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.

Exercise 1 – Submission: `ex1.py` 20 Points

Write a program that calculates the ticket price of some public transport based on the age and, conditionally, the salary of the users. Your program should behave as follows:

- Read the age as integer value (you can assume correct user input)
- If the age is negative, print the error message `Invalid age`
- If the age is 7 or below, print `Child ticket: 10$`
- If the age is between 8 and 17 (inclusive), print `Teenager ticket: 15$`
- If the age is 18 or more, the ticket price depends on the salary:
  - Read the salary as integer value (you can assume correct user input)
  - If the salary is negative, print the error message `Invalid salary`
  - If the salary is 1000 or below, print `Reduced adult ticket 1: 20$`
  - If the salary between 1001 and 2000 (inclusive), print `Reduced adult ticket 2: 25$`
  - In all other cases, print `Adult ticket: 30$`

Example program execution:

```
Enter age: 31
Enter salary: 1700
Reduced adult ticket 2: 25$
```

Exercise 2 – Submission: `ex2.py` 10 Points

Write a program that counts uppercase characters in a user-specified string and prints the result as shown in the following example:

```
Enter text: Hello, thiS is a TEXT
The input text contains 6 uppercase characters.
```

Make sure that the printed output is `1 uppercase character` (without the plural *s*) in case there is exactly 1 uppercase character.

Exercise 3 – Submission: `ex3.py` 20 Points

Write a program where you can guess some integer number that was entered by someone (game master) beforehand (you can assume correct user input). The user (player) can now guess the number by entering values in the console (for the sake of the guessing game, you can assume that the player did not see the previous game master input). Run the program until the user either guessed the correct integer or entered the string `"exit"` (you can assume correct user input, i.e.,

either integer numbers or the string `"exit"`). If the input is smaller than the integer, print `Your number is smaller`. If the input is bigger, print `Your number is bigger`. If the numbers match, print `Congratulations!`. If the user entered the string `"exit"`, print `You lost!`.

Example program execution:

```
Enter number to guess: 7
Enter number: 3
Your number is smaller
Enter number: 6
Your number if smaller
Enter number: 8
Your number is bigger
Enter number: exit
You lost!
```

Exercise 4 – Submission: `ex4.py` 20 Points

Using the built-in `range(start, stop, step)`, write a program that reads these three values (you can assume correct input) and iterates through the value range. Your program must do the following:

- Create two variables: `odd_counter` and `even_sum` (both initialized with 0).
- If the value is odd, increment `odd_counter` by 1.
- If the value is even, add it to `even_sum`.
- If the value is the second value in the range iteration (if there even is such a second value), print `2nd value in range = X`, where `X` is this second value.
- If the value is the last value in the range iteration (if there even is such a last value), print `Last value in range = X`, where `X` is this last value.
- After the iteration, print both the counter `odd_counter` as well as the sum `even_sum`.

Example input:

```
Start: 2
Stop: 20
Step: 3
```

Example output:

```
2nd value in range = 5
Last value in range = 17
odd_counter = 3
even_sum = 24
```

Hints:

- You can get the number of range elements via `len(range_object)`.

Exercise 5 – Submission: `ex5.py` 30 Points

Write a program that calculates statistics based on integer input numbers. Your program must do the following:

- Create a console user interface as shown in the example program execution below. The available actions are `"a"` to add numbers, `"v"` to view the calculated statistics and `"x"` to exit the program. All other input is invalid and leads to an error message (see example).
- The number add action `"a"` must do the following:
  - Until the user enters `"x"`, integer numbers are read from the console (you can assume correct user input, i.e., the user enters either the string `"x"` or an integer number).
  - For every added number, the following statistics must be calculated: the sum, the average, the minimum and the maximum. You are *not allowed* to use any built-in functions like `min` or `max`.
  - The count of added numbers must also be tracked.
- The view action `"v"` should display the above statistics unless no numbers have been added yet (see example).

Example program execution:

```
Welcome to Data Statistics!

Available actions:
a - Add numbers
v - View statistics
x - Exit the program
Enter action: e
Invalid action 'e'. Try again!

Available actions:
a - Add numbers
v - View statistics
x - Exit the program
Enter action: a
Enter number or 'x' when you are done: x

Available actions:
a - Add numbers
v - View statistics
x - Exit the program
Enter action: v
No numbers have been added yet!

Available actions:
a - Add numbers
v - View statistics
x - Exit the program
Enter action: a
Enter number or 'x' when you are done: 1
Enter number or 'x' when you are done: 2
Enter number or 'x' when you are done: 3
```

```
Enter number or 'x' when you are done: 4
Enter number or 'x' when you are done: 5
Enter number or 'x' when you are done: x

Available actions:
a - Add numbers
v - View statistics
x - Exit the program
Enter action: v
Count: 5
Sum: 15
Avg: 3.0
Min: 1
Max: 5

Available actions:
a - Add numbers
v - View statistics
x - Exit the program
Enter action: a
Enter number or 'x' when you are done: -1
Enter number or 'x' when you are done: x

Available actions:
a - Add numbers
v - View statistics
x - Exit the program
Enter action: v
Count: 6
Sum: 14
Avg: 2.3333333333333335
Min: -1
Max: 5

Available actions:
a - Add numbers
v - View statistics
x - Exit the program
Enter action: x
Bye!
```