can see ours. The address is <https://github.com/Juyeon-Lee/Epita_Java_project.git>

## Main features

The scope that we covered by this application is to :

* Be able to operate CRUD on Open Questions and MCQ Questions (questions and valid answers are stored in a database)
* Be able to search questions based on topics
* Be able to assemble automatically a quiz (a quiz is a set of questions) that gathers all the questions covering a given list of topics.
* Export this quiz under a plain text format if user want to export
* Run the evaluation and provide the automatic mark in the end of this execution[[1]](#footnote-1)
* Be able to operate our data which is at linked database.

## UML of classes

This diagram shows structures of our classes of datamodel package. We used almost similar UML with professor’s original one, but modified class ‘MCQQuestion’ and ‘Question’. We add ‘choice1’, ‘choice2’, ‘choice3’, ‘choice4’, and ‘answer’ elements at ‘MCQQuestion’ class. We add ‘id’ and ‘mcq’ elements at ‘Question’ class, also.

Actually, we used three classes of datamodel package. They are student, MCQQuestion, and Question classes.

About services package, there are ‘Configuration’, ‘MCQQuestionJDBCDAO’, and ‘QuestionJDBCDAO’ classes. Finally, there is a launcher class at launcher package.

텍스트, 지도이(가) 표시된 사진

자동 생성된 설명

## Methods in Launcher

1. Launcher

When you run the launcher, there will be two parts; admin questions/student(solve a quiz).

The ‘admin questions’ part will be able to operate CRUD on Open Questions and MCQ Questions, and the ‘student(solve a quiz)’ part is for quizzes which questions are collected by nothing(so all questions), topics, and difficulty levels in them.

Let’s talk about the ‘admin questions’ part. First, a manager can create/update/delete both open question and mcq question as many as he or she wants. Before update or delete, the manager can read a list of questions. It will include all information of questions. This function is same with ‘show all’. It will print everything in console. Plus, a manager can search questions by a topic.

Next is the ‘student(solve a quiz)’ part. First, a student can choose to one's quiz by nothing, topic, or difficulty and solve the quiz. Second, student can choose to extract the quiz to a file after every quiz is finished. Finally, the student scores are printed and the quiz ends.

* 1. Detail of methods in launcher

|  |  |
| --- | --- |
| private static void | [**adminStartPage**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#adminStartPage--)()  Print administrator's start Page |
| private static void | [**deleteQuestion**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#deleteQuestion-java.util.Scanner-quiz.services.QuestionJDBCDAO-quiz.services.MCQQuestionJDBCDAO-)(java.util.Scanner scanner, [**QuestionJDBCDAO**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html) dao, [**MCQQuestionJDBCDAO**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\MCQQuestionJDBCDAO.html) mcqDao)  You can delete a open question or MCQquestion in administrator's mode If you don't know well about questions, you can see the list of questions You must follow the rule of typing. |
| private static void | [**extractQuizToFile**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#extractQuizToFile-quiz.services.MCQQuestionJDBCDAO-java.util.List-)([**MCQQuestionJDBCDAO**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\MCQQuestionJDBCDAO.html) mcqDao, java.util.List<**[Question](file:///C:\\Users\\LeeJuyeon\\git\\repository\\quiz-manager2\\doc\\quiz\\datamodel\\Question.html" \o "class in quiz.datamodel)**> fileQuestion)  This is a method that saves the quizzes to a file. |
| private static [**Question**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\datamodel\Question.html) | [**insertGeneralInfo**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#insertGeneralInfo-java.util.Scanner-)(java.util.Scanner scanner)  This method will get inputs(question, topics, difficulty) from the user, and return question to [**insertQuestion(Scanner)**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#insertQuestion-java.util.Scanner-) by Question[**Question**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\datamodel\Question.html) form. |
| private static java.util.List<java.lang.String> | [**insertMCQInfo**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#insertMCQInfo-java.util.Scanner-)(java.util.Scanner scanner)  This method will get inputs(choices and answer) from the user, and return the list to [**insertQuestion(Scanner)**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#insertQuestion-java.util.Scanner-) by List of String form. |
| private static void | [**insertQuestion**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#insertQuestion-java.util.Scanner-)(java.util.Scanner scanner)  You can insert a open question or MCQquestion in administrator's mode : Once you go to [**insertGeneralInfo(Scanner)**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#insertGeneralInfo-java.util.Scanner-), if you want to create a MCQquestion, you'll go to [**insertMCQInfo(Scanner)**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#insertMCQInfo-java.util.Scanner-). |
| static void | [**main**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#main-java.lang.String:A-)(java.lang.String[] args) |
| private static void | [**searchByTopic**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#searchByTopic-java.util.Scanner-)(java.util.Scanner scanner)  This will scan a topic from user, and send it to searchByTopic of dao. |
| private static void | [**showAll**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#showAll-quiz.services.QuestionJDBCDAO-)([**QuestionJDBCDAO**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html) dao)  You can see the list of questions in administrator's mode |
| private static java.util.List<[**Question**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\datamodel\Question.html)> | [**solveQuiz**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#solveQuiz-java.util.List-java.lang.String-)(java.util.List<[**Question**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\datamodel\Question.html)> fileQuestion, java.lang.String condition)  This is a search the question and answer by the condition(where topic/difficulty( if you select the number(condition) , this search and return List of Questions |
| private static void | [**studentLogin**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#studentLogin-java.util.Scanner-quiz.datamodel.Student-)(java.util.Scanner scanner, [**Student**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\datamodel\Student.html) student)  save the student info(name and id) |
| private static void | [**updateQuestion**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\launcher\Launcher.html#updateQuestion-java.util.Scanner-quiz.services.QuestionJDBCDAO-quiz.services.MCQQuestionJDBCDAO-)(java.util.Scanner scanner, [**QuestionJDBCDAO**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html) dao, [**MCQQuestionJDBCDAO**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\MCQQuestionJDBCDAO.html) mcq)  You can update a open question or MCQquestion in administrator's mode If you don't know well about questions, you can see the list of questions You must follow the rule of typing. |

## Methods in Services

1. Configuration

This class is from my professor. It’s for configuration, setting ‘conf.location’, and getting instance.

1. QuestionJDBCDAO

|  |
| --- |
| [**create**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html#create-quiz.datamodel.Question-int-)([**Question**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\datamodel\Question.html) question, int mcq)  This method will perform 'insert' command by using parameters for question table. |
| [**delete**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html#delete-int-)(int id)  This method will perform 'delete' command by id. |
| [**getConnection**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html#getConnection--)() |
| [**ifLinkedornotWithMCQ**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html#ifLinkedornotWithMCQ-int-)(int mcqid)  This method will return 1 or 0 that is indicated if it's linked or not with MCQquestion. |
| [**printAll**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html#printAll--)()  It will print all information of all questions at console. |
| [**search**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html#search-quiz.datamodel.Question-)([**Question**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\datamodel\Question.html) question)  This method will excute query "select Id, difficulty, question from question where difficulty = ?". |
| [**searchByTopic**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html#searchByTopic-java.lang.String-)(java.lang.String topic)  It will print some information of questions of one topic at console. |
| [**showAll**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html#showAll-java.lang.String-)(java.lang.String condition)  This method will do select query with a condition, and send List of questions with id, question, linked mcq id, topics, difficulty. |
| [**showAllID**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html#showAllID--)()  The method will print all Ids of questions. |
| [**toListfromString**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html#toListfromString-java.lang.String-)(java.lang.String topic)  This method will convert a String of topics to a list of topics. |
| [**toStringofTopics**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html#toStringofTopics-java.util.List-)(java.util.List<java.lang.String> topics)  This method will convert a list of topics to a String of topics. |
| [**update**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\QuestionJDBCDAO.html#update-java.util.ArrayList-)(java.util.ArrayList<java.lang.String> list)  This method will perform 'update' command by using parameters. |

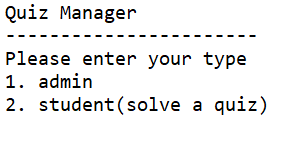
This class if for connection and CRUD with table Question. At fields, there are some statements of CRUD queries. Under table is for summary of our methods by Javadoc.

1. MCQQuestionJDBCDAO

This class if for connection and CRUD with table MCQQuestion. Under table is for summary of our methods by Javadoc.

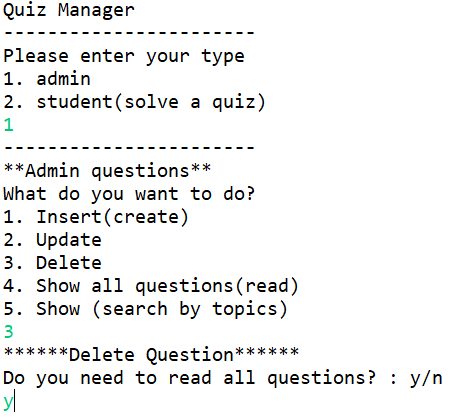
|  |
| --- |
| [**correctAnswer**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\MCQQuestionJDBCDAO.html#correctAnswer-int-java.lang.String-)(int mcqid, java.lang.String answer)  **Deprecated.** |
| [**create**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\MCQQuestionJDBCDAO.html#create-java.util.List-)(java.util.List<java.lang.String> mcqInfo)  This method will perform 'insert' command by using parameters for MCQQuestion table. |
| [**delete**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\MCQQuestionJDBCDAO.html#delete-int-)(int mcqid)  This method will perform 'delete' command by using mcqid for MCQQuestion table. |
| [**getConnection**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\MCQQuestionJDBCDAO.html#getConnection--)() |
| [**print**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\MCQQuestionJDBCDAO.html#print-int-)(int mcqid)  This method will print choice1, choice2, choice3, choice4, and answer by mcqid. |
| [**showWhereMcqid**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\MCQQuestionJDBCDAO.html#showWhereMcqid-int-)(int mcqid)  This method will be called with mcqid. |
| [**update**](file:///C:\Users\LeeJuyeon\git\repository\quiz-manager2\doc\quiz\services\MCQQuestionJDBCDAO.html#update-java.util.ArrayList-)(java.util.ArrayList<java.lang.String> list)  This method will perform 'update' command by using parameters for MCQQuestion table. |

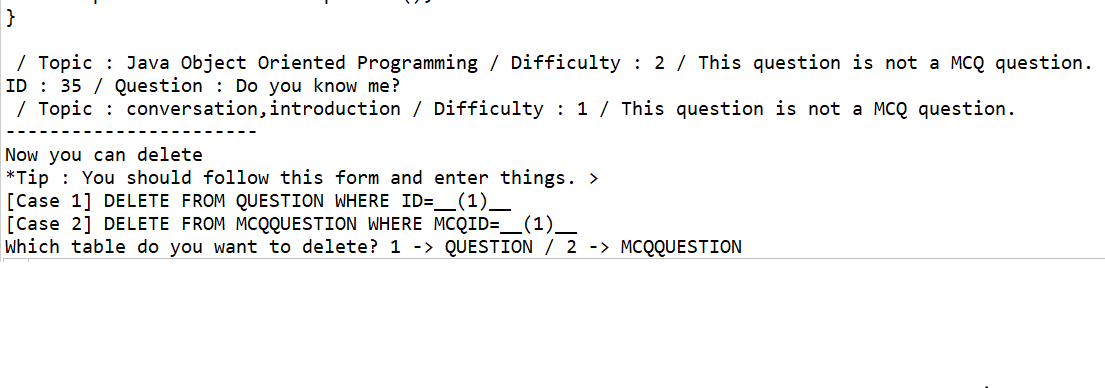
## Console Application

We used logger and System.out.println for our console application. Here is introduction of our program :

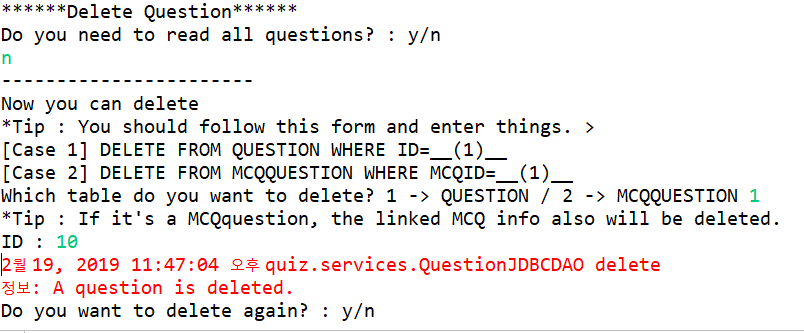
First, this is the start page;

If I enter ‘1’, the left picture is admin questions page.

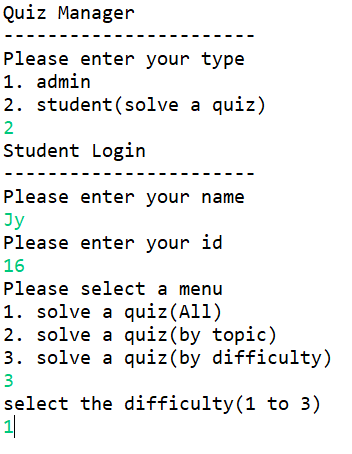
For example, next if I type ‘3’, the delete page will be shown. Before delete, you can select if you want to read all questions or not. You should enter ‘y’ or ‘n’.



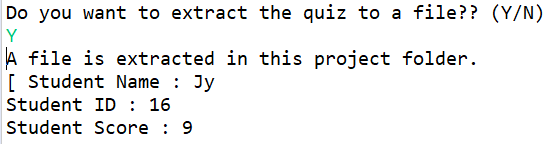
You can read all questions, because you enter ‘y’. After all questions, you can delete by following the form.



After select a table which the data you want to delete is in it, you can delete a row by entering a id. If it’s a MCQquestion, the linked MCQ info also will be deleted. Once it’s successed, there will be a sentence “A question is deleted.” Then if you want to delete another again, you have to enter ‘y’.

 This picture is for solving a quiz. Before start, you should enter your name and id. Then select a type of quiz. At the picture, there was a quiz by difficulty1.

After finishing the quiz, you can extract the quiz to a file, and see your score of the quiz.



## Schemas

We have two tables in database, which are ‘Question’ and ‘MCQQuestion’. At table Question, the primary key is ID and it’s integer. Also, there are QUESTION and TOPIC of varchar type, and there are MCQ and DIFFICULTY of integer type.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Question** | | | | |
| INT | VARCHAR(1024) | INT | VARCHAR(60) | INT |
| ID | QUESTION | MCQ | TOPIC | DIFFICULTY |

At table MCQQuestion, the primary key is MCQID and it’s integer. Also, there are CHOICE1, CHOICE2, CHOICE3, CHOICE4, and ANSWER of varchar type.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **MCQQuestion** | | | | | |
| INT | VARCHAR(1024) | VARCHAR(1024) | VARCHAR(1024) | VARCHAR(1024) | VARCHAR(1024) |
| MCQID | CHOICE1 | CHOICE2 | CHOICE3 | CHOICE4 | Answer |

## Sample Data

We had set up sample data in database, like under two table. There are about 10 sample data for each table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Question | | | | |
| INT | VARCHAR(1024) | INT | VARCHAR(60) | INT |
| **ID** | **QUESTION** | **MCQ** | **TOPIC** | **DIFFICULTY** |
| 1 | **What will happen when you compile and run the following code?**  public class Test{    public static void main(String[] args){    double d1 = Double.NaN;  double d2 = d1;    if(d1 == d2)  System.out.println("Equal");  else  System.out.println("Not Equal");  }    } | 1 | Java Language Basics | 1 |
| 2 | **What will happen when you compile and run the following code?**  public class Test{    public static void main(String[] args){  int i = 10;  int j = 25;  System.out.println(i + ' ' + j);  }  } | 2 | Java Language Basics | 1 |
| 3 | **What will happen when you compile and run the following code?**  public class Test {    public static void main(String[] args){    int i = 10;  int j = 12;  i+=++j;  System.out.println(i);  }  } | 3 | Java Language Basics | 1 |
| 4 | **What will happen when you compile and run the following code?**  public class Test {    public static void main(String[] args){  int i = 010;  System.out.println(i);  }  } | 4 | Java Language Basics | 1 |
| 5 | **What will happen when you compile and run the following code?**  public class Test{    public static void main(String[] args) {  int i = 0;  int j = i++ + ++i;  System.out.println( j );  }  } | 5 | Java Operators | 1 |
| 6 | **What will happen when you compile and run the following code?**  public class Test{    public static void main(String[] args) {  int i = 0;  int j = ++i + i++;  System.out.println( j );  }  } | 6 | Java Operators | 1 |
| 7 | **What will happen when you compile and run the following code?**  public class Test{    public static void main(String[] args) {  int i = 100;  int j = 10;  System.out.println( i%j );  }  } | 7 | Java Operators | 1 |
| 8 | **What will happen when you compile and run the following code?**  public class Test{    public static void main(String[] args) {  int i = 0;  int j = 1;  if(!i && j)  System.out.println("1");  else  System.out.println("2");  }  } | null | Java Operators | 1 |
| 9 | **What will happen when you compile and run the following code?**  class One{  int i = 1;  public int getInt(){  return i;  }  }  class Two extends One{  int i = 2;  public int getInt(){  return i;  }  }  public class Test{  public static void main(String[] args) {  One one = new One();  Two two = (Two)one;  System.out.println( two.getInt() );  }  } | 8 | Java Object Oriented Programming | 3 |
| 10 | **What will happen when you compile and run the following code?**  class One{  public static void print(){  System.out.println("1");  }  }  class Two extends One{  public static void print(){  System.out.println("2");  }  }  public class Test{  public static void main(String args[]){  One one = new Two();  one.print();  }  } | 9 | Java Object Oriented Programming | 3 |
| 11 | **What will happen when you compile and run the following code?**  class One{  public static void print(int i){  System.out.println("Parent");  }  }  class Two extends One{  public static void print(byte b){  System.out.println("Child");  }  }  public class Test{    public static void main(String args[]){  One one = new Two();  one.print(10);  }  } | 10 | Java Object Oriented Programming | 3 |
| 12 | **Will this code compile successfully?**  class One{  public void process(){  System.out.println("Parent");  }  }  public abstract class Test extends One{  public abstract void process();  } | null | Java Object Oriented Programming | 2 |
| 13 | **What’s your university name?** | null | conversation, introduction | 1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MCQQuestion | | | | | |
| INT | VARCHAR(1024) | VARCHAR(1024) | VARCHAR(1024) | VARCHAR(1024) | VARCHAR(1024) |
| **MCQID** | **CHOICE1** | **CHOICE2** | **CHOICE3** | **CHOICE4** | **Answer** |
| 1 | Equal | Not Equal | The code will not compile | The code will compile but will throw run time error | Not Equal |
| 2 | 10 25 | 35 | 67 | The code will not compile | 67 |
| 3 | 13 | 23 | 22 | 12 | 23 |
| 4 | Compilation error | Runtime error | 8 | 10 | 8 |
| 5 | 1 | 3 | 2 | 4 | 2 |
| 6 | 3 | 2 | 4 | 1 | 2 |
| 7 | 10 | 0 | Compilation error | None of the above | 0 |
| 8 | 1 | 2 | Compilation error | Runtime error | Runtime error |
| 9 | 1 | 2 | Compilation error | Runtime error | 1 |
| 10 | Parent | Child | Compilation error | Runtime error | Parent |

1. https://thomas-broussard.fr/work/java/courses/project/fundamental.xhtml [↑](#footnote-ref-1)