Important notes:

- Name each Python file for your exercise in the following way:
    - Exercise_Task_1.py for the first task.
    - Exercise_Task_2.py for the second task.
    - …
- For each task of this exercise, you only need one file.
- Use comments to describe your program and your decisions you made during the development.
- Zip your exercise including all the three files and do the upload on the eCampus.

---

1. Console Input / Output [10 Points]

Write an easy python program searching for prime numbers between a start and an end value.
Therefore, you need to read in two times a number from the command line: `<startNumber>` and `<endNumber>`
Include exception handling for reading of the two values from the command line. If an error occurred, the program asks the user again to input a number (use a loop for that process).
Check internally if the first number is lower than the second one. If not, the user needs to select two new numbers, but tell him/her the why (also show the old numbers for a better understanding).
The results of the calculation need to be printed on the command line. Use `print()` for the output of the results on the command line.
The result of the output should like this when you are using the command line parameters `0` and `21`:

```
The prime numbers between 2 and 21 are: 2, 3, 5, 7, 11, 13, 17, 19,
```

2. Command Line Parameters [14 Points]

Write an easy python program searching calculating easy operations [+, -, x, /] of two values.
Therefore, you need to read in two times a number as well as the operation to be performed from the command line: `<startNumber> <operation> <endNumber>`
Note, you need to read in the two numbers as well as the operator from at once from the command line.
Therefore, I suggest to use the following operation after reading:

```
str = input("Enter a basic math operation: e.g. 10 \[+-x/\] 2: ")
val1, op, val2 = str.split(" ")
```

Afterwards, you need to convert the strings, which are stored in `val1` and `val2` to floats `float(val1)`
Include exception handling for reading of the string from the command line as well as splitting and converting.
If an error occurred, the program asks the user again to input a number (use a loop for that process).
Examples for the command line parameters are:

- `5.0 + 4`
- `10 - 3.0`
- `10.0 x 2`
- `10 / 2`

Thus, the operations which need to be supported are: `+ - x /`
However, for use the `if(…)-else if(…)…else` for example to program the calculation selection.
The program needs to handle float values for the numbers and a string for the calculation operator.

3. Recursive calculation of the faculty [16 Points]

Write an easy python program calculating the faculty of a given number, which was read from the command line. Therefore, you need to read in once a natural number {1,2,3,4, … n} from the command line: `<number>`. If an error occurred, the program asks the user again to input a number (use a loop for that process).

The results of the recursive calculation need to be written into a file (filename should be `<Exercise_Task_3.txt>`). Additionally, use `print()` for the output of the results on the command line.

The formal description of the faculty is:

$$n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot n = \prod_{k=1}^{n} k$$

$$0! = 1$$

The recursive definition is:

$$n! = \begin{cases} 1, & n = 0 \\ n \cdot (n-1)!, & n > 1 \end{cases}$$