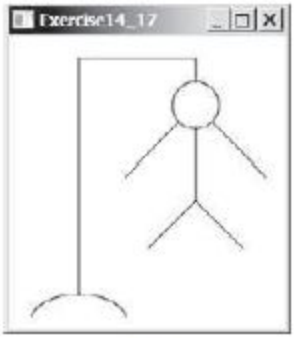
Homework 5
75 points
Due Friday March 22nd, 11:59pm

Hangman - Full Game

This programming assignment is a continuation of Homework 3, exercise 14.17.

14.17 (*Game: hangman*) Write a program that displays a drawing for the popular hangman game, as shown in Figure 14.48a.



(a)

You are allowed, even encouraged, to use your code from Homework 3. This assignment will add the following features:

1. Your JavaFX application must include "Board" with a guest phrase, loaded from a file phrases.txt, provided on the Canvas -> Assignments -> HW5 - Hangman Page. Phrases.txt includes multiple phrases, one per line. Select a random phrase, to create a board. While you are testing, you may want to use the same phrase each time, but you should implement the random phrase logic before you submit your final version.
2. You must have a Text Box for the user to enter a letter, to guess. Also include a Label next to the Text Box, so it is clear to the user.
3. The board should be blank to start. After each correct guess, update the board to show the letters that match the user's guess letter. Also output somewhere on the application "Correct!" after each correct guess. I will not add any more requirements to how you should build your board, other than make it clear, intuitive, and look good.
4. If the user's guess letter is not on the board, output "Incorrect!" somewhere on the application, and then draw one of the hangman's body parts.
5. The Hangman should include six body parts: head, torso, 2 arms, and 2 legs. Also include a symbol of any rival university of your choice, on the torso of the hangman. The

rival university symbol should appear with the hangman's torso- it is not to be drawn as a separate body part.

6. After 6 incorrect guesses, the full hangman should be displayed. Also display a losing message, and then allow the user the option to play again.
7. After the user guesses all letters on the board correctly, display a friendly winning message, and then allow the user to play again.
8. If the user chooses to play again, reset the board with a new random phrase from phrases.txt, and start the game again.
9. Export your project as a windows executable file or an executable jar, and include it in your zip file.

Submit to Canvas:

Submit 1 zip file containing all the files in your src folder, and your executable file. Name all your files clearly, so the grader can easily see, and run your program.

Scoring rubric:

The following rubric will be used:

Criteria	**% Points**
Program(s) fulfill all the requirements. All .java files are included and declare the necessary classes. Code is well organized, and easy to follow (especially for the grader). Coding stye is well utilized, including well named variables and methods. Comments are included, well written and descriptive.	100%
Program(s) fulfill almost all of the requirements. All .java files are included and declare the necessary classes. Code is fairly well organized, and somewhat easy to follow (especially for the grader). Some comments are included.	80%
Program(s) fulfill most of the requirements. All .java files are included and declare the necessary classes. Code is fairly well organized, and somewhat easy to follow (especially for the grader). Some or no comments are included.	60%
Program(s) fulfills some of the requirements, or does not run at all. Some .java files are included and declare some of the necessary classes. Some or no comments are included.	40%

Program(s) does not run at all. Some .java files are included and declare some of the necessary classes. Some or no comments are included.	20%
Either no attempt was made, or the attempt made shows no progress toward solving the problem.	0%