Naomi Kalombo

Marisleysis De La Cruz

Raveen Green

CS 105 Project Report

December 10th, 2019

Hangman/Word Catcher Game

**Overview/Summary:**

Our project was to create and implement a code that would allow two players to play the game Hangman/Word Catcher. Hangman is when Player 1 thinks of a phrase or word and writes it down making sure that the number of spaces corresponds to each letter in the phrase they’ve created. Our game consists of six rounds, in which, both Player 1 and Player 2 have three rounds to guess and write down a phrase. In the 1st round, Player 1 gives a phrase and Player 2 will have to guess the letters that go into the answer. Player 2 only has 6 chances to guess what the word/phrase is. When a letter guessed is incorrect, then Player 2 will lose one of their chances, and will have 5 remaining. The amount of chances a Player has is kept track of. When a letter guessed is correct, then the letter us added into its designated empty spot. This game will continue on until Player 2 wins or losses all of their chances. If Player 2 wins, then the word is added to their word bank, which is where all their correctly guessed words are kept. If Player 2 loses, then the word is not added to either Player 1’s or Player 2’s word bank. For the 2cnd round, Player 2 will give the phrase/word and Player 1 will guess. This continues on for 4 more rounds until all 6 rounds are completed and the person with the most words in their word bank wins. If there is a tie, than both players win.

**Background/Relation to Class:**

Our project is suitable for this class because we used lists, word dictionaries, strings, recursion, and loops. These are the topics that we have learned in class and we want to demonstrate our understanding by implementing them in a fun game. We extended our knowledge of strings by using them creatively in order to solve problems that came up while creating our game.

**Motivation: Why Hangman?**

We chose Hangman/Word Catcher because it matched with our majors and it best embodied the materials we learned in class. Naomi is a Linguistics major, Mari is a potential Computer Science major and Raveen is an English major. Our majors played a huge role in the game we decided, because ⅔ of us are passionate about words, and ⅓ of us is passionate about the process of writing code. Hangman/Word Catcher also brings back fond childhood memories; all three of us grew up playing this game with our family and friends. It was a favorite pastime. Furthermore, we wanted to make a game that any player of any age can have fun playing; our game is fun, interactive and creative especially with the added visuals.

The hangman could be seen as a simple game, but by adding the word banks element into the game of Hangman, we further challenged ourselves. Computer science is a new subject to ⅔ of us and because of that, code writing was a challenging aspect of this project, but we wanted to push ourselves further by adding a twist. We were motivated to create this game because we knew that we had a good team and that we could depend on each other to catch mistakes and give new ideas.

**Main contributions:**

We were able to accomplish a lot of our tasks and find solutions to problems that came up during writing the code for this game. The list includes:

* A player providing a word in a different language
  + We decided that the players would know to put in English words because the instructions were written in English. (Naomi)
* What if somehow both players have the same number of words in their word bank?/ there’s a tie?
  + We’ve decided to just have them both win because creating an extra round to break the tie would just let one of the players guess and it would make that one extra round unfair to the other player. (Raveen)
* How to make it more interactive
  + Created hangman visuals. We used dashes and lines to draw the actual hangman and print all of that inside a string. We had to make each drawing of the hangman appear at the same time as an incorrectly guessed word. (Mari)
* Adding the word to the word giving player when the word guessing player didn’t correctly guess the word
  + This was an obstacle that came up during the last hours of us working on this game because we were reading what our goals were to make sure that we reached when we noticed that we were missing this. Unfortunately we were unable to figure it out. (Mari, Raveen, and Naomi)

**Methods:**

We created a list of visuals to be used and then as each remaining chance decreased we accessed each element of the list using its index number. At first, we tried using Turtle, but it failed because it only makes a visual for one specific action. The image could not build on itself. In other words, Turtle can only draw one picture and that picture can not be changed. Since we wanted parts of the Hangman display drawn in time with each incorrectly guessed letter, we realized that Turtle was not a necessary tool for our project, so we had to get creative! Below is some of our thinking process and problems we solved throughout this project:

* In order to alternate the word-giving player and the word-guessing player for each round, we created a variable called **guesser** in our code and set it to be true when player 1 is the guesser and it will be false when player 2 is the guesser
* When the players are giving a word, we coded to prevent special characters and numbers from being accepted as an input
* We converted the phrase that was given into blank underscores, so the guessing player only sees the underscores and not the actual word
* When the player is guessing, we had to figure out how to print the remaining chances in time with the numbers of incorrect guesses.
* We also had to figure out how to prevent the guessing player from guessing a special characters and numbers
* We figured out how to add the correctly guessed word into the word bank of word-guessing player
* We figured out how to display the letters that were guessed, but were not in the word given by word-giving player and gave the word-guessing player six chances
* We coded the specific design of the Hangman visual, building on the design and putting it under each code to make sure that the visual pops up with incorrectly guessed letter. We did this by creating a list of hangman visuals and calling each element in the list depending on the number of chances the guesser had.

**Conclusion and future work:** Our game Hangman/Word Catcher is a two player game in which one player gives a word/phrase and the other player guesses letters for the empty spaces. If the player is correct, then the word is added to their word bank, and if the player didn’t guess correctly, no one gets the word. There are 6 rounds, and 6 chances to guess the correct answer. If we were given more time, we would have added more visuals and try to make it so that all languages are included so the game is more international. By adding more languages, the more people get to play and interact with each other. Also if we had more time, we could have figured out how to add the word to the word bank of the word giving player when the word guessing player didn’t correctly guess the word. This was one aspect of our game that we were excited about and know that there is a solution, but we didn’t have the time to figure it out. Overall, we are proud of the game that was produced through our determination, creativity and teamwork.

**References:**

* Labs/Notes from class
* Mari’s previous computer science experience
* Brett Hungar Peer Tutor

**Our Team Contribution Table:** We worked a total of 16 hours on this project and we worked the majority of those hours together.

|  |  |  |
| --- | --- | --- |
| Mari | Raveen | Naomi |
| Code that allows input for players to give the phrase and input for players to guess the phrase  Code that allows input for players to give the phrase and input for players to guess the phrase  Code that matches the letter guessed to the letter in the word and replaces the underlined space with the letter  Code that matches the letter guessed to the letter in the word and replaces the underlined space with the letter  Code that matches the letter guessed to the letter in the word and replaces the underlined space with the letter  Code that alternates between which player goes first in each round  Code that alternates between which player goes first in each round  Working on any unforeseen risks that occur and test cases & presentation (all of us) | Code that gives output when the letter inputted by the player is in the phrase being guessed/ when it’s not  Code that gives output when the letter inputted by the player is in the phrase being guessed/ when it’s not  Code that puts the same letter in multiple places if it appears more than once in the phrase  Code that puts the same letter in multiple places if it appears more than once in the phrase  Code that puts the same letter in multiple places if it appears more than once in the phrase  Code that keeps track of remaining chances and ends the game  Code that ends the game  Working on any unforeseen risks that occur and test cases & presentation (all of us) | Code that turns the letters in the words into underlined spaces and gives this new format to player 2  Code that turns the letters in the words into underlined spaces and gives this new format to player 2  Code to keep track of words that player 1 has guessed and player 2 has guessed  Code to keep track of words that player 1 has guessed and player 2 has guessed  Code to keep track of words that player 1 has guessed and player 2 has guessed  Code to raise exceptions if numbers or symbols are entered instead of string  Code to raise exceptions if numbers or symbols are entered instead of string  Working on any unforeseen risks that occur and test cases & presentation (all of us) |