C Weektaak 3:

Files en Memory allocation

# Inleveren

* Het week3-1.c bestand met je code voor opdracht 1.
* Het week3-2.c bestand met je code voor opdracht 2.
* Een link naar je GitHub Repo in de beschrijving. (Met je Practicum docent hierin toegevoegd.)

# Algemene regels

1. **Te laat inleveren:** is een automatische onvoldoende.
2. Deadlines zijn onafhankelijk van het moment dat de eerste practicum les plaatsvindt. Zorg dus dat je zelfstandig al een begin maakt met de practicumopdracht van die week!
3. Alle programmeeropdrachten worden bijgehouden in [GitHub](https://www.github.com/). Zorg dat je repository privé is en dat je je docent toegang geeft.
4. Er wordt gewerkt in duo's met **Pair Programming** (d.w.z. dat beide studenten aan een laptop tegelijk werken aan dezelfde code, spreek onderling af wanneer je het typen af wisselt. Terwijl de een typt kan de ander al nadenken/opzoeken hoe het volgende deel geschreven moet worden).
   * Als één van de twee groepsleden onvoldoende kennis van het gemaakte werk kan tonen, resulteert dit in een onvoldoende voor beide groepsleden
   * Alleen werken is bij uitzondering toegestaan. Overleg dit altijd met je practicumdocent.  Werken in drietallen is niet toegestaan.
   * In het geval van een onevenredige verdeling van de werklast mag ervoor gekozen worden om een duo, in overleg met de practicumdocent, op te breken. Dit duo mag dan niet op een later moment weer bijeenkomen.
5. Code die is overgenomen van het internet (of sterk is gebaseerd op code van het internet) dient te worden voorzien van bronvermelding. Een gebrek aan bronvermelding geldt als plagiaat en zal worden gemeld bij de examencommissie.
6. Kennis uitwisselen en van elkaar leren is toegestaan en wordt aangemoedigd! Het overnemen van code geldt echter als **plagiaat** en zal worden gemeld bij de examencommissie.
7. Namen van variabelen, functies en alle andere namespaces in de scripts zijn origineel en dus zelfbedacht.
8. Alle scripts dienen te worden voorzien van **zelfgeschreven commentaar**. Bij voorkeur in het Engels, maar Nederlands is evengoed toegestaan.
9. **Herkansingen/2e gelegenheid:** De student heeft recht om elke weektaak te herkansen.
10. **Beoordeling**: Elke opdracht wordt beoordeeld met een (O)nvoldoende of (V)oldoende.
    * Voor een voldoende dienen alle weekopdrachten te zijn afgerond met een Voldoende.

Vak docenten: Roy van Leeuwen (LERO) en Jan-Wiepke Knobbe (KNJA).

# Opdrachten

**In de weektaak ga je aan de slag met Files en Memory allocation**

1. **Files**

Schrijf een C programma dat een bestand opent (.txt) en de volgende acties uitvoert.

* Het aantal karakters van het gehele bestand wordt geteld en aan het einde uitgeprint. Karakters, zoals LF en EOF, die niet zichtbaar zijn voor iemand die het bestand opent in een tekst editor moeten niet worden meegeteld.
* De tekst wordt gekopieerd naar een nieuw bestand, waarbij alle karakters 1 [ASCII](https://en.wikipedia.org/wiki/ASCII) waarde opschuiven tenzij de nieuwe waarde ervoor zorgt dat de indeling in regels wordt gewijzigd (zie onderstaande bullet)
* Zorg dat het programma ook werkt voor .txt bestanden met meer dan één regel tekst: tekst die in originele bestand op één regel staat moet op dezelfde regel in het gekopieerde bestand blijven staan.

Enkele voorbeelden van het opschuiven van ASCII-karakters:

’A’ → ’B’

’@’ → ’A’

’Z’ → ’[’

Lever het bestand in met de naam: “week3-1.c”.

1. **Memory allocation**

Schrijf een C programma dat een data structuur (“struct”) aanmaakt voor het opslaan van informatie over personen. De gewenste structuur beschrijft een persoon en is als volgt gedefinieerd:

* De structuur heeft als naam “Persoon”
* De “Persoon” structuur bevat een string om de “naam” van de “Persoon” op te slaan
* De “Persoon” structuur bevat een variabele om de “leeftijd” van de “Persoon” op te slaan

De gebruiker kan steeds weer de informatie van een extra persoon invoeren, en het programma zal deze steeds weer in een nieuwe structuur opslaan. Hiervoor zal dus ook steeds meer geheugen gereserveerd moeten worden doormiddel van realloc() ( initieel malloc() ).

Het plaatsen van de structuur elementen in niet-sequentiële geheugenadressen en de pointers hierbij in 1 blok alloceren mag ook, maar dat is puur bonus.

Na elke invoeren van een nieuw element wordt alle informatie van alle aangemaakte structuren uitgeprint.

Lever het bestand in met de naam: “week3-2.c”.