Juan Rodriguez

Professor Kanemoto

CPSC-39

December 6, 2024

Report

Report Requirements (The report must explain the following):

- **What your game or app does and why it is entertaining.**
    - The game I have created drew inspiration from, and is based on the widely known guessing game known as Hangman. In the game I've made, the user may pick from one of three topics of words to guess from. Ranging from dinosaurs, to chemical elements and lastly, car brands. I think this program can be considered entertaining because it's simple, makes the user test their knowledge, and sometimes makes the user actually try on some of the words to guess. In no way can the level of entertainment provided by this game compete with popular mainstream games. However, if one were to be immensely bored, seeking but the slightest amount of entertainment to cure their boredom , this game may just do the trick.
- **Includes at least 3 algorithms as steps or a flowchart, and a snapshot of the algorithms code.**
    - Algorithms:
        - 1.) Not Terminate:
            - Steps:

1. Start

2. Print/Display (“You guessed the word, good job! Play Again?

   1 = Yes 0 = No”)

3. Read int userDecision
4. If userDecision = 1 → retry = true → clearScreen()
5. Else if userDecision = 0 → retry = false → clearScreen() →

   Print/Display(“Thank you for playing!”)

6. End

- 2.) Terminate:
  - Steps:

1. Start
2. Print/Display (“Game Over, Try Again?, 1 = Yes 0 = No”)
3. Read int userDecision
4. If userDecision = 1 → retry = true → clearScreen()
5. Else if userDecision = 0 → retry = false → clearScreen() →

   Print/Display(“Thank you for playing!”)

6. End

- 3.) Catch Error
  - Steps:

1. Print/Display (“Enter the character correlated to the topic you want

   to play,

2. Read int userPick
3. If userPick.isEmpty = false → topicChar = userPick.charAt(0)

4. If topicChar = d, e, or c → catchError = false → else →

   Print/Display (“Invalid input. Please enter 'd', 'e', or 'c'.”)

5. Else → Print/Display (“Input cannot be empty. Please try again.")

Snapshots:

```
while(catchError){
    System.out.print("Enter the character correlated to the topic you want to play: ");
    userPick = scanner.nextLine();
    if(!userPick.isEmpty()){
        topicChar = userPick.charAt(0); //The letter the user entered.
        if(topicChar == 'd' || topicChar == 'e' || topicChar == 'c'){
            catchError = false;
        }else{
            System.out.println("Invalid input. Please enter 'd', 'e', or 'c'.");
        }
    }else{
        System.out.println("Input cannot be empty. Please try again.");
    }
}//----------------------------------------------------------------------------------------
```

```
if (!terminate) {
    System.out.println("You guessed the word, good job!");
    System.out.println();
    System.out.println("Play Again?");
    System.out.println("1 = Yes     0 = No");
    int userDecision = scanner.nextInt();
    if (userDecision == 1) {
        retry = true;
        clearScreen();
    } else if (userDecision == 0) {
        retry = false;
        clearScreen();
        System.out.println("Thank you for playing!");
    }
```

```
} else if (terminate) {
    System.out.println("Game Over");
    System.out.println();
    System.out.println("Try Again?");
    System.out.println("1 = Yes     0 = No");
    int userDecision = scanner.nextInt();
    if (userDecision == 1) {
        retry = true;
        clearScreen();
    } else if (userDecision == 0) {
        retry = false;
        clearScreen();
        System.out.println("Thank you for playing!");
    }
}//_____________________________________
```

- **Explains in a paragraph or three the algorithms that you created and how they are used in the game or app. How you created them is important, and if you used ChatGPT here, you can explain how you used it - ChatGPT should not be able to write your algorithms in totality.**
    - Speaking of the three algorithms I presented as a series of steps, the algorithms I created were Not Terminate, Terminate, and Catch Error. These algorithms serve important purposes in the code of my program. For example, Not Terminate indicates the game over switch didn't activate and the user fully guessed the word correctly. Specifying so by printing out the statement "You guessed the word,

good job!". For Terminate, if the boolean terminate is true, then that indicates the user ran out of lives and the game over screen was activated, printing out the message "Game Over." Once the game ends, it will ask the user if they desire to continue playing. To continue playing, the user enters 1, or to stop, they enter 0. Lastly, Catch Error plays a vital part in making the program as error proof as possible. This is done by verifying/ensuring the user does not mistakenly provide the incorrect input that would otherwise lead to the game crashing. This would ultimately lead to a smoother playing experience. As for the usage of chatGPT for these algorithms, I resorted to using chatGPT for the development of the algorithm Catch Error due to running into an IndexOutOfBoundsException error. Initially I wasn't sure why this error was occurring. I tried implementing try and catch statements using the InputMismatchException. I was stuck trying to figure this out so I asked chatGPT why this issue was happening. Turns out it was due to the possibility of the user entering an empty string as input, therefore, indicating an IndexOutOfBoundsException. Instead of worrying about the user entering the wrong character, I had to worry about the user mistakenly not entering anything which would cause the program to crash. Asking chatGPT for help clarifying this issue greatly helped me save time, rather than debugging this for hours trying to find the cause of the issue.

- **Discusses the Big O time of these algorithms.**
    - The Big O time for two of these algorithms; Not Terminate, and Terminate would be most closely determined to be Constant Runtime (O(1)) since the size of input the algorithms take doesn't change. However, Catch Error may be an algorithm of

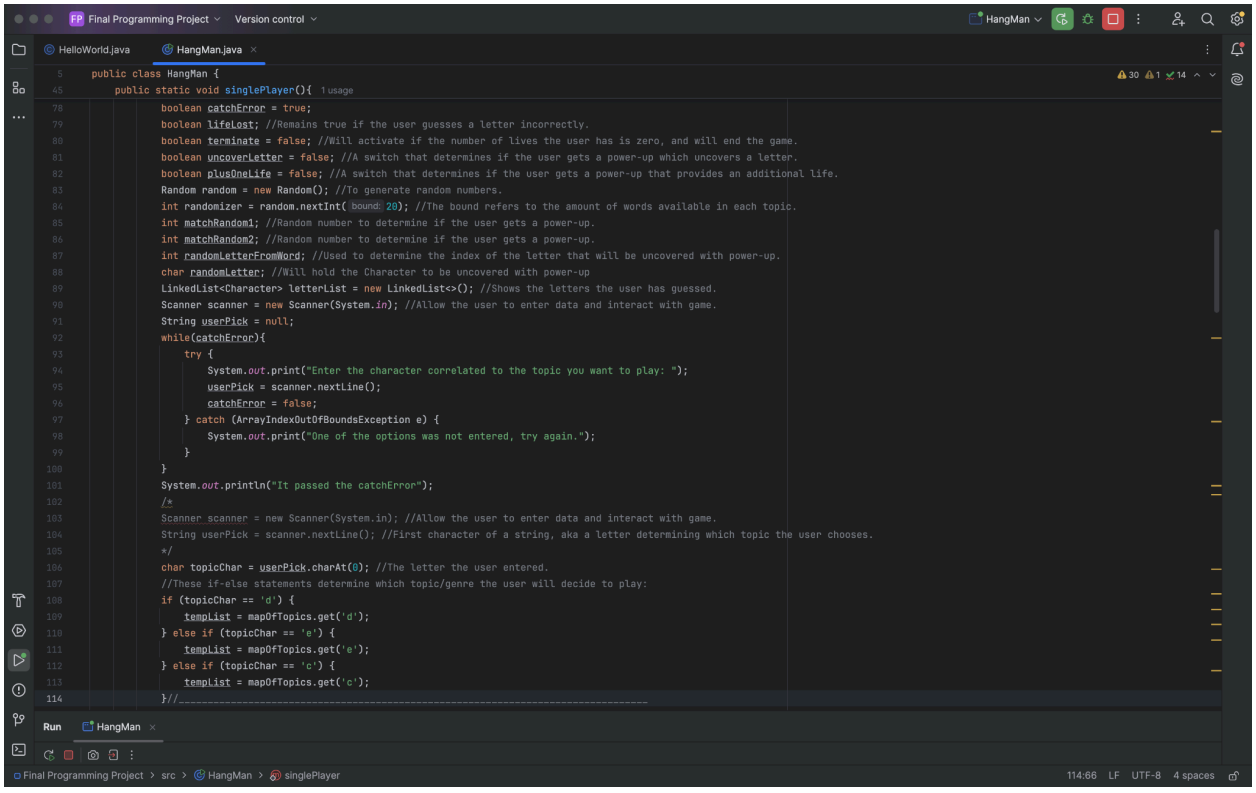
linear runtime since the iterations of the program can theoretically reach n number of iterations.

- **An explanation of the data structures that you used, why you chose them, and how they were used.**
    - The data structures I used were: Strings, Arrays, Linked Lists, Array Lists, and Hashmaps. I used Strings to store both input from the user and words from arrays. I used arrays to store all the information of the words of different topics from which the user could guess from. I used Linked Lists to easily manipulate data, for example, in my game, I used a linked list to store the letters the user guesses and print them to the terminal. They are printed in the terminal in the sequence in which the user guessed them. This way the user is able to see which letters it already entered. Lastly, I used Hashmaps to store ArrayLists containing information in regard to the words used for the game. I stored these ArrayLists along with a key that is a character that matches the topic of each Arraylist stored as a value in the Hashmap. For example, the key I used for the ArrayList value that contained the car brands was key Character ‘c’.
- **Explains a step in the design or development process where you encountered and error and how you resolved this**
    - Like I previously mentioned, when developing the algorithm, Catch Error, I was trying to find a way to prevent the game from crashing if the user ever mistyped or input the incorrect key when the program would ask the user for input. I initially thought of using try and catch statements to catch onto these errors but kept encountering the error IndexOutOfBoundException. I had no idea what was

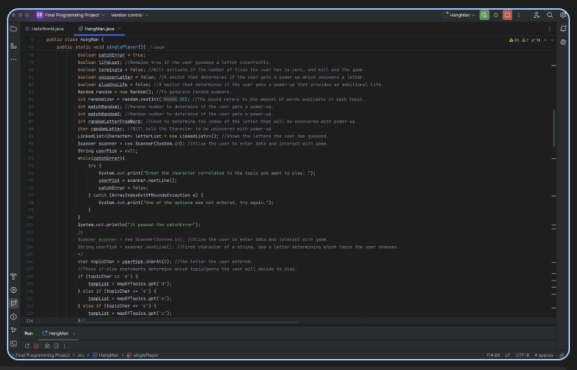
causing this issue even after debugging the parts of the code I thought were responsible for this error. Ultimately after spending a while trying to fix this, I asked chatGPT for feedback since I wasn't going anywhere. Turns out this had a simple fix. Instead of catching the error of an InputMismatch exception, I had to catch the error of an IndexOutOfBounds exception. Like previously stated, the reason being due to the possibility of the user entering an empty string as user input. I was oblivious to the cause of this issue, part of this lack of experience comes from the fact that this is the first time I'm coding an entire project with Java. Due to this, I also referred to the website W3Schools for enlightenment on specific syntax or learn how to use the tools optimal for the goal in mind.

- Citation for the portion of chatGPT help:

Code I was trying to debug(lines 92 - 100):



Trying to resolve the issue asking chatGPT for feedback:



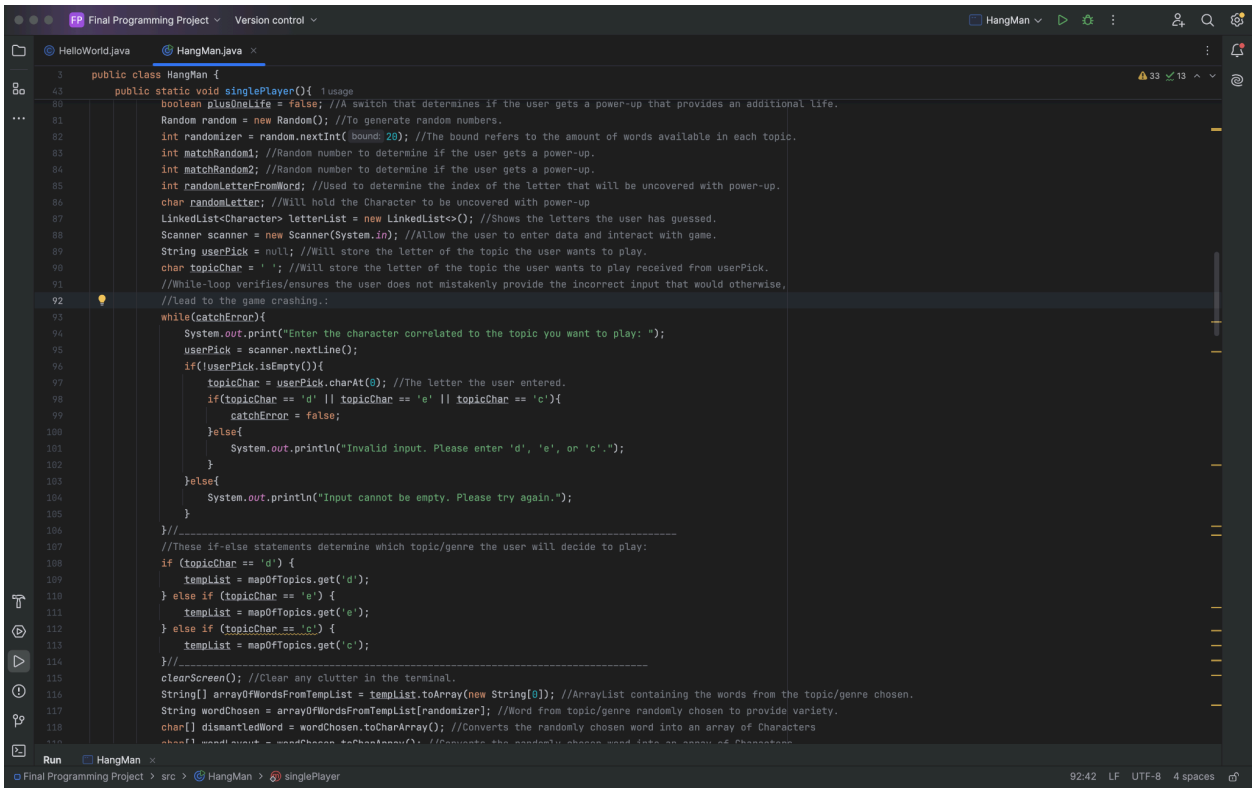
For line 93, I want to have a try and catch that'll prevent the program from crashing if the user enters anything other than 'c', 'e', or 'd' but when I try to activate the error. Instead of activating a InputMismatchException error, it activates an ArrayIndexOutOfBoundsException. Why is that?

The `ArrayIndexOutOfBoundsException` is being triggered instead of an `InputMismatchException` because of how the program is processing the input. Let's break it down:

Why `ArrayIndexOutOfBoundsException` is Triggered

1. In **line 93**, you're extracting the first character from the `userPick` string using `char topicChar = userPick.charAt(0);`.
    - If the user enters an **empty string** (e.g., just presses "Enter"), `userPick` will have a length of 0.
    - When you call `charAt(0)` on an empty string, it tries to access the first character, which doesn't exist. This causes an `ArrayIndexOutOfBoundsException`.

After the issue was resolved:



```
public class HangMan {
    public static void singlePlayer(){ 1 usage
        boolean plusOneLife = false; //A switch that determines if the user gets a power-up that provides an additional life.
        Random random = new Random(); //To generate random numbers.
        int randomizer = random.nextInt( bound: 20); //The bound refers to the amount of words available in each topic.
        int matchRandom1; //Random number to determine if the user gets a power-up.
        int matchRandom2; //Random number to determine if the user gets a power-up.
        int randomLetterFromWord; //Used to determine the index of the letter that will be uncovered with power-up.
        char randomLetter; //Will hold the Character to be uncovered with power-up
        LinkedList<Character> letterList = new LinkedList<>(); //Shows the letters the user has guessed.
        Scanner scanner = new Scanner(System.in); //Allow the user to enter data and interact with game.
        String userPick = null; //Will store the letter of the topic the user wants to play.
        char topicChar = ' '; //Will store the letter of the topic the user wants to play received from userPick.
        //While-loop verifies/ensures the user does not mistakenly provide the incorrect input that would otherwise,
        //lead to the game crashing.:
        while(catchError){
            System.out.print("Enter the character correlated to the topic you want to play: ");
            userPick = scanner.nextLine();
            if(!userPick.isEmpty()){
                topicChar = userPick.charAt(0); //The letter the user entered.
                if(topicChar == 'd' || topicChar == 'e' || topicChar == 'c'){
                    catchError = false;
                }else{
                    System.out.println("Invalid input. Please enter 'd', 'e', or 'c'.");
                }
            }else{
                System.out.println("Input cannot be empty. Please try again.");
            }
        }//_____
        //These if-else statements determine which topic/genre the user will decide to play:
        if (topicChar == 'd') {
            tempList = mapOfTopics.get('d');
        } else if (topicChar == 'e') {
            tempList = mapOfTopics.get('e');
        } else if (topicChar == 'c') {
            tempList = mapOfTopics.get('c');
        }//_____
        clearScreen(); //Clear any clutter in the terminal.
        String[] arrayOfWordsFromTempList = tempList.toArray(new String[0]); //ArrayList containing the words from the topic/genre chosen.
        String wordChosen = arrayOfWordsFromTempList[randomizer]; //Word from topic/genre randomly chosen to provide variety.
        char[] dismantledWord = wordChosen.toCharArray(); //Converts the randomly chosen word into an array of Characters
```

- **Explain what you would change or add in the next version of your app or game.**
    - Something I would change or add in the next version of my game is to add a player vs. player mode. I managed to create a single-player mode, but it would also be fun to play and compete against someone else.