**Name**: Steve Hommy

**Pair: -**

**Amount of completed tasks: 10**

**Which tasks were left undone or incomplete: 0**

Self-assessment:

This exercise was easy for me because I have been coding with Python for many years and these tasks were beginner level. I didn’t really learn anything, but it was fun to recap these kinds of tasks. So, in summary I understood everything.

## Test report

| **Task** | **Input / action** | **Desired output** | **Actual output (use red color if desired output != actual output)** |
|---|---|---|---|
| **1** | Run code | Hello | Hello |
| | | | |
| **2** | E.g. numbers: 2, -5, 4, 7, 9, 11, 0, 445, -100, 4<br>E.g. strings: abc, 34re5, word, qwerty, cat-doc, def, 4, #-!?bc, alkf, oooooo | Integers: 2, -5, 4, 7, 9, 11, 0, 445, -100, 4<br>Strings: : abc, 34re5, word, qwerty, cat-doc, def, 4, #-!?bc, alkf, oooooo | Enter number of elements in list: 10<br>Enter number: 2<br>Enter number: -5<br>Enter number: 4<br>Enter number: 7<br>Enter number: 9<br>Enter number: 11<br>Enter number: 0<br>Enter number: 445<br>Enter number: -100<br>Enter number: 4<br>Type anything: :<br>Type anything: abc<br>Type anything: 34re5<br>Type anything: word<br>Type anything: qwerty<br>Type anything: cat-doc<br>Type anything: def<br>Type anything: 4<br>Type anything: #-!?bc<br>Type anything: alkf<br>[2, -5, 4, 7, 9, 11, 0, 445, -100, 4]<br>[':', 'abc', '34re5', 'word', 'qwerty', 'cat-doc', 'def', '4', '#-!?bc', 'alkf']<br>[68, 97, 38, 61, 63, 49, 61, 38, 60, 28] |
| | | | |
| **3** | E.g. numbers: 2, -5, 4, 7, 9, 11, 0, 445, -100, 4 | Arranged list: -100, -5, 0, 2, 4, 4, 7, 9, 11, 445<br>(Don’t worry about the formatting, it may be without commas as well.) | Enter number of elements in list: 10<br>Enter number: 2<br>Enter number: -5<br>Enter number: 4 |

| | | | |
|---|---|---|---|
| | | | Enter number: 7<br>Enter number: 9<br>Enter number: 11<br>Enter number: 0<br>Enter number: 445<br>Enter number: -100<br>Enter number: 4<br>Type anything: abc<br>Type anything: 34re5<br>Type anything: word<br>Type anything: qwerty<br>Type anything: cat-doc<br>Type anything: def<br>Type anything: 4<br>Type anything: #-!?bc<br>Type anything: alkf<br>Type anything: ooooo<br>[-100, -5, 0, 2, 4, 4, 7, 9, 11, 445]<br>['#-!?bc', '34re5', '4', 'abc', 'alkf', 'cat-doc', 'def', 'ooooo', 'qwerty', 'word'] |
| | | | |
| **4** | User inputs integers 5, -34 and 0. | Please give an integer: 5<br>Please give an integer: -34<br>Please give an integer: 0<br><br>Number of negative integers is: 1 | Please give an integer: 5<br>Please give an integer: -34<br>Please give an integer: 0<br>Number of negative integers: 1 |
| | | | |
| **5** | User inputs integers 16, -34, 17, 0. | Please give an integer: 16<br>Please give an integer: -34<br>Please give an integer: 17<br>Please give an integer: 0<br><br>Number of even integers is: 2 | Please give an integer: 16<br>Please give an integer: -34<br>Please give an integer: 17<br>Please give an integer: 0<br>Number of even integers: 2 |
| | | | |
| **6** | User inputs integers -3, 7, 30, 9, 0. | Please give an integer: -3<br>Please give an integer: 7<br>Please give an integer: 30<br>Please give an integer: 9<br>Please give an integer: 0<br><br>Sum of positive integers divisible by three is: 39 | Please give an integer: -3<br>Please give an integer: 7<br>Please give an integer: 30<br>Please give an integer: 9<br>Please give an integer: 0<br>Sum of positive integers divisible by three is: 39 |
| | | | |
| **7** | User inputs 13 | Give maximum value: 13<br>Procession is: 3, 6, 9, 12<br>Number of terms is: 4<br>Sumf of terms is: 30<br>Sum of squared terms is: 270 | Give maximum value: 13<br>Procession is: [3, 6, 9, 12]<br>Number of terms is: 4<br>Sum of term is: 30<br>Sum of squared terms is: 270 |
| **7** | User inputs 0 | Give maximum value: 0<br>Procession is:<br>Number of terms is: 0<br>Sumf of terms is: 0<br>Sum of squared terms is: 0 | Give maximum value: 0<br>Procession is: []<br>Number of terms is: 0<br>Sum of term is: 0<br>Sum of squared terms is: 0 |

|  |  |  |  |
|---|---|---|---|
| **7** | User inputs -15 | Give maximum value: -15<br>Procession is:<br>Number of terms is: 0<br>Sumf of terms is: 0<br>Sum of squared terms is: 0 | Give maximum value: -15<br>Procession is: []<br>Number of terms is: 0<br>Sum of term is: 0<br>Sum of squared terms is: 0 |
|  |  |  |  |
| **8** | User inputs R<br>User inputs P<br>User inputs S<br>User inputs S<br>User inputs P | Give your choice (R, P, S): R<br>Computer’s choice is Paper.<br>Paper covers Rock.<br>Computer 1 You 0<br><br>Give your choice (R, P, S): P<br>Computer’s choice is Paper.<br>It’s a tie!<br>Computer 1 You 0<br><br>Give your choice (R, P, S): S<br>Computer’s choice is Paper.<br>Scissors cuts Paper.<br>Computer 1 You 1<br><br>Give your choice (R, P, S): S<br>Computer’s choice is Rock.<br>Rock crushes Scissors<br>Computer 2 You 1<br><br>Give your choice (R, P, S): P<br>Computer’s choice is Scissors.<br>Scissors cuts Paper<br>Computer 3 You 1<br><br>You lost! | Give your choice (R, P, S): R<br>Computer's choice is S<br>Rock crushes Scissors<br>Computer 0 You 1<br>Give your choice (R, P, S): P<br>Computer's choice is S<br>Scissors cuts Paper<br>Computer 1 You 1<br>Give your choice (R, P, S): S<br>Computer's choice is P<br>Scissors cuts Paper<br>Computer 1 You 2<br>Give your choice (R, P, S): S<br>Computer's choice is R<br>Rock crushes Scissors<br>Computer 2 You 2<br>Give your choice (R, P, S): S<br>Computer's choice is P<br>Scissors cuts Paper<br>Computer 2 You 3<br>You win! |
|  |  |  |  |
| **9** | Run program multiple times | Random number is: 5<br><br>Random number is: 1<br><br>Random number is: 6 | Random number is: 4<br><br>Random number is: 2<br><br>Random number is: 5 |
|  |  |  |  |

**10**

Explain the following terms (use your own words, do not copy paste answers from Internet). You can answer in Finnish or English.

a. **Procedural programming**
   Procedural programming is written as step-by-step instructions and its verbose which means that its very explicit because we need to write down the step-by-step instructions in a very detailed way. This makes procedural programming code very hard to read, understand, reuse and lengthy.

b. **Functional programming**
   In functional programming we describe our programming logic using functions. Functional programming is very declarative that means just by reading the code we will know the intention right away. It avoids concepts of shared state, mutable data observed in OOP.

c. **Object oriented programming**
In object oriented programming everything is based on objects so now instead of writing step-by-step instructions in one place we will delegate or separate the tasks into smaller chunks and put them into entities called object. In other words, we are hiding the programming logic within object and this concept is called encapsulation. So, by splitting our logic into smaller modules it helps us to reuse and maintain our code better in the long run. If something breaks, we will know which object is causing the problem and we will just need to fix the logic inside that object rather than going through a few thousand lines of code and find out which line is causing the problem

d. **Class (in programming)**
Class describes all the attributes of objects as well as the methods that implement the behavior of member objects. So, the class is like an outline, constructor, or a blueprint for creating a new object.

e. **Object (in programming)**
Object is anything that you wish to manipulate or change while working through the code. It's basically an encapsulation of data variables and methods acting on that data into a single entity.

f. **Instance (in programming)**
Instances are the items that can be defined by our class that we just created. So, it's like subbranch of the class. An individual object of a certain class.

g. **Encapsulation (in programming)**
Encapsulation is worn to mask the values or status of a standardized data thing within a folder, avoiding straight entry to them by unauthorized gathering. Encapsulation is a way to protect sensitive data and maintain compliance with industry-specific data security and privacy requirements.