PIERIAN DATA

(https://www.udemy.com/user/joseportilla/)

Guessing Game Challenge

Let's use `while` loops to create a guessing game.

The Challenge:

Write a program that picks a random integer from 1 to 100, and has players guess the number. The rules are:

1. If a player's guess is less than 1 or greater than 100, say "OUT OF BOUNDS"
2. On a player's first turn, if their guess is
   - within 10 of the number, return "WARM!"
   - further than 10 away from the number, return "COLD!"
3. On all subsequent turns, if a guess is
   - closer to the number than the previous guess return "WARMER!"
   - farther from the number than the previous guess, return "COLDER!"
4. When the player's guess equals the number, tell them they've guessed correctly *and* how many guesses it took!

You can try this from scratch, or follow the steps outlined below. A separate Solution notebook has been provided. Good luck!

First, pick a random integer from 1 to 100 using the random module and assign it to a variable

Note: `random.randint(a,b)` returns a random integer in range `[a, b]` , including both end points.

In [ ]:

```
```

Next, print an introduction to the game and explain the rules

In [ ]:

```
```

Create a list to store guesses

Hint: zero is a good placeholder value. It's useful because it evaluates to "False"

In [ ]:

```

```

Write a `while` loop that asks for a valid guess. Test it a few times to make sure it works.

In [ ]:

```
while True:
    
    pass
```

Write a `while` loop that compares the player's guess to our number. If the player guesses correctly, break from the loop. Otherwise, tell the player if they're warmer or colder, and continue asking for guesses.

Some hints:

- it may help to sketch out all possible combinations on paper first!
- you can use the `abs()` function to find the positive difference between two numbers
- if you append all new guesses to the list, then the previous guess is given as `guesses[-2]`

In [ ]:

```
while True:

    # we can copy the code from above to take an input

    pass
```

That's it! You've just programmed your first game!

In the next section we'll learn how to turn some of these repetitive actions into *functions* that can be called whenever we need them.

Good Job!