CSCI 1133, Fall 2018

Programming Examination 1

Wheel of Fortune!
100 pts

Part 1 Due: 5:00 pm, Friday, Oct 19
Part 2 Due: 5:00 pm, Friday, Oct 26

PLEASE NOTE: This is a graded assessment of individual programming understanding and ability, and is ***not*** a collaborative assignment; you must design, implement and test the solution(s) completely on your own without outside assistance from anyone. You may not consult or discuss the solution with anyone. In addition, you may not include solutions or portions of solutions obtained from any source other than those provided in class. Note that *providing* a solution or assisting another student in any way on this examination is also considered academic misconduct. Failure to heed these directives will result in a failing grade for the course and an incident report filed with the Office for Student Conduct and Academic Integrity for further sanction.

This examination consists of two parts. The first part involves creating a short text file that must be committed and pushed to your GitHub examination repository (discussed below) prior to the Part 1 deadline on October 19. The second part requires designing and writing a single well-structured Python program that must be committed and pushed to your GitHub examination repository *prior* to the Part 2 deadline on October 26.

Late submissions will not be accepted and will result in a zero score for the exam.

TA help for this examination **will not** be provided. If you have clarification questions, they must be addressed to a graduate TA for the class.

The total point value will be awarded for solutions that are *complete*, *correct*, and *well structured*. A *"well structured"* program entails good design that employs functional decomposition, appropriate comments and general readability (descriptive names for variables and procedures, appropriate use of blank space, etc.) If you are not sure what this means, review the "well-structured" program requirements provided in Lab2.

Note that your work will be graded using, and must function correctly with, the current version of Python 3 on CSE Labs UNIX machines. If you complete this programming exam using a different system, it is *your* responsibility to ensure it works on CSELabs machines prior to submitting it.

The rubric includes the following specific (accumulative) point deductions:

- Missing academic integrity pledge -100 points
- Syntax errors -50 to -100 points
- Misnamed source file or incorrect repository -25 points

- Use of disallowed global variables -25 points
- Missing main program function -25 points

Examination Repository

Examination files must be submitted to GitHub using a special "exam repository". Exam repositories have already been created for each registered student and are named using the string `exam-` followed by your X500 userID (e.g., `exam-smit1234`). You must first clone your exam repository in your local home directory before submitting the materials for this exam. If you are having difficulty, consult the second Lab from earlier in the semester. If your exam repository is missing or something is amiss, please contact a graduate TA. **DO NOT SUBMIT YOUR EXAM FILES TO YOUR LAB/ EXERCISE REPOSITORY!**

Part 1 (*10 points*)

Using a text editor, type the following academic integrity pledge (exactly as it appears), replacing the last line with your full name and X500 ID. Save the text file with the name `academicpledge.txt` and commit/push it to your GitHub examination repository:

> I understand this is a graded, individual examination that may not be discussed with anyone. I also understand that obtaining solutions or partial solutions from outside sources or discussing any aspect of the examination with anyone will result in failing the course.
>
> < replace this line with your name and x500 ID (e.g., John Smith smit1234) >

If you do not commit and push the `academicpledge.txt` file prior to the deadline for Part 2, your entire examination solution will not be graded, resulting in a score of zero. In order to receive the 10 points for Part 1, you must submit the pledge file to your exam repository prior to the deadline for part 1.

Part 2 (*90 points*)

Write a well-structured Python program to create a simplified, interactive one-player version of Wheel of Fortune.

If you've never seen Wheel of Fortune, you can get an idea of how the game is played here: https://www.youtube.com/watch?v=A8bZUXi7zDE

In our version of Wheel of Fortune, the game will proceed as follows:

General Overview

In Wheel of Fortune, users will attempt to accumulate as much money as possible by "spinning a wheel" and guessing letters in an attempt to figure out a secret phrase. Money will be awarded to the player for correct guesses, and deducted for incorrect guesses. The game ends when the player correctly guesses the phrase or has guessed all possible letters

Beginning the Game

At the beginning of a game, your program should randomly select a puzzle **phrase** (see below) and identify its corresponding **category** (see below). Your program should output "Welcome to the Wheel of Fortune!" and also print the randomly selected puzzle phrase to the screen with "blanks" (underscore characters) in place of the letters. Below the puzzle phrase, your program should also output the category of the phrase to the screen.

Throughout the game, the user will be earning their "winnings" by spinning a wheel and correctly guessing letters. At the beginning of the game, a user's "winnings" should be set to $0, and the player's winnings should also be output to the screen.

For example, suppose the phrase 'SOMEWHERE OVER THE RAINBOW CONNECTION' is randomly selected by the program at the start of the game. This phrase has a corresponding category of "Before and After." The following information should be printed to the screen:

```
Welcome to the Wheel of Fortune!

The phrase is:
_ _ _ _ _ _ _ _ _   _ _ _ _   _ _ _   _ _ _ _ _ _ _   _ _ _ _ _ _ _ _ _ _

The category is:  Before and After

Your current winnings are: $0
```

- **Phrases**: We have compiled a list of phrases into a file called `phrasebank.txt`. You should save this file in the same directory ("location") as your program file (when we test your program, we will save this file in the correct location — you do *not* need to submit `phrasebank.txt` with your solution). To use this file, you should include the line:

  ```
  PhraseBank = open("phrasebank.txt").read().splitlines()
  ```

  into your code. This will create a structure, called PhraseBank, that stores a large variety of phrases. For example, `PhraseBank[0]` contains the object 'SOMEWHERE OVER THE RAINBOW CONNECTION' — this is the first phrase in the phrase bank. There are 100 total phrases in the bank. You may assume phrases include only the 26 letters in the English alphabet (no numbers or special characters), do not have any punctuation (for example, words like DON'T and O'DELL will not appear), and commas, question marks, exclamation points, etc. are excluded. You may assume that all phrases will be in all caps.

- **Categories**: There are 5 total categories in our version of this game. Phrases are stored in the `PhraseBank` in order: the first 20 entries in `PhraseBank` are from the category "Before and After," the second 20 entries are from the category "Song Lyrics", the third 20 entries are from the category "Around the House", the fourth 20 entries are from the category "Food and Drink", and the last 20 entries are from the category "Same Name."

For example, if you randomly select the entry stored in `PhraseBank[75]`, that phrase will be from the category "Food and Drink".

After the phrase has been selected and printed to the screen with the appropriate blanks (along with the category and current winnings), the user should be asked if they would like to **Spin the Wheel** (see below), **Buy a Vowel**, or **Solve the Puzzle**.

Following our example above, the screen would look like this (user input is typed in red)

```
Welcome to the Wheel of Fortune!

The phrase is:
_ _ _ _ _ _ _ _ _   _ _ _ _   _ _ _   _ _ _ _ _ _ _   _ _ _ _ _ _ _ _ _ _

The category is:  Before and After

Your current winnings are: $0

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type
'vowel'), or Solve the Puzzle (type 'solve')? spin
```

Spinning the Wheel

If the user has typed `spin`, it means the user wishes to "Spin the Wheel" and guess a letter in the puzzle. A function called **spinTheWheel** should be called. **spinTheWheel** should perform the following actions:

- **Randomly Select a Wheel Value:** A "spin of the wheel" should randomly select one of the following 24 values: $50, $100, $100, $100, $100, $100, $100, $200, $200, $200, $200, $250, $250, $250, $500, $500, $750, $750, $1000, $2000, $5000, $10000, Bankrupt, Bankrupt. The amount the player has spun should be output to the screen. **IMPORTANT:** If a player spins a "Bankrupt", their current winnings should be erased and re-set to $0. The user should not be allowed to guess a consonant; their turn is over.

- **Ask the user to guess a consonant**: If the player has not "spun" a Bankrupt, they should be asked to guess a consonant (consonants are all letters in the English alphabet except for A, E, I, O, U). If the user *correctly* guesses a consonant in the phrase, they should have the following added to their current winnings: `(the number of occurrences of that consonant)*(value of the wheel spin)`. If, however, the user guesses *incorrectly* (the consonant does not appear in the phrase) the dollar amount of the spin should be *deducted* from their winnings. The player should be informed if the letter they guessed is in the phrase, and if so, how many times. The should also be informed how much they have won (or lost) on the turn.

- **Return the user's winnings, the consonant guessed, and if the guess was correct:** The function should return the user's updated total winnings, the consonant the player guessed, and if their guess was in the phrase.

- **HINT**: **spinTheWheel** should have two input arguments: the puzzle solution (this is the random phrase drawn at the start of the game) and the player's current balance *before* the turn is taken

After the player has completed a spin turn, the phrase should be printed with all correctly guessed letters in place of blanks, the category should be re-printed, the vowels guessed so far should be printed, the consonants guessed so far should be printed, the current winnings should be printed, and the player should be asked what they would like to do next.

Example (user input is in red, newly added code begins at the arrow)

```
Welcome to the Wheel of Fortune!

The phrase is:
_ _ _ _ _ _ _ _ _   _ _ _ _   _ _ _   _ _ _ _ _ _ _   _ _ _ _ _ _ _ _ _ _

The category is:  Before and After

Your current winnings are: $0

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type
'vowel'), or Solve the Puzzle (type 'solve')?  spin

You've spun $200!  Guess a letter: R

Congratulations, R appears in the phrase 3 times!  You've won $600.

The phrase is:
_ _ _ _ _ _ _ R _   _ _ _ R   _ _ _   R _ _ _ _ _ _   _ _ _ _ _ _ _ _ _ _
```

```
Vowels Guessed:
Consonants Guessed: R
Your current winnings are: $600

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type
'vowel'), or Solve the Puzzle (type 'solve')? vowel
```

Buying a Vowel

If a user types `vowel`, the player has opted to "buy a vowel." Since A, E, I, O, and U are common letters in English words, in order to guess one of these letters the user must pay $250. The price to buy a vowel is *always* $250, regardless of how many times the vowel appears in the phrase (even if the vowel does not appear at all!). Note that if a user chooses to buy a vowel, they must have accumulated at least $250. Since users start the game with $0, this means a user cannot buy a vowel

on their first turn. When the user types `vowel`, a function named **buyAVowel** should be called. **buyAVowel** should perform the following actions:

- **Check the player's winnings:** A user must have at least $250 in their winnings to buy a vowel. If the user does not have at least $250, they should not be allowed to proceed with buying a vowel.
- **Ask the user which vowel they would like to buy:** The user should be informed that $250 will be deducted from their winnings, and asked which vowel they would like to buy. If the vowel selected by the player appears in the phrase, the player should be informed how many times the letter appears. If it does not appear, they should be informed of this as well.
- **Update the user's winnings:** Regardless of whether the vowel appears in the phrase, $250 should be deducted from the user's winnings.
- **Return the user's winnings, the vowel guessed, and if the guess was correct:** The function should return the user's updated winnings, the vowel the player guessed, and if their guess was in the phrase.
- **HINT**: **buyAVowel** needs two input arguments: the puzzle solution (this is the random phrase drawn at the start of the game) and the player's current balance *before* the turn is taken

After the player has completed buying a vowel, the phrase should be printed with *all* correctly guessed letters in place of blanks, the category should be re-printed, the vowels guessed so far should be printed, the consonants guessed so far should be printed, the current winnings should be printed, and the player should be asked what they would like to do next.

Example (user input is in red, newly added code begins at the arrow)

```
Welcome to the Wheel of Fortune!

The phrase is:
_ _ _ _ _ _ _ _ _   _ _ _ _   _ _ _   _ _ _ _ _ _ _   _ _ _ _ _ _ _ _ _ _

The category is:  Before and After

Your current winnings are: $0

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type
'vowel'), or Solve the Puzzle (type 'solve')?  spin

You've spun $200!  Guess a letter: R

Congratulations, R appears in the phrase 3 times!  You've won $600.

The phrase is:
_ _ _ _ _ _ _ R _   _ _ _ R   _ _ _   R _ _ _ _ _ _   _ _ _ _ _ _ _ _ _ _

Vowels Guessed:
```

```
Consonants Guessed: R
Your current winnings are: $600

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type
'vowel'), or Solve the Puzzle (type 'solve')? vowel

Ok!  $250 will be deducted from your winnings.  Which vowel would you like
to buy (A, E, I, O, U)?:  E

Congratulations!  E appears in the phrase 6 times!

The phrase is:
_ _ _ E _ _ E R E   _ _ E R   _ _ E   R _ _ _ _ _ _   _ _ _ _ E _ _ _ _ _

Vowels Guessed: E
Consonants Guessed: R
Your current winnings are: $350

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type
'vowel'), or Solve the Puzzle (type 'solve')?
```

Solving the Puzzle

If the user types `solve`, this means they would like try and guess the puzzle phrase. A function named **solveThePuzzle** should be called. **solveThePuzzle** should perform the following actions:

- **Ask the user what their guess is:** The user should be prompted to enter a guess for the phrase. The user should be asked to enter their response with single spaces.

- **Check to see if the guess matches the phrase:** The player's guess should be checked against the original phrase. If the guess is correct, the user has won - a message should be output to the user indicating they have correctly solved the puzzle. If the user has not correctly guessed the puzzle, they should be informed their guess was incorrect and their winnings should be re-set to $0.

- **Return the user's winnings and if the guess was correct:** The function should return the user's winnings and if they correctly guessed the phrase.

At this point, the user has either correctly guessed the puzzle and reached the End of Game (see below), or incorrectly guessed the puzzle. If they have incorrectly guessed the puzzle, the phrase should be printed with *all* correctly guessed letters in place of blanks, the category should be re-printed, the current winnings should be printed, and the player should be asked what they would like to do next.

Example (user input is in red, newly added code begins at the arrow)

```
Welcome to the Wheel of Fortune!
```

```
The phrase is:
_ _ _ _ _ _ _ _ _   _ _ _ _   _ _ _   _ _ _ _ _ _ _   _ _ _ _ _ _ _ _ _ _

The category is:  Before and After

Your current winnings are: $0

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type
'vowel'), or Solve the Puzzle (type 'solve')?  spin

You've spun $200!  Guess a letter: R

Congratulations, R appears in the phrase 3 times!  You've won $600.

The phrase is:
_ _ _ _ _ _ _ R _   _ _ _ R   _ _ _   R _ _ _ _ _ _   _ _ _ _ _ _ _ _ _ _

Vowels Guessed:
Consonants Guessed: R
Your current winnings are: $600

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type
'vowel'), or Solve the Puzzle (type 'solve')? vowel

Ok!  $250 will be deducted from your winnings.  Which vowel would you like
to buy (A, E, I, O, U)?:  E

Congratulations!  E appears in the phrase 6 times!

The phrase is:
_ _ _ E _ _ E R E   _ _ E R   _ _ E   R _ _ _ _ _ _   _ _ _ _ E _ _ _ _ _

Vowels Guessed: E
Consonants Guessed: R
Your current winnings are: $350

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type
'vowel'), or Solve the Puzzle (type 'solve')? solve

What's your best guess (be sure to enter your guess with single spaces!)?
Somewhere Over The Rainbow Skittles

Sorry, that guess is incorrect!  Your winnings will start over at $0 :(

The phrase is:
_ _ _ E _ _ E R E   _ _ E R   _ _ E   R _ _ _ _ _ _   _ _ _ _ E _ _ _ _ _

Your current winnings are: $0
```

```
Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type
'vowel'), or Solve the Puzzle (type 'solve')?
```

End of Game

The game ends when a user has either correctly guessed the puzzle (they have won!), or when they have guessed all 26 letters in the alphabet (they have lost). If the user has won, the user should be congratulated for winning the game and their winnings should be printed to the screen. If the user has lost, their winnings should be set to $0, and a message of condolences should be printed to the screen along with their current winnings of $0.

IMPORTANT: to end the game, a user must "solve the puzzle" even if they have correctly guessed all the letters in the word! The only exception to this is if the user has guessed all 26 letters in the alphabet - at this point, the user has automatically "lost."

Program Requirements

Your program must follow all of the guidelines outlined above. Specifically, your program must:

- Include and correctly implement the functions **spinTheWheel, buyAVowel,** and **solveThePuzzle** as described above
- Correctly print the puzzle phrase with appropriate "blanks" (underscores) at the start of the game, and after the end of each turn. HINT: we suggest writing a function called **phrasePrint** that will print underscores for letters that have not been guessed by the user, and letters for letters that have been guessed by the user. WARNING: spaces between words in a phrase should NOT be underscores.
- Correctly print the vowels a user has guessed so far after each turn
- Correctly print the consonants a user has guessed so far after each turn
- Correctly update the player's winnings during each turn, and correctly display the winnings after the end of each turn
- Prompt the player to continue taking turns until they have reached the "End of the Game" as described above (either by correctly solving the puzzle, or by guessing all 26 letters).
- Correctly implement the "End of Game" as described above

Constraints

- You may use any *built-in* Python object class methods (string, list, etc.)
- You may use *imported* functions and class methods from the random module only
- You may use any built-in Python functions/operations as appropriate
- You may assume the user does not guess the same letter twice
- You should not assume the user correctly guesses consonants and vowels where appropriate (HINT: create an object containing all vowels, for example, and see if the user's guess for a vowel is IN that object: see the membership test `in` here, which we will also discuss in class: https://docs.python.org/3/reference/expressions.html#membership-test-operations)
- You may not assume the user correctly uses uppercase letters or phrases in their guesses, but the player should NOT be penalized for that — for example, both "Somewhere over the rainbow connection" and "SOMEWHERE OVER THE RAINBOW CONNECTION" should be

correct solutions to the puzzle phrase in the example (hint: use the string method `upper()` described here: https://docs.python.org/3/library/stdtypes.html#string-methods). The user should also be allowed to guess a vowel 'e' and have this be considered a valid guess.

- You may not assume the user correctly types `spin`, `vowel`, or `guess`
- In general, you may not use global variables. Exceptions: you may create, if you wish, a global variable called WHEEL containing the values on a Wheel of Fortune wheel as described above, and a global variable VOWEL_COST = 250. You may also use global variables CONSONANTS and VOWELS (strings of all the consonants and all the vowels, respectively).

Submission Instructions

Store all of your source code in a single module named wheeloffortune.py Make sure your files are named correctly! Commit/push your source file to your exam repository *prior* to the deadline for Part 2.

Helpful Hints:

- You will find this project much easier if you employ the principles of "top-down, functional decomposition" as discussed in lecture and implement your program as a collection of short, simple functions. Although we require you to use 3 specific functions, you may find it useful to construct others and are encouraged to do so.
- You are allowed and encouraged to use Python3 documentation: **https://docs.python.org/3/contents.html**
- Solve this problem with pencil and paper first! There are several moving parts.
- Try to implement this in stages - get a simplified version of the game working first before moving onto the entire thing. For example, try to get a version working where you assume the player enters correct input, and then worry about fixing the cases where the player enters input incorrectly.
- Test out several different games, including several where the user makes "mistakes."
- Make sure to consider special cases. For example, if a player spins "Bankrupt," they cannot guess a letter following this spin.
- Think of all the different "structures" you will need in this game and plan out how you would like to implement them. For example, you will find it helpful to have a structure that stores vowels guessed, and another that stores consonants guessed.
- Start Early!

SAMPLE GAME (user input in RED)

```
Welcome to the Wheel of Fortune!

The phrase is:
_ _ _ _ _ _ _ _ _   _ _ _ _   _ _ _   _ _ _ _ _ _ _   _ _ _ _ _ _ _ _ _ _

The category is:  Before and After
```

Your current winnings are: $0

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type 'vowel'), or Solve the Puzzle (type 'solve')? spin

You've spun $200! Guess a letter: R

Congratulations, R appears in the phrase 3 times! You've won $600.

The phrase is:
_ _ _ _ _ _ _ R _ _ _ _ R _ _ _ R _

Vowels Guessed:
Consonants Guessed: R
Your current winnings are: $600

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type 'vowel'), or Solve the Puzzle (type 'solve')? vowel

Ok! $250 will be deducted from your winnings. Which vowel would you like to buy (A, E, I, O, U)?: E

Congratulations! E appears in the phrase 6 times!

The phrase is:
_ _ _ E _ _ E R E _ _ E R _ _ E _ _ _ _ _ _ _ _ _ _ _ _ E _ _ _ _ _

Vowels Guessed: E
Consonants Guessed: R
Your current winnings are: $350

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type 'vowel'), or Solve the Puzzle (type 'solve')? solve

What's your best guess (be sure to enter your guess with single spaces!)?
Somewhere Over The Rainbow Skittles

Sorry, that guess is incorrect! Your winnings will start over at $0 :(

The phrase is:
_ _ _ E _ _ E R E _ _ E R _ _ E _ _ _ _ _ _ _ _ _ _ _ _ E _ _ _ _ _

Your current winnings are: $0

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type 'vowel'), or Solve the Puzzle (type 'solve')? Fly

Whoops, I don't recognize that input! Try again.

```
Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type
'vowel'), or Solve the Puzzle (type 'solve')? spin

You've spun $1000!  Guess a letter: c

Congratulations, C appears in the phrase 2 times!  You've won $2000.

The phrase is:
_ _ _ E _ _ E R E   _ _ E R   _ _ E   R _ _ _ _ _ _   C _ _ _ E C _ _ _ _

Vowels Guessed: E
Consonants Guessed: R C
Your current winnings are: $2000

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type
'vowel'), or Solve the Puzzle (type 'solve')? spin

Uh oh, you've spun a Bankrupt!  Your winnings will go down to $0.

The phrase is:
_ _ _ E _ _ E R E   _ _ E R   _ _ E   R _ _ _ _ _ _   C _ _ _ E C _ _ _ _

Vowels Guessed: E
Consonants Guessed: R C
Your current winnings are: $0

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type
'vowel'), or Solve the Puzzle (type 'solve')? vowel

Sorry, you don't have enough winnings to buy a vowel!

The phrase is:
_ _ _ E _ _ E R E   _ _ E R   _ _ E   R _ _ _ _ _ _   C _ _ _ E C _ _ _ _

Vowels Guessed: E
Consonants Guessed: R C
Your current winnings are: $0

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type
'vowel'), or Solve the Puzzle (type 'solve')? spin

You've spun $500!  Guess a letter: T

Congratulations, T appears in the phrase 2 times!  You've won $1000.

The phrase is:
_ _ _ E _ _ E R E   _ _ E R   T _ E   R _ _ _ _ _ _   C _ _ _ E C T _ _ _

Vowels Guessed: E
Consonants Guessed: R C T
Your current winnings are: $1000
```

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type 'vowel'), or Solve the Puzzle (type 'solve')? spin

You've spun $200! Guess a letter: I

Whoops, that's not a consonant! Guess again: F

Sorry, F does not appear in the puzzle. You will have $200 deducted from your winnings.

The phrase is:
_ _ _ E _ _ E R E _ _ E R T _ E R _ _ _ _ _ _ C _ _ _ E C T _ _ _

Vowels Guessed: E
Consonants Guessed: R C T F
Your current winnings are: $800

Would you like to Spin the Wheel (type 'spin'), Buy A Vowel (type 'vowel'), or Solve the Puzzle (type 'solve')? solve

What's your best guess (be sure to enter your guess with single spaces!)?
Somewhere Over the rainbow connection

That's correct - you solved the puzzle!

Congratulations, you've won the game! Your winnings are $800. Thank you for playing the Wheel of Fortune!