Ems Deel opdracht 1

# Inleiding

Voor de cursus EMS30 was het doel om een CLI (Command Line Interface) te ontwerpen, implementeren en testen zowel de PC als CC3220S Launchpad. In dit verslag wordt er ingegaan op ontwerpbeslissingen, verschillende testen en de verwerking van deze opdracht. Ook zijn er citaten en referenties toegevoegd die hebben zorggedragen bij het opleveren van de CLI.

# De Analyse

Tijdens de analyse van de CLI is er gebruik gemaakt van het onderstaand figuur (Figure 1) en de eisen (of te wel rangevallen) Figure 2. Ook is er rekening gehouden met de beoordeling op basis van het nakijkmodel weergegeven in Figure 3. Voor een totaaloverzicht van de opdracht kan er gebruik worden gemaakt van de URL (zie referentie) [1].



1. Figure 1. Een mogelijke implementatie van de CLI [1]

De eisen behorend bij de CLI zijn opgesplitst in de commando’s en hun bijbehorende functies (Figure 2)



Figure 2. Enkele eisen voor de op leveren CLI

Naast de bovenstaande eisen moet het programma van de CLI:

* ‘case senstive’ zijn. vb. Een, eEn en een zijn allemaal verschillend.
* De CLI moet na het intoetsen van de ‘ENTER’-toets de ingetypte commando uitvoeren en bij als deze niet bestaat een foutmelding teruggeven.
* De CLI moet gebruikt kunnen worden via de UART-interface van de CC3220S Launchpad.
* Elk printbaar karakter moet na ontvangst terrugestuurd worden over de UART-verbinding.
* Bij het indrukken van ‘BACKSPACE’-toets moet het laatst ontvangen karakter worden genegeerd.

Mocht er tijd over zijn, kon er gewerkt worden aan de extra features van de CLI. Deze features bestaan uit de command completion en command history.

**Extra features (eisen):**

* Als de ‘TAB’-toets volgt na het intypen van de eerste letter(s) van een geldig commando moet het commando automatisch worden aangevuld. (Zie command completion van document) [1]
* Door middel van de ↑ (up) en ↓ (down)-toets moet het mogelijk zijn door de geschiedenis van de commando’s heen te lopen. (Zie command history van document) [1].

Als laatst is het nakijkmode ook meegenomen om een idee te schetsen hoe ervaren de student/programmeur is. Deze is slechts meegenomen in het verslag om bij te kunnen houden in hoeverre de student dit project kan halen.

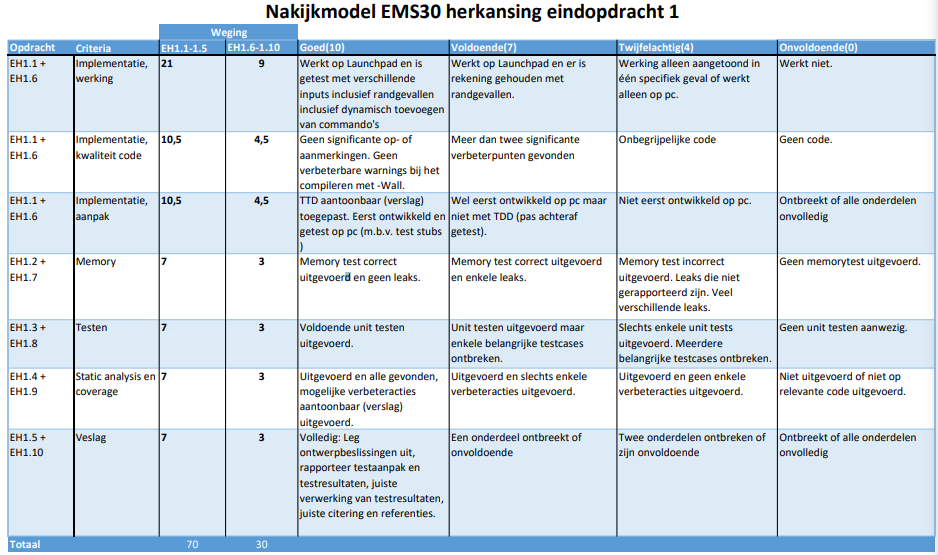


Figure 3. Nakijkmodel EMS30 Herkansing Eindopdracht 1

In het volgend hoofdstuk wordt er ingegaan op het ontwerp van de CLI. De ontwerpbesluiten worden vervolgens van code snippets onderbouwd.

# Het Ontwerp

Voor het ontwerp van de CLI worden er afhankelijk van de uit te voeren taken in de deelopdrachten verschillende besluiten genomen. Deze besluiten worden indien mogelijk onderbouwd met stukjes code of wordt er verwezen naar het mee ingeleverde code bestand.

Tijdens het ontwerpen was het een idee om per deel, uitleg te geven over de werking en het verloop van ieder deel. Maar echter uit tijdnoot was het niet haalbaar dit goed te verwerking in het verslag. Hierom is er besloten om via korte code of te wel code functie te refereren naar de desbetreffende ontwerpkeuzen.

**Deel 1**

Dit deel van het ontwerp bestaat uit de samenvoeging van opdracht EH1.1 (Figure 4) en EH1.6 (Figure 5).

Tijdens dit deel wordt er onder andere gewerkt aan het ontwerp van de CLI op de pc, echter is deze niet succesvol afgelopen omdat er enkele problemen ontstonden bij het overzetten van de pc-versie naar de Launchpad versie (via de terminal van Apple). Het schrijven van testen waaronder de TDD-test en het oplossen van warnings.

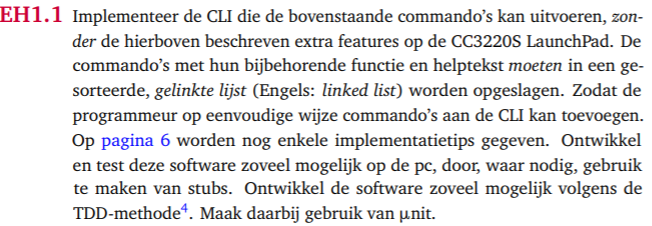


Figure 4. Deelopdracht 1

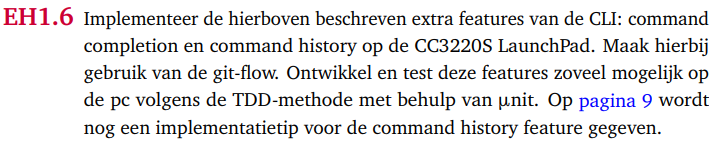
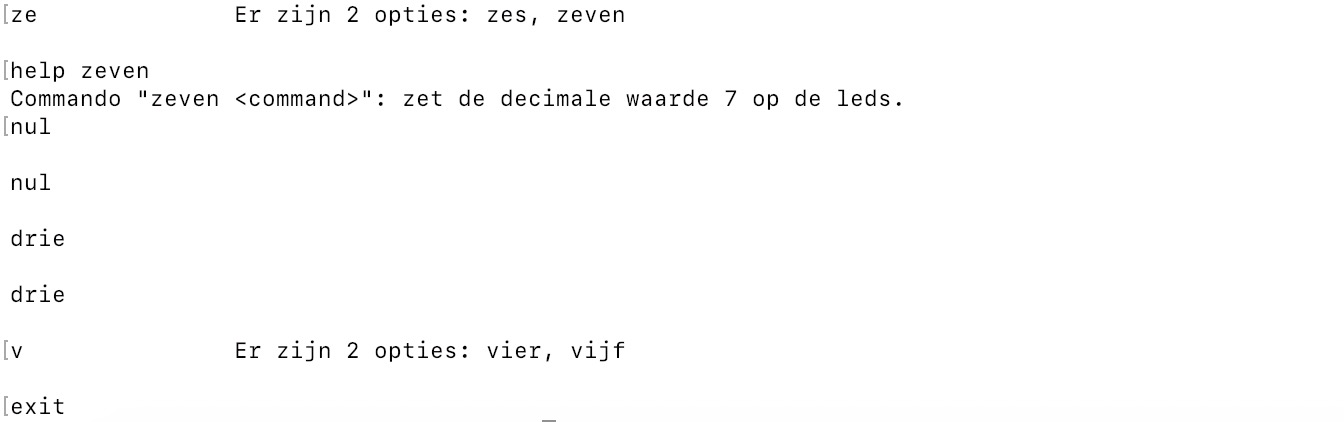


Figure 5. Deelopdracht 6

Voor het uitvoeren van de CLI zonder de extra features op de CC3220S launchpad, is als eerst zoveel mogelijk software (code) ontwikkeld op de pc in de Eclipse IDE zie [bijlagen].

Bij de ontwikkeling is er rekening gehouden met de randgevallen waaronder het dynamisch toevoegen van een commando’s met hun bijbehorende functies en helptekst in een gesorteerde lijst, gelinkte lijst [2] [snippet van de code] en zijn de verschillende mogelijkheden zoals uitgevoerd in Figure 1 meegenomen. Ook zijn er testen ontwikkeld die kunnen controleren dat een bepaalde invoer de juiste uitvoer levert [verwijzen naar een of meerdere testen].



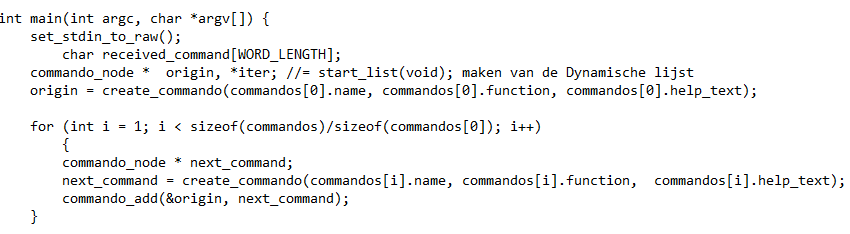


Figure 6. de dynamische linked list

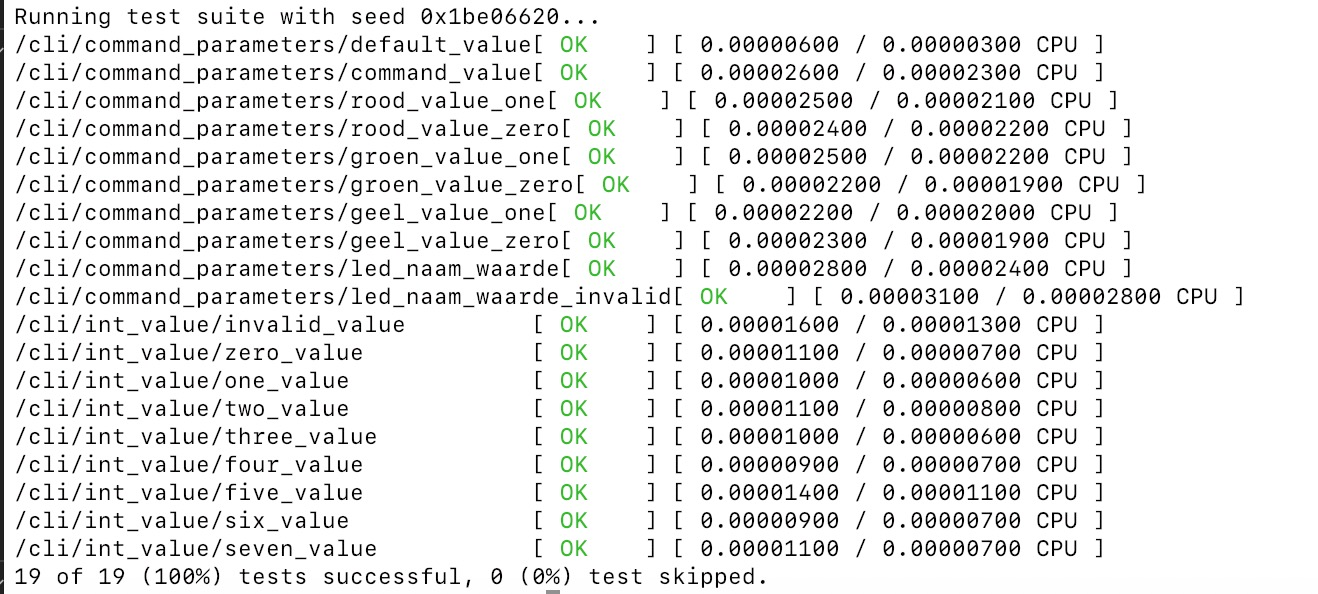


Figure 7. testen behorend bij het omzetten van de commando parameters

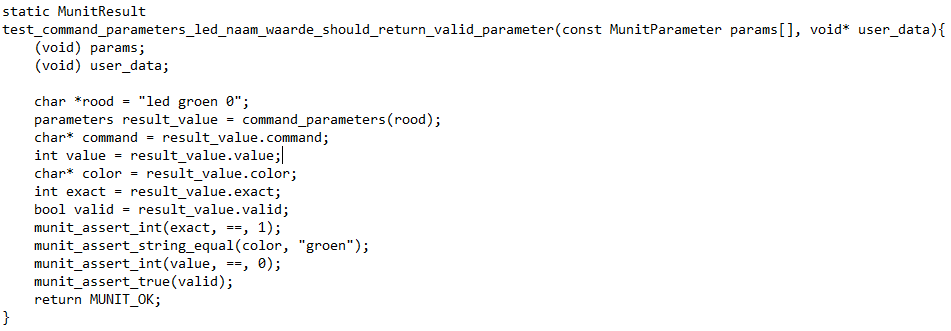


Figure 8. Testen van de paramater van led

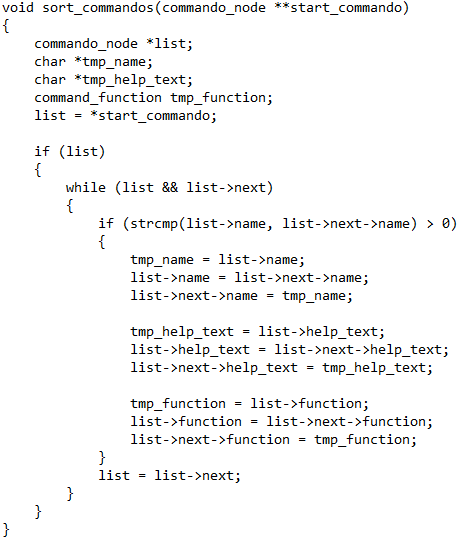


Figure 9. Sorteren van de linked list

**Deel 2**

Dit deel van het ontwerp bestaat uit de samenvoeging van opdracht EH1.2 (Figure 6) en EH1.7 (Figure 7).

Tijdens dit deel zullen er memorytesten worden uitegevoerd en indien mogelijk worden de ‘leaks’ gefixt.



Figure 10. Deelopdracht 2



Figure 11. Deelopdracht 7

Voor het testen van de ontwikkelde software op geheugenfouten wordt er gebruik gemaakt van DrMemory. DrMemory biedt de mogelijkheid om ….

**Deel 3**

Dit deel van het ontwerp bestaat uit de samenvoeging van opdracht EH1.3 (Figure 8) en EH1.8 (Figure 9).

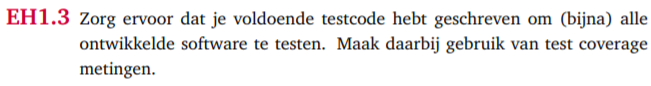
****

Figure 12. Deelopdracht 3



Figure 13. Deelopdracht 8

**Deel 4**

Dit deel van het ontwerp bestaat uit de samenvoeging van opdracht EH1.4 (Figure 10) en EH1.9 (Figure 11).

Zie test-voor testcodes

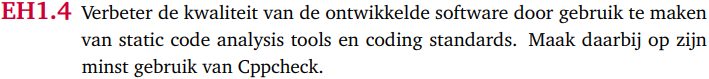


Figure 14. Deelopdracht 4

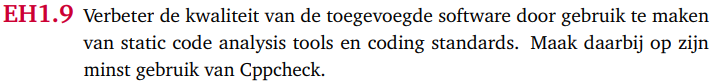


Figure 15. Deelopdracht 9

**Deel 5**

Deelopdracht 5 en 10 bevat het schrijven van dit verslag met de daarbij behorende ontwerpbeslissingen en testresultaten op zowel de pc als de CC3220S Launchpad.

Voor de verbetering van de code is er gebruik gemaakt van static code analysis tools en coding standaards met het programma Cppcheck.

# Referenties

[1]<https://bytebucket.org/HR_ELEKTRO/ems30/wiki/Opdrachten/Eindopdracht_1_Herk.pdf?rev=ac497646c2ecce58e54f48c229ef152825f7d9c4>

[2]<https://ide.geeksforgeeks.org/nuKYhx>