Conditions and Loops

Solve the following exercises and upload your solutions to Moodle until the specified due date. Make sure to use the *exact filenames* that are specified for each individual exercise. Unless explicitly stated otherwise, you can assume correct user input and correct arguments. You are *not allowed* to use any concepts and modules that have not yet been presented in the lecture.

Exercise 1 – Submission: `a2_ex1.py` 25 Points

Write a program that calculates and outputs the subscription price of the fictional magazine *Python – The Programmer's Magazine*. To do this, read the duration of the subscription as an integer number of months.

- For a negative number or 0, print an error message (`"Invalid subscription duration"`) and terminate the program by calling `exit(0)`
- For a duration less than 6 months, the price is 6.50 per month.
- For a duration of at least 6 months but less than 12 months, the price is 5.90 per month.
- For subscription periods of one year or more, further read the customer's postal code, which should be 4 digits (you can assume that the user only enters numbers but they might not be 4 digits long).
  - For a postal code < 1000 or > 9999 (i.e., for numbers that do not have exactly 4 digits), print an error message (`"Invalid postal code"`).
  - For valid postal codes, the monthly price is $4.xx$, where xx is the middle two digits (hundreds and tens places) of the postal code.

In case of valid inputs (i.e., no errors as described above), print both the monthly amount and the total subscription amount (floats with 2 decimal places; see the example input and output below for how it must look like). Do not perform unnecessary checks, i.e., do not check for values that have already been excluded by previous conditions.

Example input and output:[1]

```
Please enter the duration of your subscription (in months): -1
Invalid subscription duration
```

Example input and output:

```
Please enter the duration of your subscription (in months): 0
Invalid subscription duration
```

Example input and output:

```
Please enter the duration of your subscription (in months): 4
The price per month is 6.50
The full price for 4 months is 26.00
```

[1] Green colored text indicates user input from the console.

Example input and output:

```
Please enter the duration of your subscription (in months): 10
The price per month is 5.90
The full price for 10 months is 59.00
```

Example input and output:

```
Please enter the duration of your subscription (in months): 14
Please enter your postal code: 123
Invalid postal code
```

Example input and output:

```
Please enter the duration of your subscription (in months): 14
Please enter your postal code: 12345
Invalid postal code
```

Example input and output:

```
Please enter the duration of your subscription (in months): 14
Please enter your postal code: 4020
The price per month is 4.02
The full price for 14 months is 56.28
```

Exercise 2 – Submission: `a2_ex2.py` 25 Points

Write a program that repeatedly reads positive integer numbers from the console using a while loop (see the example input and output below for how it must look like). You do not need to store the all values, only the current number and the previous number. The loop should exit in two situations:

- When the user inputs an `"x"` instead of a positive integer number. In this case, print `"All numbers had the same digit in the ones place"`.
- When the current number and the previous number have different digits in the ones place. In this case, print `"X and Y differ in the ones place"`, where `X` and `Y` are these two values.

Ensure that your program works for an empty sequence of numbers (i.e., the user immediately entered `"x"`), in which case you must print `"Empty sequence"`.

Example input and output:

```
Enter number or 'x': x
Empty sequence
```

Example input and output:

```
Enter number or 'x': 12
Enter number or 'x': x
All numbers had the same digit in the ones place
```

Example input and output:

```
Enter number or 'x': 12
Enter number or 'x': 622
Enter number or 'x': 502
Enter number or 'x': x
All numbers had the same digit in the ones place
```

Example input and output:

```
Enter number or 'x': 12
Enter number or 'x': 622
Enter number or 'x': 47
622 and 47 differ in the ones place
```

Exercise 3 – Submission: `a2_ex3.py` 25 Points

Using the built-in `range(start, stop, step)`, write a program that reads these three integer numbers and iterates through the value range. Your program must do the following (see the example input and output below for how it must look like):

- Count the number of even values.
- Create the sum of all odd values.
- For the first 5 values, print them to the console. If the user-specified range contains fewer than 5 values, just print these fewer values.
- After the iteration, print both the even-value counter as well as the sum of odd values. If the range does not contain any values (an empty range) or if there are no even/odd values, the respective counter and sum must be set to 0.

Example input and output:

```
Start: 3
Stop: 2
Step: 1
Even number count = 0
Sum of odd numbers = 0
```

Example input and output:

```
Start: 0
Stop: 10
Step: 2
Number in iteration 0 = 0
Number in iteration 1 = 2
Number in iteration 2 = 4
Number in iteration 3 = 6
Number in iteration 4 = 8
Even number count = 5
Sum of odd numbers = 0
```

Example input and output:

```
Start: 1
Stop: 10
Step: 4
Number in iteration 0 = 1
Number in iteration 1 = 5
Number in iteration 2 = 9
Even number count = 0
Sum of odd numbers = 15
```

Example input and output:

```
Start: 10
Stop: 50
Step: 3
Number in iteration 0 = 10
Number in iteration 1 = 13
Number in iteration 2 = 16
Number in iteration 3 = 19
Number in iteration 4 = 22
Even number count = 7
Sum of odd numbers = 217
```

Exercise 4 – Submission: a2_ex4.py 25 Points

Write a program that prints a big 0 (zero) over a user-specified number of rows/lines (integer). If this number is less than 3, then print "Invalid number of lines". Otherwise print a 0 over the entered number of lines consisting of the characters | and - (see the example input and output below for how it must look like). Examples for such "zeros" over 4 and 5 lines:

```
----
|  |
|  |
----

-----
|   |
|   |
|   |
-----
```

If we look at the example with 4 lines, we see the following:

- The first line consists of 4 hyphens -.
- Two lines consist of a | character, 2 spaces and again a | character.
- The last line consists of 4 hyphens -.

If we look at the example with 5 lines, we see the following:

- The first line consists of 5 hyphens -.
- Three lines consist of a | character, 3 spaces, and again a | character.
- The last line consists of 5 hyphens -.

Before you start programming, try to find a general pattern depending on the given number of lines: How many hyphens - must be printed in the first and last line? How many "in-between" lines are necessary? How many spaces are to be printed in each of these lines between the two | characters?

Example input and output:

```
Enter number of lines: 2
Invalid number of lines
```

Example input and output:

```
Enter number of lines: 3
---
| |
---
```

Example input and output:

```
Enter number of lines: 5
-----
|   |
|   |
|   |
-----
```

Example input and output:

```
Enter number of lines: 7
-------
|     |
|     |
|     |
|     |
|     |
-------
```

Important Information!

Please try to *exactly match the output* given in the examples (naturally, the input can be different). We are running automated tests to aid in the correction and grading process, and deviations from the specified output lead to a significant organizational overhead, which we cannot handle in the majority of the cases due to the high number of submissions.

For example, if the exercise has an output of
`Enter number of lines: XYZ`
(where `XYZ` is some user input), do not write
`Enter the number of rows: XYZ`
(additional `the` and `rows` instead of `lines`) or
`Enter number of lines:XYZ`
(missing space after the colon).

Feel free to copy the output text from the assignment sheet, and then change it according to the exercise task.