Leerdoelen per week, kwartaal 3 – EMS31

In dit document worden alle leerdoelen die aan bod zijn gekomen in de lessen van kwartaal 3 overzichtelijk weergegeven. Mocht je ergens tegenaan lopen tijdens het maken van de eindopdrachten, dan kun je dit document gebruiken om de nodige informatie te vinden.

Inhoudsopgave

Week 3.1 – Modules in C

Je hebt deze les geleerd hoe je:

- de *interface* van een module in een `.h` bestand kunt *declareren*;
- de *implementatie* van een module in een `.c` bestand kunt *definiëren*;
- deze module kunt gebruiken en testen op een pc met behulp van professionele tools zoals Windows Subsystem for Linux (WSL), Visual Studio Code, make en CMake;
- snel en efficiënt kunt werken met WSL door een aantal eenvoudige Linux-commando's te gebruiken zoals `cd`, `wget`, `unzip`, `rm`, `mkdir` en `cp`.

Week 3.2 – Unittesten en Git submodules

Je hebt deze les geleerd hoe je:

- C-code systematisch kan testen met behulp van het eenvoudige test-framework Google-Test;

- een ontwikkelmethode voor software genaamd Test Driven Development (TDD) kan toepassen;
- code die je in meerdere applicaties wilt gebruiken in een Git submodule kan plaatsen;
- zo'n submodule kunt gebruiken onder Windows Subsystem for Linux;
- zo'n submodule kunt gebruiken in Code Composer Studio.

Week 3.3 – Dynamisch geheugenallocatie

Je hebt deze les geleerd hoe je:

- dynamisch geheugen kunt alloceren en weer vrij kunt geven;
- een FIFO-buffer kunt implementeren die dynamisch groeit en krimpt met behulp van een *singly linked list*;
- *stubs* kunt gebruiken om te testen of jouw code goed werkt als het geheugen vol is en om te testen of jouw code het gealloceerde geheugen netjes vrijgeeft.

Week 3.4 – Trunk-based development met Feature flags

Je hebt deze les geleerd hoe je:

- een FIFO-buffer als een *user defined type* kunt definiëren waardoor je meerdere buffers in een programma kunt gebruiken.
- softwareontwikkeling kan doen volgens de zogenoemde Trunk-based development met Feature flags methode;
- gelijktijdig meerdere nieuwe features voor een embedded applicatie kunt ontwikkelen.

Week 3.5 – Memory tests en test coverage

Je hebt deze les geleerd hoe je:

- fouten bij het gebruik van dynamische geheugenallocatie zoals geheugenlekken kunt opsporen met speciale tools zoals valgrind;
- weet dat je genoeg testcode hebt geschreven door gebruik te maken van test coverage metingen.

Week 3.6 – Coding standards en static code analysis

Je hebt deze les geleerd hoe je:

- gebruik kunt maken van verschillende *C coding standards*;
- gebruik kunt maken van verschillende *static code analysis* tools, waaronder Cppcheck, om de kwaliteit van je C-code te verbeteren.