This homework is due in class on **Monday, November 6** (no extensions). You will have lab time on October 30 and November 1 to work on it.

What to submit: Submit your code both electronically (via Blackboard) and in hard-copy form.

As always, legibility counts: if your code is difficult to read or unnecessarily complicated, you may lose points. ***In addition:*** *You should be using only the constructs we've discussed in lecture and labs. Do not use iterators or other advanced features or libraries of Python.*

Your Tasks

This assignment in based on material through Chapter 8 of the textbook. You will need to modify the word-guessing game from Lab 12 to account for the following extensions:

1. Before each guess, the user may elect to solve the puzzle instead. If so, the user is prompted for their answer, which is compared with the true solution:

   - If the user is incorrect, they lose.
   - If they're correct, they win the amount of money in the prize pool (see next item).

   When checking the user's answer, do not distinguish between uppercase and lowercase characters. For example, `MagIC` *should be considered the same as* `magic`*.*

2. The program no longer keeps track of a user's guesses. Instead, the game starts out with a prize pool of $1500: each time the user guesses a letter, that balance is reduced by either $100 (if the letter is in the puzzle and hadn't previously been guessed) or $150 (if the letter is not in the puzzle, or if it had already been guessed on a previous turn). If this balance ever reaches (or goes below) $0, the game is over, and the user loses.

   For example, suppose the user's first guess is 'A', which is in the solution: the balance dips to $1400. If the user's next guess is 'K' (which is not in the solution), the balance dips to $1250. If the user then guesses 'A' again, the balance dips to $1100. If the user then decides to solve the puzzle and gets it correct, she wins $1100.

3. Once you have the first two items working correctly, it's time to improve the user interface.

   Use the `graphics` package to create a window that displays the current balance and status of the puzzle: each time the user guesses a letter, both the balance and status are updated accordingly.

   In addition, the user should be able to select (via mouse) whether to solve the puzzle or guess another letter, and then be guided (by some sort of text entry box) to enter his solution/guess.

   It's up to you how you want to set this up, but all user interaction should be via the graphical interface. *Important note: Despite the extra code necessary for the graphical interface, keep the main function relatively clean. Introduce functions as a way of minimizing the "junk" in your main function.*

Some notes and suggestions:

- The first two tasks are relatively straightforward: get them working correctly before moving on to the third. You can then move the various user-interaction components to the graphical interface one at a time.
- The grader will be running your programs to try things out. Make sure it's straightforward for a user to figure out how to interact with your program.