Structuur van Computerprogrammas II: Week 7

Elisa Gonzalez Boix Maarten Vandercammen Robbe De Greef

1 Dynamische allocatie

1. (**Dodona**) Schrijf een functie `char *lowercase(char *string)` die een string als input neemt en een kopie van deze string returnt waarin alle hoofdletters zijn omgevormd naar kleine letters. Een karakter omvormen naar de lowercase versie kan je doen via de functie `tolower` uit de <`ctype.h`> library. Je kan de functie `strlen` uit de <`string.h`> library gebruiken om de lengte van een string te berekenen.

2. (**Dodona**) Schrijf een functie `int *generate_primes(int n)` die een array teruggeeft van de eerste `n` priemgetallen. Op Dodona en in `oefeningen.c` op Canvas vind je de implementatie van de `print_primes` functie uit het WPO van week 3 waarop je je kan baseren om deze oefening te maken.

3. (**Dodona**) Unions laten je toe om polymorfisme na te bootsen in C. Je krijgt voor deze oefening op Dodona en in het bestand `oefeningen.c` op Canvas de definitie van de `union poly` die bestaat uit ofwel een getal, ofwel een karakter, ofwel een string. Je krijgt ook de definitie van de `struct variant` die bestaat uit zo'n `union poly` en een `enum type` dat aangeeft als welke van de drie mogelijkheden de union moet geïnterpreteerd worden, of als een `ERROR` indien geen van die drie.

   Schrijf een functie `struct variant plus(struct variant a, struct variant b)` die twee zulke `struct variant`s als argument neemt en een nieuwe `struct variant` teruggeeft. Het resultaat van deze functie hangt af van het type van de twee unions: als de unions geïnterpreteerd moeten worden als twee integers returnt de functie gewoon de som van de getallen, als ze moeten geïnterpreteerd worden als twee strings, return je de concatenatie van deze strings, en als ze moeten geïnterpreteerd worden als twee karakters, return je een string bestaande uit die twee karakters. Als de types verschillen, return je een `struct variant` met als type `ERROR`. Om twee strings te concateneren kan je het volgende doen:

   ```
   char *new_string = (char *) calloc(length, sizeof(char));
   strcat(new_string, string1);
   strcat(new_string, string2);
   ```

2 Pointers naar pointers

4. (**Dodona**) Schrijf een functie `float *calculate_all_averages(float **all_grades, int *all_lengths, int nr_of_students)` om de puntengemiddeldes van een aantal studenten te berekenen.

   We houden per student de punten die de student behaald heeft bij in een array van float getallen. Aangezien niet alle studenten evenveel vakken bijwonen, kunnen we geen gewone tweedimensionale array gebruiken om alle punten van alle studenten bij te houden. Daarom maken we gebruik van een **array van pointers** (`float **all_grades`), waarbij elke pointer wijst naar een **array van float getallen** (de punten van de student). Naast die array van pointers krijgt de functie als argument ook een **array van integers** (`int *all_lengths`) mee, die aangeeft hoe lang elke overeenkomstige array van float getallen is. Als laatste argument krijgt de functie ook het aantal studenten (`int nr_of_students`) mee.

   Als returnwaarde geeft de functie een **array van float getallen** terug die de gemiddeldes voorstellen die elke student behaald heeft.

3 Extra

5. (**Dodona**) Schrijf een functie `struct Piece **read_chess_pieces(int n)` die `n` karakters inleest uit de console, waarbij elk karakter een schaakstuk voorstelt, en een array teruggeeft van **pointers naar** de ingelezen schaakstukken. Op Dodona en in het bestand `oefeningen.c` op Canvas vind je al een definitie van de `Piece` struct waarmee je schaakstukken kan voorstellen. Een karakter inlezen uit de console kan je doen via `scanf("%c", &c)` waarbij `c` een `char` variabele is. De tabel hieronder geeft aan welke karakters overeenkomen met welke schaakstukken. Dealloceer op het einde van het programma ook al het geheugen dat voor deze oefening dynamisch werd gedealloceerd. Het dealloceren zal je niet kunnen testen in Dodona.

Karakter	Kleur	Type
`'P'`	BLACK	PAWN
`'B'`	BLACK	BISHOP
`'N'`	BLACK	KNIGHT
`'R'`	BLACK	ROOK
`'Q'`	BLACK	QUEEN
`'K'`	BLACK	KING

Karakter	Kleur	Type
`'p'`	WHITE	PAWN
`'b'`	WHITE	BISHOP
`'n'`	WHITE	KNIGHT
`'r'`	WHITE	ROOK
`'q'`	WHITE	QUEEN
`'k'`	WHITE	KING