CS-1180 Project 2: Simon Says

PURPOSE: Demonstrate use of conditionals, loops, String methods, and random numbers. Use of methods is **not** required for this project; however, it will significantly improve your code cleanliness.

PROBLEM: Your task is to write a program that plays "Simon Says" with the program user. Simon Says is a game where a user repeats a presented pattern. Your Simon Says program will support two modes. The user will choose which mode to play at the start of the game and beginning of each round.

- "easy" mode will play the game with colors
    - colors: yellow, green, red, and blue
- "hard" mode will play the game with numbers
    - numbers (single digit): 0 1 2 3 4 5 6 7 8 9

At the end of each round (i.e. the user has "lost"), the user should be presented with the maximum number of sequences they correctly repeated and then prompted to do another round and, if so, select mode.

REQUIREMENTS:

- You may only use tools discussed so far in this course: data types & Strings, random numbers, if statements and conditionals, switch statements, and loops (for, while, and do while). You are not required to use methods, but they will help with code cleanliness.
- Use of ArrayLists, HashMaps, etc. is **not** allowed.
- You are permitted to use primitive arrays (e.g. String[], int[], etc.).
- If the user enters an invalid mode, continue prompting them until they enter a valid mode
- After "Simon says: ___ ", program should wait 3 seconds, clear the screen, and then get input from the user. (See Provided Code Hints section)
- In a round (regardless of mode), Simon adds one thing to remember.
- When checking if the user has entered the correct sequence, you should **ignore spaces AND ignore casing** (upper / lower case lettering).
    - ex. `` `Simon Says: red green yellow` ``, user can repeat `` `red green yellow` `` OR `` `redgreenyellow` `` OR `` `RED greenYELLow` ``
- When a round is "lost" (the user did not enter the correct sequence), the user should be offered to play another round - if the user enters "no", the program should exit. If the user enters "yes", prompt the user for a mode and start a new round. Repeat question until user enters "yes" or "no"
- The score should be reset each round.

PROVIDED CODE/HINTS:

Here is some code that pauses for three seconds and clears the screen. You can now call this code from other methods (like `main`)

```
public static void pauseClear(){
    try {
        System.out.print("3...  ");
        Thread.sleep(1000);
        System.out.print("2...  ");
        Thread.sleep(1000);
        System.out.print("1...");
        Thread.sleep(1000);
    } catch (InterruptedException e) {
        // ignore
    }

    for(int i = 0; i < 100; i ++){
        // print 100 newlines
        System.out.println();
    }
}
```

Source code getting messy? Your Java extension for VSCode includes a formatter. In the Command Palette (accessed with `` `Ctrl + Shift + P` `` in Windows), select "Format Document".

EXAMPLE:

```
Let's play Simon Says!
Select difficulty (easy / hard): nice
Invalid difficulty
Select difficulty (easy / hard): easy
Easy mode - colors
Simon says: green
[wait 3 seconds, clear screen]
Player repeats: green
Score: 1
Simon says: green yellow
[wait 3 seconds, clear screen]
Player repeats: green yellow
Score: 2
Simon says: green yellow green
[wait 3 seconds, clear screen]
Player repeats: greenyellowgreen
Score: 3
Simon says: green yellow green blue
[wait 3 seconds, clear screen]
Player repeats: ahhh
Round over!  Your score was 3
Would you like to play another round? (yes / no) yes
Select difficulty (easy / hard): hard
Hard mode - numbers
Simon says: 3
[wait 3 seconds, clear screen]
Player repeats: 2
Would you like to play another round? (yes / no) no
Thanks for playing!
```

RUBRIC: Projects that don't compile will receive a 0. Use block comments for any non-compiling code.

(80 pts) **Base Functionality**

[10] User can choose difficulty level. Invalid choices are rejected until user enters valid choice

[20] Easy mode enabled

[20] Hard mode enabled

[10] Spacing and capitalization are ignored when validating the user's answer

[10] User's score is displayed during and at the end of each round

[10] User can choose if they play again. Invalid choices are rejected until user enters valid choice

(20 pts) **Style**

[10] Program is well-commented and clearly organized

[10] Program follows standard coding conventions, including variable and class names