Let’s practice and…. Let’s play a game….

Exercise 1 (10pts)

Create a program that asks the user to enter their name and their age. Print out a message addressed to them that tells them the year that they will turn 100 years old.

Exercise 2 (15pts)

Ask the user for a number. Depending on whether the number is even or odd, print out an appropriate message to the user. *Hint: how does an even / odd number react differently when divided by 2?*

Exercise 3 (15pts)

Take a list, say for example this one:

```
  a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
```

and write a program that prints out all the elements of the list that are less than 5.

Exercise 4 (20pts)

Ask the user for a string and print out whether this string is a palindrome or not. (A **palindrome** is a string that reads the same forwards and backwards.)

Exercise 5 (30pts)

Make a two-player Rock-Paper-Scissors game. (*Hint: Ask for player plays (using `input`), compare them, print out a message of congratulations to the winner, and ask if the players want to start a new game*)

Remember the rules:

- Rock beats scissors
- Scissors beats paper

- Paper beats rock

Exercise 6 (15pts)

Implement a function that takes as input three variables and returns the largest of the three. Do this without using the Python `max()` function!

Exercise 7 (15pts)

Write a python program to print the square of all numbers from 0 to 10

Exercise 8 (15pts)

Write a python program to find the sum of all even numbers from 0 to 10

Exercise 9 (15pts)

Write a python program to get the following output

```
1-----99

2-----98

3-----97

.     .

.     .

.     .

98-----2

99-----1
```

Exercise 10 (20pts)

Generate a random number between 1 and 9 (including 1 and 9). Ask the user to guess the number, then tell them whether they guessed too low, too high, or exactly right.

Extras: (+10pts/extra)

- Keep the game going until the user types “exit”
- Keep track of how many guesses the user has taken, and when the game ends, print this out.

Exercise 11 (20pts)

Write a password generator in Python. Be creative with how you generate passwords - strong passwords have a mix of lowercase letters, uppercase letters, numbers, and symbols. The passwords should be random, generating a new password every time the user asks for a new password.

(Hint: use the import random and research for the command random.sample)