Complex programs

When you learn programming, you also develop a better eye for reading code (yours and others). In this part we understood the use of lists and practiced separating the UI from the program logic.

The following is from On the role of scientific thought by Edsger W. Dijkstra .

*Let me try to explain to you, what to my taste is characteristic for all intelligent thinking. It is, that one is willing to study in depth an aspect of one's subject matter in isolation for the sake of its own consistency, all the time knowing that one is occupying oneself only with one of the aspects. We know that a program must be correct and we can study it from that viewpoint only; we also know that it should be efficient and we can study its efficiency on another day, so to speak. In another mood we may ask ourselves whether, and if so: why, the program is desirable. But nothing is gained - on the contrary! - by tackling these various aspects simultaneously. It is what I sometimes have called "**the separation of concerns**", which, even if not perfectly possible, is yet the only available technique for effective ordering of one's thoughts, that I know of. This is what I mean by "focusing one's attention upon some aspect": it does not mean ignoring the other aspects, it is just doing justice to the fact that from this aspect's point of view, the other is irrelevant. It is being one- and multiple-track minded simultaneously.*

The core of Dikstra's message is, that the problem areas of a program must be separated from each other — this is exactly what we have been doing with object-oriented programming and by separating the UI from the program logic. Each problem area has been separated into its own part. This can also be viewed through the lens of program responsibilities. In his blog Robert "Uncle Bob" C. Martin describes the term "**single responsibility principle**":

When you write a software module, you want to make sure that when changes are requested, those changes can only originate from a single person, or rather, a single tightly coupled group of people representing a single narrowly defined business function. You want to isolate your modules from the complexities of the organization as a whole, and design your systems such that each module is responsible (responds to) the needs of just that one business function.

[..in other words..] Gather together the things that change for the same reasons. Separate those things that change for different reasons.

Proper program structure and following good naming principles leads to clean code. When you code a bit more, you'll learn that each program part can be given one clear area of responsibility.

Finish by answering to the quiz below.

Quiz:

Part 6 learning objectives

Points: 1/1

Below are some learning outcomes of the sixth part. Consider how well each of them applies to you. The scaling is 1 = entirely disagree, 7 = entirely agree.

I know how to use lists as instance variables.	1 ○	2 ○	3 ○	4 ○	5 ○	6 ○
	7 ◉					
I can write a program where an object is retrieved from an instance variable list.	1 ○	2 ○	3 ○	4 ○	5 ○	6 ○
	7 ◉					

I know it's a good idea to separate the user interface and the program logic.	1 ○	2 ○	3 ○	4 ○	5 ○	6 ○
	7 ◉					
I can name at least one catastrophe caused by a programming error.	1 ○	2 ○	3 ○	4 ○	5 ○	6 ○
	7 ◉					
I know what a unit test is and what they are used for.	1 ○	2 ○	3 ○	4 ○	5 ○	6 ○
	7 ◉					
I can write unit tests.	1 ○	2 ○	3 ○	4 ○	5 ○	6 ○
	7 ◉					
I'm aware what the concept of test-driven development means.	1 ○	2 ○	3 ○	4 ○	5 ○	6 ○
	7 ◉					

Answered The number of tries is not limited

You have reached the end of this section!

Remember to check your points from the ball on the bottom-right corner of the material!

In this part:

1. Objects on a list and a list as part of an object

2. Separating the user interface from program logic

3. Introduction to testing

4. Complex programs

Source code of the material

This course is created by the Agile Education Research -research group of the University of Helsinki.

Credits and about the material.



HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

MASSIIVISET AVOIMET VERKKOKURSSIT
MASSIVE OPEN ONLINE COURSES · MOOC.FI