# Assignment 2

# SENG 401

# Gerardo Garcia de Leon 30172099

# Feb 25th, 2025

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Category** | **Level** | **Rating** | **Location**  **(File/line #)** | **Comments** |
| **Data Level** |  |  |  |  |
| Magic Numbers  (instead of constants) | Data | A | Question.java/20 | 5 is used with no context |
| Data members are public | Data | C | GameRecord.java/10, 11, 12, 13, 14  QA.java/8, 9, 10, 11  Question.java/10, 11, 12, 13 |  |
| Out of bounds array indexes | Data |  |  |  |
| Other, Which one? | Data | B | GameFrame.java/25, 91, 92, 110, 405 420, 429 | The array name should be “rewards” not “rewords” causing confusion |
| Other, Which one? | Data | A | QA.java/11 | Can change to a list rather than an array |
|  |  |  |  |  |
| **Statement Level** |  |  |  |  |
| Commented-out code fragments | Statement | A | Question.java/57 | Line is commented out and does not execute |
| Improper or inconsistent naming convention | Statement |  |  |  |
| Incorrect conditions in loops or conditionals | Statement |  |  |  |
| Other. Which one? |  |  |  |  |
|  |  |  |  |  |
| **Routine**  **(Method/Function)**  **Level** | Routine |  |  |  |
| Code is duplicated | Routine | C | GameFrame.java/195, 203, 211, 219, 227, 235 | actionPerformed is repeated in every answer button and can be combined into one action listener |
| A method is too long | Routine |  | GameFrame.java/169-374 | Method is too long and can be split into multiple methods |
| A loop is too deeply nested | Routine |  |  |  |
| Parameter list has too many parameters | Routine |  |  |  |
| A method uses more features of another class than its own | Routine |  |  |  |
| Confusing or incorrect error messages | Routine |  |  |  |
| Unhandled edge cases | Routine |  |  |  |
| Redundant or unnecessary comments | Routine |  |  |  |
| Lack of method headers | Routine |  |  |  |
| Other. Which one? | Routine | A | QuestionLoader.java/26 | Since sets only allow unique values, questionsList.size() is less than number, this causes an infinite loop |
|  |  |  |  |  |
| **Class Level** |  |  |  |  |
| Inappropriate accessibility (public, private, etc.) | Class |  |  |  |
| Inheritance hierarchies must be modified in parallel | Class |  |  |  |
| Changes require  parallel modifications to multiple classes | Class |  |  |  |
| A class has poor cohesion | Class | B | MillionaireConnection.java/83, 114, 137, 158, 179, 200 | Class handles multiple responsibilities such as login, register and retrieving money and records |
| A class doesn’t do very much | Class |  |  |  |
| One class is overly intimate with another | Class |  |  |  |
| Other. Which one? | Class | A | GameFrame.java/52 | answerButtons can be initialized in the beginning rather than inside the constructor |
|  |  |  |  |  |
| **System Level** |  |  |  |  |
| Architecture issue | System |  |  |  |
| Security issue | System |  |  |  |
| Other. Which one? | System | A | MillionaireConnection.java/45 | The catch statement should close the connection as it failed |

PART 2:

**Data level:**

Replaced the magic number in Question.java with a variable that has a proper name. This improves readability and ensures that there is an explanation to the number being used in the code. Without this, it may cause confusion to what the number is meant to represent.

BEFORE:

A white background with black and white clouds

AI-generated content may be incorrect.

AFTER:

A black background with blue and white stripes

AI-generated content may be incorrect.

Deleted the commented line in Question.java to eliminate useless code which was previously crowding the method. By deleting this, we ensure that every line inside the method brings value and is performing an action required to make the program run. Commented-out lines of code do nothing but extend the length of the code and add redundancy.

BEFORE:

A close-up of a white background

AI-generated content may be incorrect.

AFTER:

A black background with colorful text

AI-generated content may be incorrect.

**Routine Level:**

Shortened the method in GameFrame.java to ensure it is not too long. This was done by extracting the other methods and creating separate ones to increase readability and facilitate future maintenance of the code. The code was previously too long to read and was initializing everything in an unorganized manner. With this change, it is now clear and more organized what was changed to the code.

BEFORE (Shortened for sake of space, would require too many screenshots):

A white background with black text

AI-generated content may be incorrect.A white background with black and red text boxes

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

AFTER (Also shortened for sake of space):

A computer screen with blue and green text

AI-generated content may be incorrect.

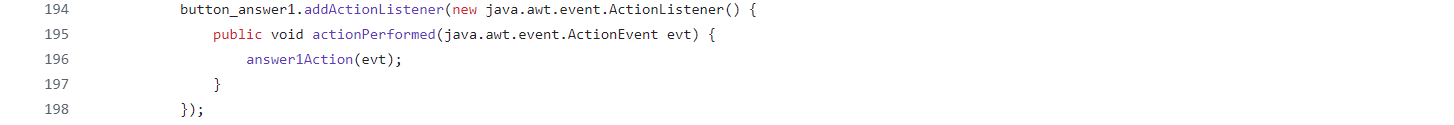


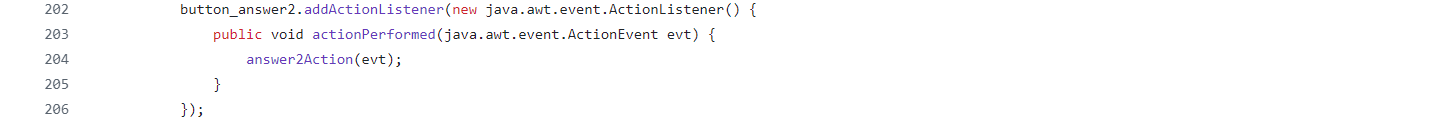
A screen shot of a computer code

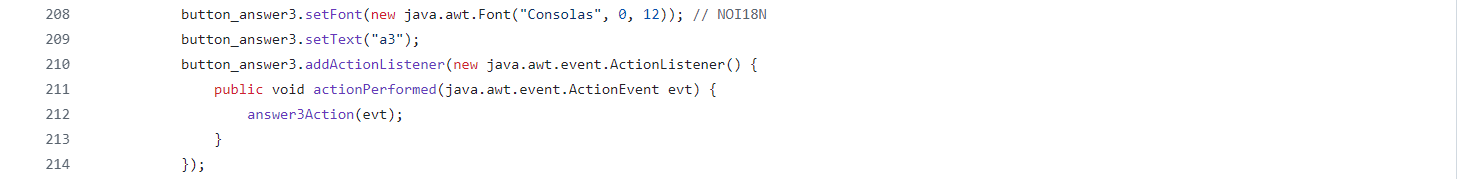
AI-generated content may be incorrect.

Created one common action listener in GameFrame.java and connected them to all buttons which determine the input using a switch statement. This was done to remove the duplicated code in each button for “actionPerformed” and simplified it. The new version of the code is now more readable and eliminated the repetition of code.

BEFORE (Shortened for space):



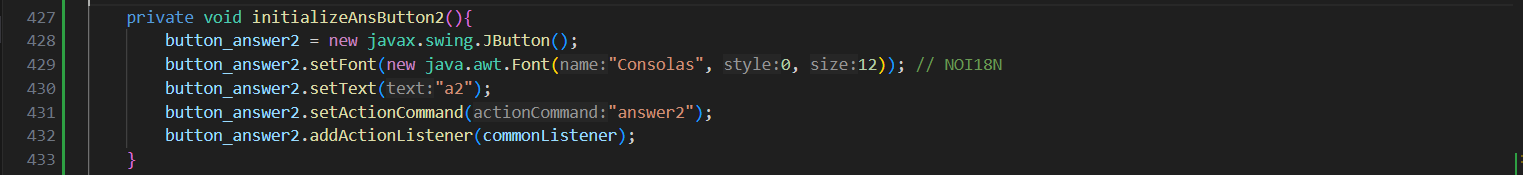


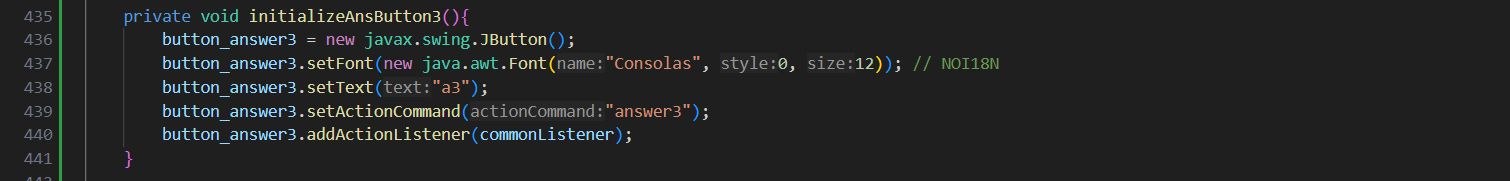


AFTER (Shortened):

A screen shot of a computer

AI-generated content may be incorrect.





**Class Level:**

The MilionaireConnection.java class should be refactored and separated to handle less responsibilities as the current code has very poor cohesion. It currently handles creating the database connection, retrieving the questions, logging in, registering, retrieving a user’s balance, adding records to the database and retrieving previous records. These can all be separated into separate classes that can handle similar tasks to improve cohesion and allow the MillionaireConnection class to handle the database connection and other similar processes. This is necessary as a class with poor cohesion makes it increasingly difficult to maintain a program as it is unorganized, and each class does not perform what it is meant to perform. By improving this, the issue is resolved, and the program can be more easily tested and maintained for future updates.