Final Project: Strong Password Creation Game for Kids

Project Phase 1 Discussion - Feb 7

Phase 1 Submission – Wed Feb 12

Project Phase 2 Discussion - Feb 14

Phase 2 Submission – Feb 19

Project Phase 3 Discussion – Feb 21

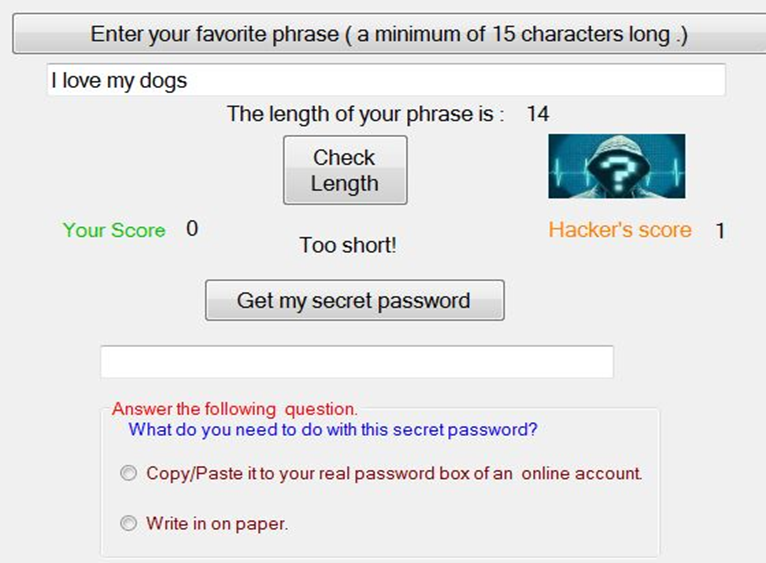
Project Last Phase Discussion – Mar 7

Phase 3 Submission – Mar 19

NOTE that your GUI does not need to be the same as shown below.

Figure 1: A remembered phrase is too short.

Purpose of the game

****

The game is used to teach kids to create a strong non-dictionary password.

Download and play the example game [PasswordGame.zip](https://cbc.instructure.com/courses/2531542/files/263578238?wrap=1). Your game may or may not contain the same features. Don't reduce your creativity based on the example.

# Phase 1: Class Designs and GUI (Feb 2)

Design a graphical user interface (GUI) using the screen above as an example. Work with your team member, and use Word, for instance, to draft a GUI.

## Phase 1 Grading Criteria

* A Graphical User Interface (GUI) is needed for the project. Therefore, state each of the built-in classes needed for such GUI components. (50 points) For example, based on the GUI sample (**use Word or a piece of paper to draw what the GUI looks like)**, there are two click buttons, labels, and text boxes (text fields), etc. Based on your GUI design, state exactly what and how many controls you need.
* Class and interface designs (more than one class and at least one interface and abstract class). The classes must use inheritance and object composition (Aggregation). Also, class fields and methods are logical. (25 points)
* Class UML diagrams of the classes (25 points)
  + Refer to the example diagram on how to show aggregation/composition and inheritance. Use of an interface is also denoted by an inheritance arrow.
* A Graphical User Interface (GUI) is needed for the project. Therefore, state each of the built-in classes needed for such GUI components. (50 points)

For example, based on the GUI sample, there are two click buttons, labels, and a text boxes (text fields), etc. Based on your GUI design, state exactly what and how many controls you need.

* Class and interface designs (more than one class and at least one interface and abstract class). The classes must use inheritance and/or object composition (Aggregation). Also, class fields and methods are logical. (40 points)
* The information of your class designs are presented in class UML diagrams (30 points)
* If you are planning to use dynamic binding call it out because I will give you extra credit in the last phase if you implement it.

**What to turn in : A document file that contains your project GUI (a picture if hand-drawn), and class UML diagrams (just your classes excluding the main() ).**

# Phase 2: Class Implementations

## Phase 2 Grading Criteria

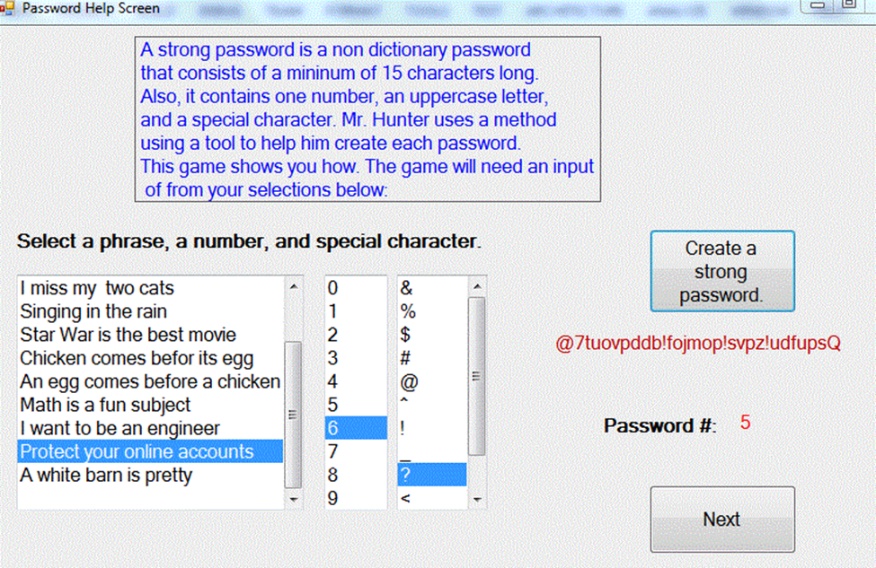
* All class methods of the super class and constructor(s) are implemented. The most important logic is your algorithm of strong password creation. (40 points)
* A minimum of one method and one constructor of a subclass are implemented (40 points)
* Use the main ( ) to test if the implemented code works. (20 points). Hint. You may use the main ( ) without using the GUI for this phase of the project.

What to turn in: A word document that contains all of your java files. Each class must have its methods and implementations. The main( ) also must test your classes. Also, submit screenshots of the program outputs thus far.

# Phase 3: File Processing/ Databases

## Phase 3 Grading Criteria

* All class methods of all sub classes and constructors are implemented (40 points)
* Implement your GUI coding (20 points).
* A modified GUI is as follows:



* Store each remembered phrase on disk using a database or a random access file. The user is not required to enter phrases, but instead provide them (see above). At least three phrases will be selected. (20 points)
* Use the main ( ) to test the application (20 points)

What to turn in: Your source code, a text file created, and screenshots of your outputs.

# Phase 4: Playing a game

## Phase 4 Grading Criteria

* Test and revise the application to do the required tasks stated in first three phases. (70 points)
* If you are going to use two screens, how would you do it? The new screen displays the remembered phrases entered from the first screen.(10 points)
* Implement the application to do this task. (20 points)

**Detailed grading criteria for the final project (Phase 4).**

**The following information will be emailed to you as well.**

1. (5 points) Copy and paste every line of code in the project into a Word document in which the classes and/or interfaces that manage passwords and scores will be displayed first. These two classes are not one that extends Application or JPanel or JFrame. Then copy the classes that contain GUI code components followed by the main ( ).

(5 points) UML class diagrams that align with the code implementations.

1. (10 points) Highlight the lines of code that demonstrate a class relationship (“is a “and/or “has a “relationship) between two classes and/or interfaces. Below the code as a comment, **use dark green text for an explanation about the class relationship. Explain if your UML class diagrams align with the implementations. Also, state if there is anything you would have done better and any code duplications in these classes that could be revised.**
2. (10 points) Highlight the lines of code that demonstrate where you declare and instantiate the objects of the classes that manage data (passwords and scores/games). Below the code as a comment, use blue text for an explanation about the required tasks and how they work.
3. (10 points) Highlight the lines of code that demonstrate where you save the information on disk.  Below the code as a comment, use brown text for an explanation about the required tasks and how they work. Is the file access binary and random access or a database and did you retrieve the information from disk?

(5 points) files on disk- must be submitted.

1. (10 points) Highlight the lines of code that show class methods that create strong passwords. Below the code as a comment, use blue text for an explanation where string inputs come from and how they are manipulated.
2. (10 points) Highlight the lines of code that show GUI for a game-play (as supposed to be done in phase 1). Below the code as a comment, use red text for an explanation where string inputs come from and how they are manipulated.
3. (10 points) Highlight the lines of code that show GUI for string selections used to pass to class methods for strong password creation (as supposed to be done in phase 1). Below the code as a comment, use purple text for an explanation where string inputs come from and how they are manipulated.
4. (10 points) Highlight the lines of code that show the second window and its implementation in Phase 4.
5. (5 points) **Provide two screenshots of outputs** that show results of a game-play.
6. (5 points) **Provide two screenshots of outputs** that show results of password created.