Lab: Error Handling

Problems for in-class lab for the **Python Advanced Course @SoftUni.**

1. So Many Exceptions

You are provided with the following code. This code raises many exceptions. Fix it, so it works correctly.

It is given a sequence of numbers, separated by a ", ". Iterate through each number by its index, and if the number is smaller or equal to 5, make a multiplication. If the number is larger than 5 and smaller or equal to 10, divide the result by the number. In the end, print the final result.

```
numbers_list = int(input()).split(", ")
result = 1

for i in range(numbers_list):
    number = numbers_list[i+1]
    if number <= 5
        result *= number
    elif 5 < number <= 10:
        result /= number

print(total)
```

Examples

Input	Output
2, 5, 10	1.0
4, 5, 6, 1, 3	10.0
1, 4, 5	20.0
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11	0.003968253968253968

Hints

First, let us start the program, to see the first exception we will hit:

```
so_many_exceptions.py ×
1    numbers_list = int(input()).split(", ")
2    result = 1
3
4   for i in range(numbers_list):
5        number = numbers_list[i+1]
6        if number <= 5
7            result *= number
8        elif 5 < number <= 10:
9            result /= number
10
11   print(total)
12
```

```
Run:    so_many_exceptions ×
    C:\Users\User\AppData\Local\Programs\Python\Python310\python.exe "Z:/Python Advanced Module/Jan-2023/Python-Ad
        File "Z:\Python Advanced Module\Jan-2023\Python-Advanced\05-Error-Handling\so_many_exceptions.py", line 6
            if number <= 5
                          ^
    SyntaxError: expected ':'

    Process finished with exit code 1
```

It is a SyntaxError on line 6 of our code. It is missing the ":" sign. Let us **add it** and again **run the program**:

```
so_many_exceptions.py ×
1    numbers_list = int(input()).split(", ")
2    result = 1
3
4   for i in range(numbers_list):
5        number = numbers_list[i+1]
6        if number <= 5:   # new
7            result *= number
8        elif 5 < number <= 10:
9            result /= number
10
11   print(total)
12
```

```
Run:    so_many_exceptions ×
    C:\Users\User\AppData\Local\Programs\Python\Python310\python.exe "Z:/Python Advanced Module/Jan-2023/Python-Advanced/05-Err
    2, 5, 10
    Traceback (most recent call last):
        File "Z:\Python Advanced Module\Jan-2023\Python-Advanced\05-Error-Handling\so_many_exceptions.py", line 1, in <module>
            numbers_list = int(input()).split(", ")
    ValueError: invalid literal for int() with base 10: '2, 5, 10'

    Process finished with exit code 1
```

When we put the input code, the program raises a ValueError on line 1. It says that the input data is not of type "int". So, to escape the error, we should remove the int() converter and again run the program:

```
so_many_exceptions.py ×
1    numbers_list = input().split(", ")   # new
2    result = 1
3
4   for i in range(numbers_list):
5        number = numbers_list[i+1]
6        if number <= 5:
7            result *= number
8        elif 5 < number <= 10:
9            result /= number
10
11   print(total)
12
```

```
Run:    so_many_exceptions ×
    C:\Users\User\AppData\Local\Programs\Python\Python310\python.exe "Z:/Python Advanced Module/Jan-2023/Python-Advanced/05-Err
    2, 5, 10
    Traceback (most recent call last):
        File "Z:\Python Advanced Module\Jan-2023\Python-Advanced\05-Error-Handling\so_many_exceptions.py", line 4, in <module>
            for i in range(numbers_list):
    TypeError: 'list' object cannot be interpreted as an integer

    Process finished with exit code 1
```
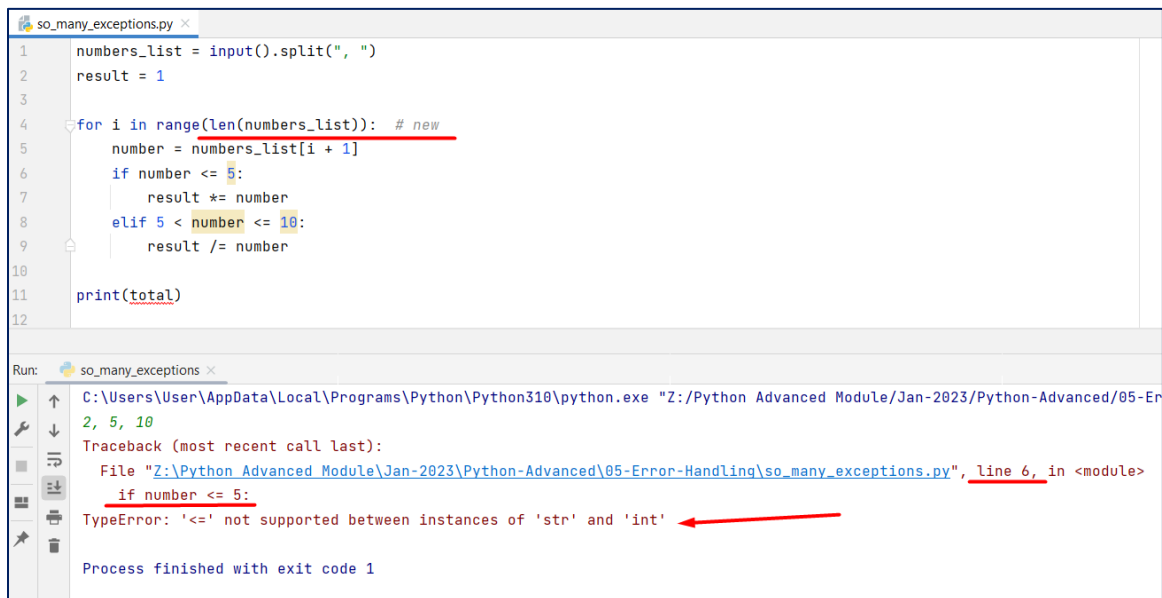
Now, the program raises a TypeError on line 4. It says that the list cannot be interpreted as an integer. As you know, when we want to use the range() function, we should give it a start value (integer), an end value (integer), and a step value (integer). However, it is given a list as an argument of that function. We should change it, as we want to use the index of the list, not the elements in it. Then, run the program again:
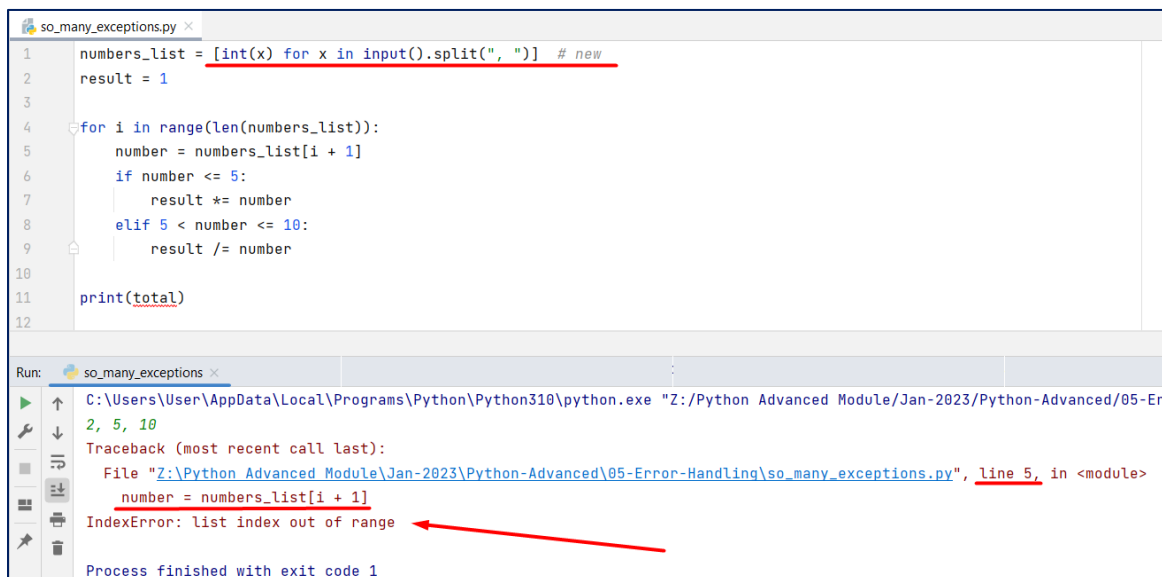
```python
numbers_list = input().split(", ")
result = 1

for i in range(len(numbers_list)):   # new
    number = numbers_list[i + 1]
    if number <= 5:
        result *= number
    elif 5 < number <= 10:
        result /= number


print(total)
```

```
C:\Users\User\AppData\Local\Programs\Python\Python310\python.exe "Z:/Python Advanced Module/Jan-2023/Python-Advanced/05-Er
2, 5, 10
Traceback (most recent call last):
  File "Z:\Python Advanced Module\Jan-2023\Python-Advanced\05-Error-Handling\so_many_exceptions.py", line 6, in <module>
    if number <= 5:
TypeError: '<=' not supported between instances of 'str' and 'int'

Process finished with exit code 1
```

The program raises a TypeError on line 6. It says that the conditional operator cannot be supported between str and int values. As we can see, the int value is 5, so the string value (number) should be changed to an integer. Let us change all elements in the list to integers at the beginning of the program:

```python
numbers_list = [int(x) for x in input().split(", ")]   # new
result = 1

for i in range(len(numbers_list)):
    number = numbers_list[i + 1]
    if number <= 5:
        result *= number
    elif 5 < number <= 10:
        result /= number


print(total)
```

```
C:\Users\User\AppData\Local\Programs\Python\Python310\python.exe "Z:/Python Advanced Module/Jan-2023/Python-Advanced/05-Er
2, 5, 10
Traceback (most recent call last):
  File "Z:\Python Advanced Module\Jan-2023\Python-Advanced\05-Error-Handling\so_many_exceptions.py", line 5, in <module>
    number = numbers_list[i + 1]
IndexError: list index out of range

Process finished with exit code 1
```

The program now raises an IndexError on line 5. It says that the index is out of the list range. We can debug that and see that when the "i" value is the last index (2 in this test), the program tries to reach the 3rd index of the list that does not exist. In this case, we want to remove the additional summation of the index:

```python
numbers_list = [int(x) for x in input().split(", ")]
result = 1

for i in range(len(numbers_list)):
    number = numbers_list[i]  # new
    if number <= 5:
        result *= number
    elif 5 < number <= 10:
        result /= number

print(total)
```

```
C:\Users\User\AppData\Local\Programs\Python\Python310\python.exe "Z:/Python Advanced Module/Jan-2023/Python-Advanced/05-Error-
2, 5, 10
Traceback (most recent call last):
  File "Z:\Python Advanced Module\Jan-2023\Python-Advanced\05-Error-Handling\so_many_exceptions.py", line 11, in <module>
    print(total)
NameError: name 'total' is not defined

Process finished with exit code 1
```

Now, the program raises a NameError on line 11 saying that the name "total" is not defined. Let us remove that error by adding the right variable name - "result":

```python
numbers_list = [int(x) for x in input().split(", ")]
result = 1

for i in range(len(numbers_list)):
    number = numbers_list[i]
    if number <= 5:
        result *= number
    elif 5 < number <= 10:
        result /= number

print(result)   # new
```

```
C:\Users\User\AppData\Local\Programs\Python\Python310\python.exe
2, 5, 10
1.0

Process finished with exit code 0
```

We can see that now we receive the right output and the program finishes with code 0.

2. Repeat Text

Write a program that receives a **text** on the first line and **times** (to repeat the text) that must be an **integer**. If the user passes a **non-integer** type for the times variable, handle the exception and print a message **"Variable times must be an integer"**.

Examples

Input	Output
Hello Bye	Variable times must be an integer
Hello 2	HelloHello

3. Value Cannot Be Negative

Create your own exception called **ValueCannotBeNegative**. Write a program that reads **five numbers** from the console (on separate lines). If a **negative** number occurs, raise the exception.

Examples

Input	Output

1
4
-5
3
10
	Traceback (most recent call last):
	  File ".\value_cannot_be_negative.py", line 8, in <module>
	    raise ValueCannotBeNegative
	__main__.ValueCannotBeNegative