In the last article, we did a guessing game where the computer generates a random number using <code>random.randint()</code> and then we try to guess the number. You can read the game strategy <a href=https://dmb100days.herokuapp.org>here</a>.

Anyways, today we would be reversing roles. We would pick a number in our head and then the computer would try to guess your number. To do this, we are going to make use of functions, loops and if…else statements again. The major difference is just our logic.

When the computer generates a random number, we want to be able to provide feedback if the number is too high (H), too low (L) or if it is correct (C). To do this we are going to make use of

<code>input()</code>

As far as the number is not yet correct, the computer would keep guessing so we would need a while loop. We want that loop to stop when the feedback is equal to “c” so in order to use the while loop, we define a variable <code>feedback = “”</code> which is set to a empty string.

<first snippet>

At the end of the input(), there is

<code>.lower()</code>

What this does is it converts the string coming in from the input to lowercase so even if the user types in upper case or lower case, our feedback variable is always going to contain a lower case letter. At this point in the code the computer is generating a random number and we can provide feedback. We need to make decisions based on that feedback so we would need if…else statements. If the number guessed by the computer is to low (i.e. feedback == “l”), we need to increase the low parameter of the <code>random.randint(low, high)</code> and if it is too high (i.e. feedback ==”h”), we need to reduce the high parameter so that the next guess of the computer would be much closer to our number.

<second snippet>

There might come a point where our high would be equal to our low (for example, if our number is 50 and computer guesses 51 and we say that is too high, our new high becomes 50. The computer then guesses 49 and we say that is too low, the new low becomes 50). At that point, your number has been narrowed down. There is no need to generate a random number in that instance. We can address this with an if…else statement. If our high is not equal to our low, we can generate a random number and then use the feedback to adjust the values of our high and low but else if they are equal, the computer guess is equal to either one of them and not equal to a random number generator.

<third snippet>

Below we can see what our result would look like. <br />

<img src=”/images/computer\_guess>” <br />

Cheers 🥂