Cairo University Faculty of Computers and Information

CS352 – Software Engineering II

Phase 1 Template

CS352: Phase 1 – CIS

Phase 1 document

2017

Project Team

ID	Name	Email	Mobile
	1st name is team leader		

Staff:

Dr Amr Kamel

a.kamel@fci-cu.edu.eg

Dr Khadiga Mohamed

kelbedweihy@fci-cu.edu.eg

[Write here your TA name only in your lab]

CS352: Phase 1 – CIS
Phase 1 document

Eng Omar Khaled Ali Ragab o.khaled@fci-cu.edu.eg

Contents

Design and Code Checklist

Design Principles

1- Does the design follow SOLID principles? Yes

What % 80 of design follow SOLID principles
Related Issues:
Code applies the majority of SOLID principles, as an example, each class has single functionality and had created separately that make it easily to be extended without modifying the class itself. On the other hand, code does not apply liskov substitution principle like in “Student” class which inherits from “Teacher” class but “Student” class shouldn’t apply “addGame” method based on the functional requirement.

2- Does the design follow OOP rules?

Yes What % 90
Related Issues:
Classes and methods have meaningful names, code does not have many if and while condition , avoid global state and static classes ,there is no long method , low complexity level, there is no long methods, classes in a package are reused together, but there are numerous of classes that need a lot of time to testing .

3- Is the design simple and easy to modify?

Yes What % 90
Related Issues:
The design document almost is as is the implementation,
code is readable, easy to understand and have low complexity level,
Boundary class separate from the databases and business classes that any modify in class will not affect in other classes. This all make the design simple and easy to modify.

Coding Standards

4- Is the code understandable and readable? <u>Yes</u> What % 90

Related Issues:

Well-written code with:

- Good-representative and meaningful variables, methods and classes names.
- Single-purpose variables, classes and methods.
- Consistent indentation and formatting style.
- Reduction of the nesting level in code.

5- Does the code follow Java Coding Style? <u>Yes</u> What % 90

Related Issues:

The code follows java coding conventions as follows:

- Classes' names are nouns or noun phrases those start with uppercases letters e.g. "Account" and "CategoryManager" Classes.
- Variables' names start with lowercase letters e.g. "gender" and any words after start with uppercase letters e.g. "gameModel".
- Methods' names start with lowercase letters and any words after start with uppercase letters. They are typically verbs but can be nouns e.g. "showFormHeader ()" and "getPassword ()".
- Files' names are the same as classes' names and beginning with uppercase letters, too e.g. "Account.java" and "CategoryManager.java".
- Comments are simply notes that the user types such as the one-line comment in line 31 in "Question" Class.
- Indentations are concerned in the code clearly.
- Code structure is concerned, too. This is the proper structuring of code, such as, indentation, quotation, semi-colons, commas, parentheses, etc. to ensure the program is correctly done.
- Use of braces, parentheses, are used to surround classes, methods and loops when writing the code.
- Java code has a column limit of 100 characters. Except as noted below, any line that would exceed this limit must be line-wrapped. This is not applied to some lines of code like 61 and 64 in "TournamentManager"

Class.

6- Is indentation used properly? Yes What % 95

Related Issues:

Code concerns for indentation and programming style as a sort of code organization to keep it readable and understandable.

7- Do variable have good names? Yes What % 80

Related Issues:

The majority of variables names are meaningful and well-representative names with a reasonable length in average but some names seems so long like this method name: “createdSuccessfullyMessage()” it could be named as createdMsgSuccess() or something like this!

Comments

8- Is the code commented enough? No What % 8 of code is commented enough.

Related Issues:

It is very rare to find a comment! Classes and methods are not commented to tell their work or purpose. Some magic numbers used not commented also, such as, 1 and 4.

9- Is every class and method commented? No What %? 2% of classes and methods commented.

Related Issues:

Two methods only which are commented.

10- Do comments follow Javadoc style? No What %? 0% of comments follow Javadoc style.

Related Issues:

There is no single comment follows Javadoc style.

11- Is Javadoc generated for all the code? No What %? 0% of code was generated Javadoc for it.

Related Issues:

Javadoc is not generated for all the code.

12- Are there useless / wrong comments? Yes What % 60

Related Issues:

Useless comment like in line 28 in "Home" Class. Also, 55, 73, 85, etc. since they are not-used commented statements!

Code Structure

13- Does the code follow the design precisely? Yes What % 97

Related Issues:

There is some difference between code and design ,for example in design document "Student" class inherited from "Account" class but in code "Student" class inherited from "Teacher" class .

14- Are there very long classes or methods? No

What 5% of classes or methods are long.

Related Issues:

Classes and methods are moderate , some methods can be merged in one method like in class TournamentForm there is a lot of insert function for different input instead of one insert function for all related input.

15- Is there repeated code? (put in a function) Yes

What %5

Related Issues:

InsertQuestion() function repeated in GameForm class and TournamentForm class .

Error Handling

16- Does the code handle errors and exceptions? [✓ Yes] What % 70

Related issues:

It handles errors but there are exceptions were not handled, for example, when entering the category as an input with lowercase while it is required to be written with uppercase there will be no choice for correction. Another thing, in case of having a miss-spell entry, there won't be any exception for modifying it and instead you will have to run the code again and enter the data from scratch.

17- Is defensive programming used to avoid errors? [✓ Yes] What % 100

Related Issues:

Defensive programming used to avoid errors.

Logic

18- Do loops have correct conditions and bounds? [✓ Yes] What % 90

Related Issues:

A bad issue about conditions, that there are some long conditions likes in line 103 and 111 in "AccountModel" class, line 81 in "AccountManager" Class and in line 64 in "TournamentManager" class, etc.

19- Do loops always terminate? [✓ Yes] What % 95

Related Issues:

Loops always terminate.

Overall

20- **Are the design and code of good quality?** [✓ Yes] What % 80

The code is running, usable, good performance and testable. Variables have good names but some variables have too long names. There are no comments to tell how the classes and methods work or their purposes. There are some useless comments. Also, there are some exceptions not handled.

no.	Testing function	Description	Result
1.	signUpTest(a, b, c, d, e, f, g, h, i)	Testing function for sign up function . This test tests the normal sign up scenario	Passed 90%
2.	createGameTest (u , c , n , t , h)	Testing function for create game function. This test tests the normal create game scenario	Failed
3.	playGameTest(category, gameName, userName)	Testing function for run a game. This test tests the normal play game scenario	Failed
4.	testDate(state ,date)	Testing function for check date function. This test tests checking date with different formats .	Failed
5.	testCheckTournamentExist(state , tournament)	Testing function for Check Tournamednt Exist function. This test tests whether the tournament exists or not .	Passed

CS352: Phase 1 – CIS

Phase 1 document

Git repository link HERE