Aryan Aziz

Assignment 6 Report

Exercises

List of Files:

A6E1.cpp
A6E2.cpp
A6E3.cpp
A6E4.cpp
A6E5.cpp
A6P1.cpp

Projects

Understanding:

This project, luckily, was very similar to last week's project. However, the biggest difference is that there is no input for the word from the user, instead it pulls that info from a predefined list. On top of this, instead of being a single word, it's a full phrase selecting 3 random numbers from the array.

The idea behind the game is that it starts out by selecting 3 words randomly from a list of pre-defined words and then has the user guess the phrase 1 letter at a time until they either guess the word or run out of guesses.

Design:

Attached are all PDFs for all exercises.

I, admittedly, re-used my previous week's assignment for the backbone of my code. Since all of the code is already written out for the guessing and design, the only change that needed to be made was to get rid of the function asking for the user to enter a word to guess and instead create an array with a predefined list of words (for some reason the phrase I kept thinking was "the man goes to the beach and got some food to eat while running"….so that became my phrase).

Then, from that list, it was to randomly select 3 words from the list. It's doing this by using the random number phrase and checking if the number has been previously used or not.

Testing:

Pretty much all testing for this project was done in the previous week. Of the new stuff, I immediately noticed that my random number loop was actually just taking the first random number and returning that as the word everytime. I had to re-create the loop and have it check on itself if the word was created or not.

Reflection:
Interestingly enough, the most difficult part of this was getting the random phrase. The problem being, with such a small group of words it selects the same random number a lot. Even now, it is selecting 3 unique numbers from the group but it definitely takes a little bit of time to do so. I'm not sure if there is a more efficient way of doing this.

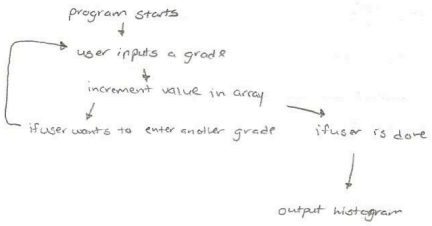
I like the idea of using arrays to track if values have been used. It's relatively simple and cleaner than searching for a string/integer for a certain value; you can instead check for that specific spot in the array.

CS163 Assignment 6

3. User inputes grades (0-5)

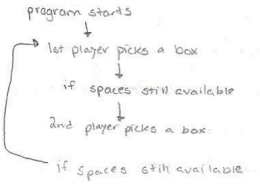
- counts the # of 0's, 1's...
- handle any # of grades

$\{0, 1, 2, 3, 4, 5\}$



4. Tic-Tac-Toe

2 players



CS165 Assignment 6

- Project.
  - Array of words
    - Select 2-5 words from array @ random
    - user guesses 1 letter @ a time
      - show what they have guessed so far
      - list valid letters
      - tell user how many guesses they have left
  - let them play again