Assignment 2 – Hangman Report Template

Your Name

CSE 13S – Winter 24

Purpose

Audience for this section: Pretend that you are working in industry, and write this paragraph for your boss. You are answering the basic question, "What does this thing do?". This section can be short. A single paragraph is okay.

Do not just copy the assignment PDF to complete this section, use your own words.

This program simulates the game Hangman. There is a secret phrase and the goal is to figure it out. You do this by guessing letters. If a letter you guessed is in the phrase, you can see its relative position and its occurrences in the phrase. If not, you get a mistake. Too many mistakes and you lose the game.

Questions

Please answer the following questions before you start coding. They will help guide you through the assignment. To make the grader's life easier, please do not remove the questions, and simply put your answers below the text of each question. To fill in the answers and edit this file, you can upload the provided zip file to overleaf by pressing [New project] and then [Upload project].

Guesses

One of the most common questions in the past has been the best way to keep track of which letters have already been guessed. To help you out with this, here are some questions (that you must answer) that may lead you to the best solution.

- How many valid single character guesses are there? What are they?

    - In total, there are 26 valid character guesses. They are the letters of the alphabet in lower case

- Do we need to keep track of how many times something is guessed? Do you need to keep track of the order in which the user makes guesses?

    - How many times something is guessed and the order of the guesses are irrelevant to this game. All that matters is whether or not something has been guessed.

- What data type can we use to keep track of guesses? Keep in mind your answer to the previous questions. Make sure the data type you chose is easily printable in alphabetical order. [1]

    - We can use an array to keep track of guesses

- Based on your previous response, how can we check if a letter has already been guessed. [2]

    - Loop through the array of guessed letters to check if a letter has already been guessed

---

[1] Your answer should not involve rearranging the old guesses when a new one is made.

[2] The answer to this should be 1-2 lines of code. Keep it simple. If you struggle to do this, investigate different solutions to the previous questions that make this one easier.

Strings and characters

- Python has the functions `chr()` and `ord()`. Describe what these functions do. If you are not already familiar with these functions, do some research into them.

  - chr() takes in an ascii value and returns the corresponding character. ord() takes in a character and returns the corresponding ascii value.

- Below is some python code. Finish the C code below so it has the same effect. [3]

  ```
  x = 'q'
  print(ord(x))
  ```

  C Code:

  ```
  char x = 'q';
  printf("\%d\n", (int)x)
  ```

- Without using `ctype.h` or **any** numeric values, write C code for `is_uppercase_letter()`. It should return false if the parameter is not uppercase, and true if it is.

  ```
  #include <stdbool.h>
  char is_uppercase_letter(char x){
      return x >= 'A' && x<='Z'
  }
  ```

- What is a string in C? Based on that definition, give an example of something that you would assume is a string that C will not allow.

  - A string is just an array of characters. If you try to instantiate something in single quotes, C will not recognize it as a string.

- What does it mean for a string to be null terminated? Are strings null terminated by default?

  - If a string is null terminated, it has a null terminator at the end indicating that the string is ending. Strings are null terminated by default.

- What happens when a program that is looking for a null terminator and does not find it.

  - The program will start to act strangely and not as intended.

- In this assignment, you are given a macro called `CLEAR_SCREEN`. What is it's data type? How and when do you use it?

  The macro is a string. You use it by printing "033[1;1H33[2J" which clears the entire screen.

Testing

List what you will do to test your code. Make sure this is comprehensive. [4] Remember that you will not be getting a reference binary [5].

I will create print statements and run my code every few lines written to make sure each line of code is working as intended. This way if I an error occurs I know that the error must lie only in the recent lines of code that are untested.

---

[3]Do not hard code any numeric values.

[4]This question is a whole lot more vague than it has been the last few assignments. Continue to answer it with the same level of detail and thought.

[5]The output of your binary is not the only thing you should be testing!

How to Use the Program

Audience: Write this section for the user of your program. You are answering the basic question, "How do I use this thing?". Don't copy the assignment exactly; explain this in your own words. This section will be longer for a more complicated program and shorter for a less complicated program. You should show how to compile and run your program. You should also describe any optional flags or inputs that your program uses, and what they do.

To show "code font" text within a paragraph, you can use \lstinline{}, which will look like this: `text`. For a code block, use \begin{lstlisting} and \end{lstlisting}, which will look like this:

```
Here is some code in lstlisting.
```

And if you want a box around the code text, then use \begin{lstlisting}[frame=single] and \end{lstlisting}

which will look like this:

```
Here is some framed code (lstlisting) text.
```

Want to make a footnote? Here's how.[6]

Do you need to cite a reference? You do that by putting the reference in the file `bibtex.bib`, and then you cite your reference like this[1][2][3].

To begin the program, type ./hangman ¡desired secret word in here¿. Afterwards, the program will show you a line for the phrase, a list for eliminated letters, and a prompt to enter a letter. Type in one lowercase character to guess a character. This repeats until you either guess every letter or you make too many mistakes. If you manage to guess every letter, you win the game. If you make too many mistakes, you lose.

Program Design

Audience: Write this section for someone who will maintain your program. In industry you maintain your own programs, and so your audience could be future you! List the main data structures and the main algorithms. You are answering the basic question, "How is this thing organized so that I can have a chance of fixing it?". This section will be longer for a more complicated program and shorter for a less complicated program.

Pseudocode

Give the reader a top down description of your code! How will you break it down? What features will your code have? How will you implement each function.

Function Descriptions

For each function in your program, you will need to explain your thought process. This means doing the following

- The inputs of every function (even if it's not a parameter)

- The outputs of every function (even if it's not the return value)

- The purpose of each function, a brief description about a sentence long.

- For more complicated functions, include pseudocode that describes how the function works

- For more complicated functions, also include a description of your decision making process; why you chose to use any data structures or control flows that you did.

---

[6]This is my footnote.

Do not simply use your code to describe this. This section should be readable to a person with little to no code knowledge. **DO NOT JUST PUT THE FUNCTION SIGNATURES HERE. MORE EXPLANATION IS REQUIRED.**

is_lowercase_letter(char c):

takes in a char and outputs a boolean. It will return true if the char is a lower case letter and false otherwise. The purpose of this function is to check to make sure that the user is inputting a lowercase letter as that is what is required of the secret.

validate_secret(const char *secret):

takes in a char and checks if it is a valid hangman secret. If it is valid, it prints nothing and returns true. If it is invalid, it will print the first invalid character to the screen. The error message will print to stdout. A valid hangman secret may only contain lowercase letters, spaces, hyphens, and aspotrophes. The purpose of this function is to make sure that the secret the user inputs is one that consists of valid characters and not random gibberish.

string_contains_character(const char *s, char c)

Takes if string s contains the character c. If c is in string s, return true. Otherwise return false. The purpose of this function is to check if a guess the user makes towards figuring out the secret was correct or not.

read_letter(void)

prompts user to to guess and reads a letter from stdin and then returns it. The purpose of this function is to grab the guesses the user makes and use it in our program.

References

[1] Wikipedia contributors. C (programming language) — Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/C_(programming_language), 2023. [Online; accessed 20-April-2023].

[2] Robert Mecklenburg. *Managing Projects with GNU Make, 3rd ed.* O'Reilly, Cambridge, Mass., 2005.

[3] Walter R. Tschinkel. Just scoring points. *The Chronicle of Higher Education*, 53(32):B13, 2007.