LAB B.1

As Instructor, I want all team exercise and improve basic Python programming skills so they can get more familiar with the environment, language, and also prepare the foundation for more complex development. All the team should participate. This is a team assignment

This story is done when

- The solutions for the basic problems are generated
- The source code is uploaded in your repository, a folder for each solution. For example LAB-B/ Problem 1
- The testing is performed using python unit test framework

https://www.slideshare.net/micropyramid/unit-testing-with-python

https://jeffknupp.com/blog/2013/12/09/improve-your-python-understanding-unit-testing/

https://docs.python.org/3/library/unittest.html

Problem 1. Odd or Even
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?* Extras: 1. If the number is a multiple of 4, print out a different message. 2. Ask the user for two numbers: one number to check (call it num) and one number to divide by (check). If check divides evenly into num, tell that to the user. If not, print a different appropriate message. 3. If the number if a prime number. **Help:** • The official Python documentation • Python for beginners explains conditionals

Problem 2. List Confusion
Take two lists, say for example these two:

```
a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]
```

and write a program that returns a list that contains only the elements that are common between the lists (without duplicates). Make sure your program works on two lists of different sizes.

Extras:

1. Randomly generate two lists to test this
2. Write this in one line of Python (don't worry if you can't figure this out at this point - we'll get to it soon)

Problem 3. List Analysis

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.)

Problem 4. File Overlap

Given two `.txt` files that have lists of numbers in them, find the numbers that are overlapping. ***One .txt file*** has a list of all prime numbers under 1000, and the ***other .txt file*** has a list of happy numbers up to 1000. The output should be stored in a third file, named as ***output.txt***.

(If you forgot, prime numbers are numbers that can't be divided by any other number. And yes, happy numbers are a real thing in mathematics - you can look it up on Wikipedia. The explanation is easier with an example, which I will describe below.)

Problem 5. Reverse Word

Write a program (using functions!) that asks the user for a long string containing multiple words. Print back to the user the same string, except with the words in backwards order. For example, say I type the string:

```
My name is Michele
```

Then I would see the string:

```
Michele is name My
```

Problem 6. Cows and Bulls

Create a program that will play the "cows and bulls" game with the user. The game works like this:

Randomly generate a 4-digit number. Ask the user to guess a 4-digit number. For every digit that the user guessed correctly *in the correct place*, they have a "cow". For every digit the user guessed correctly *in the wrong place* is a "bull." Every time the user makes a guess, tell them how many "cows" and "bulls" they have. Once the user guesses the correct number, the game is over. Keep track of the number of guesses the user makes throughout the game and tell the user at the end.

Say the number generated by the computer is 1038. An example interaction could look like this:

```
Welcome to the Cows and Bulls Game!
Enter a number:
>>> 1234
2 cows, 0 bulls
>>> 1256
1 cow, 1 bull
...
```