Oefeningen “Programmeren in C”

1. Write a C program that displays a rectangle, built from the letters x, with width = 10 and height = 6:

xxxxxxxxxx

x x

x x

x x

x x

xxxxxxxxxx

2. Write a program that takes an integer value from the keyboard and returns the number with its digits reversed.

For example, given the number 7631, the program should return 1367.

3. Convert the program from exercise 1 in a function that takes the width and height as parameters.

Call that function from the main program.

4. Indicate in red the code lines that are syntactically incorrect. Indicate in green the expressions that are syntactically correct, but that are useless or might cause problems during execution.

Finally, what will appear on the display? Do not use your computer.

int i,j;

double d;

int \*ip = NULL, \*jp = NULL;

double \*dp = NULL;

i = 7;

ip = &7;

jp = &i;

\*jp = j;

\*ip = i;

ip = &j;

&i = ip;

j=4;

(\*ip)++;

&d = dp;

\*ip\*= i;

\*jp=\*&j;

ip++;

i = ip-&i;

dp = &i;

dp = ip;

&dp = &&d;

\*ip+=1;

\*ip++;

cout <<endl<< i << j << d;

printf("i = %d, j = %d",i, j);

printf("i = %d, j = %d",\*ip, \*jp);

5. Gebruik een tweedimensionale rij om het volgende probleem op te lossen.

- Een bedrijf heeft 4 verkopers (1 tot 4)

- en verkoopt 5 verschillende producten (1 tot 5).

- Elke dag brengt elke verkoper een briefje binnen voor elk verkocht product.

- Dit briefje vermeldt:

- het nr. van de verkoper

- het productnr.

- de totale verkochte waarde van dat product op die dag.

- Elke verkoper dient per dag dus tussen 0 en 5 briefjes in.

- Maak een dummy-proof programma om de informatie van al deze briefjes (over een periode van bijv. één maand)

- in te geven

- de totalen te berekenen per persoon en per product

- Alle totalen moeten opgeslagen worden in een 2-dimensionale array.

- Druk de resultaten af als tabel waarbij

- elke kolom een verkoper voorstelt

- elke rij een product voorstelt

- in de meeste rechtse kolom staan de producttotalen

- in de onderste rij staan de totalen per verkoper

6. Schrijf twee functies:

- tripleByValue, die een ingegeven getal binnenkrijgt "by value" en met 3 vermenigvuldigt.

- trypleByReference, die een ingegeven getal binnenkrijgt "by reference" en eveneens met 3 vermenigvuldigt.

Plaats de prototypes in een header file en de functiedefinities in een .C file.

Toon de resultaten van beide functies, bijv.

Geef een decimaal getal: 123.65

tripleByValue: 3 x 123.65 = 370.95

tripleByReference: 3 x 123.65 = 370.95

Bespreek de verschillen tussen beide functies.

7. Alloceer dynamisch geheugen voor een array van integers waarvan de lengte wordt opgevraagd aan de gebruiker. Controleer of de allocatie gelukt is. Experimenteer bij het testen zodat de allocatie mislukt. Als de allocatie gelukt is, vul de array dan op met de elementen van de rij van Fibonacci (an = an-1 + an-2, met a0 = 0 en a1 = 1). Toon tenslotte de inhoud van de array.

8. Een functie voert een bewerking uit op twee doubles, die als parameter worden doorgegeven. De functie geeft een double terug die het resultaat van de bewerking bevat. Daarnaast wordt ook doorgegeven welke bewerking dient te gebeuren: optellen, aftrekken, vermenigvuldigen, …. Implementeer dit op een zo flexibel mogelijke manier, zodat er achteraf nog bewerkingen kunnen bijkomen, zonder dat de functie moet aangepast worden.

9. Schrijf een C-programma dat een aantal veelvoorkomende bitmanipulatieoperaties uitvoert op een 8-bit integer (uint8\_t). Het programma moet de gebruiker vragen om invoer te geven en vervolgens de volgende acties uitvoeren:

* Bit Instellen (Set bit):
  + Vraag de gebruiker om een bitpositie (0-7) op te geven die moet worden ingesteld (op 1 gezet).
  + Gebruik bitmasking om de specifieke bitpositie in te stellen zonder de andere bits te veranderen.
* Bit Wissen (Clear bit):
  + Vraag de gebruiker om een bitpositie (0-7) op te geven die moet worden gewist (op 0 gezet).
  + Gebruik bitmasking om de specifieke bitpositie te wissen zonder de andere bits te veranderen.
* Bit Lezen (Read bit):
  + Vraag de gebruiker om een bitpositie (0-7) op te geven die moet worden gelezen.
  + Gebruik bitmasking om de waarde van het specifieke bit te lezen en geef deze weer.
* Bit Inverteren (Toggle bit):
  + Vraag de gebruiker om een bitpositie (0-7) op te geven die moet worden geïnverteerd.
  + Gebruik bitmasking om de specifieke bitpositie te inverteren zonder de andere bits te veranderen.

Verwachte Uitvoer:

Het programma zou bijvoorbeeld als volgt kunnen werken:

Voer een getal in (0-255): 172

Kies een operatie:

1. Set bit

2. Clear bit

3. Read bit

4. Toggle bit

Keuze: 4

Welke bitpositie wil je gebruiken? (0-7): 3

Het ingevoerde getal is: 172 (binair: 10101100)

Het nieuwe getal is: 164 (binair: 10100100)

Deze oefening helpt je om een diepgaand begrip te krijgen van bitmanipulatie in C, wat een essentiële vaardigheid is voor het werken met embedded systemen en low-level programmering.