Tentamen Programmeermethoden
Maandag 5 januari 2004, 14.00–17.00 uur
Universiteit Leiden — Informatica

Bij alle te schrijven functies moeten de variabelen in de heading voorkomen (niet stiekem globale variabelen gebruiken). De opgaven tellen alle vier even zwaar mee. Veel succes!

1. Deze opgave gaat over array's met `n` karakters: `char A[n];`, met `const int n = 8;`.
a. Geef een C++-functie `void sorteer (A,n)` die `A` *aflopend* sorteert met behulp van *bubblesort*.
b. Stel dat zo'n array `A` de tientallige representatie voorstelt van een positief geheel getal. Ieder karakter stelt één cijfer van het getal voor. Een voorbeeld: het getal 1273 wordt gerepresenteerd als `'0' '0' '0' '0' '1' '2' '7' '3'`, waarbij bijvoorbeeld `A[n-2] '7'` is en `A[n-1] '3'`. Schrijf een C++-functie `int getal (A,n)` die het getal oplevert dat door `A` wordt voorgesteld.
c. Schrijf een functie `void som (a,b,c,over)` die de `char`'s `a` en `b` "modulo 10 optelt" in `c`, en de boolean `over false` maakt precies als er overflow is, dat wil zeggen wanneer de som van de twee getallen groter dan 9 is — en anders `true`. Een voorbeeld: met `a` gelijk aan `'8'` en `b` gelijk aan `'7'`, moet `c` (via $8 + 7 = 15$) `'5'` worden en `over false`.
d. Schrijf een C++-functie `bool totaalsom (A,B,C,n)` die de som van de "getallen" uit de array's `A` en `B` in `C` zet, en `false` teruggeeft als er overflow optreedt, dus als de som van `A` en `B` te groot is voor `C` — en anders `true`. Als er overflow optreedt maakt het niet uit wat er in `C` terecht komt. Gebruik **c**, en niet **b**.

2. a. Bij een functie kun je te maken hebben met *call by value* en *call by reference*, en ook met *locale* en *globale* variabelen. Verder heb je ook nog *formele* en *actuele* parameters. Leg deze zes begrippen duidelijk uit.
b. Een zeker C++-programma bevat de volgende programmaregels:

```
int art (int n, int m) {
  n--; return n+m-1; }  // art
int paul (int n, int m) {
  int a = 3; m--; n += 2; a++;
  m = n + art (n,m) + art (m,n) + a;
  cout << a << m << n << endl; return a+m-n; }  // paul
```

Wat levert op (met globale variabelen `a` en `m` van type `int`):

```
a = 1; m = 4;
cout << paul (m,a) << endl; cout << a << m << endl;
```

Wat wordt er afgedrukt? Geef hierbij uiteraard uitleg.
c. We voegen twee maal een `&` toe, en wel bij de parameters in de heading van `paul`. Beantwoord opnieuw vraag **b**.
d. Voeg nu ook nog twee maal een `&` toe bij de parameters in de heading van `art`. Beantwoord opnieuw vraag **c**. Waarom zijn verschillende uitkomsten mogelijk? Geef deze.
e. Stel dat er in `art` een aanroep van `paul` moet bijkomen. Mag dit en hoe kan dit in C++ worden gerealiseerd? En is er dan sprake van recursie?

3. Gegeven is een m bij n array `cijfers`, gevuld met integers die alle een waarde hebben tussen 1 en $m \times n$. Twee voorbeelden met `m = 3` en `n = 5`:

```
 1   2   3   4   5        1   2  13   4   5
 2   4   6   8  10        2   4   3   8  10
11  12  13  14  15       11  12   6  14  15
```

a. Schrijf een C++-functie `bool tweemaal (cijfers,i,j)` die bepaalt of de getallen in rij `i` allemaal gelijk zijn aan twee keer de overeenkomstige getallen uit rij `j`. Voor het linker voorbeeld is rij `1` twee maal rij `0`, en de functie levert daar dus `true`.
b. Schrijf een C++-functie `int fraai (cijfers,m)` die nagaat in hoeverre het array precies de getallen 1 tot en met $m \times n$ bevat, en wel in oplopende volgorde van boven naar beneden en van links naar rechts (rij `0`: `1 2 3` ...). De functie geeft het aantal plekken in het array waar niet de goede waarde staat. In het linker voorbeeld zijn alleen de getallen `2`, `4`, `6` en `8` in rij `1` fout, en levert de functie dus 4 op.
c. Schrijf een C++-functie `void schuifdoor (cijfers,j,m)` die alle elementen uit kolom `j` een plek naar beneden schuift, waarbij het laatste getal op de bovenste plek komt. In het linker voorbeeld kolom `2` doorschuiven levert het rechter voorbeeld.
d. Schrijf een C++-functie `int hoevaak (cijfers,m)` die kijkt hoe vaak je het volgende kunt doen zonder het array uit te lopen (beginnend bij array-element (`0`,`0`) linksboven): ga `cijfers[0][0]` naar rechts naar (zeg) (`p`,`q`), dan `cijfers[p][q]` naar beneden, dan weer naar rechts etcetera. In het linker voorbeeld: van `1` naar `2` naar `12`, klaar: 2 keer.

4. Voor het beschrijven van alle films die gedraaid worden tijdens de jaarlijkse filmmarathon gebruiken we het type `film`. De filmmarathon is een enkelverbonden lijst van films, toegankelijk via de pointer `marathon` van type `film*`.

```
class film { // een struct mag ook
public:   char naam;       // de naam van de film
          int lengte;      // de duur van de film in minuten
          film* volgende;  // pointer naar volgende film
};  // film
```

a. Net voordat de filmmarathon begint blijkt de tweede film spoorloos verdwenen. Schrijf een C++-functie `void verwijder (marathon)`, die de tweede film uit de lijst verwijdert. Je mag aannemen dat de lijst ten minste twee films bevat.
b. Schrijf een C++-functie `void toevoegen (marathon,titel,duur)`, die een nieuwe film met naam `titel` en lengte `duur` vooraan de lijst toevoegt. Houd hierbij rekening met het feit dat de lijst mogelijk leeg is.
c. Er blijkt dat de film met de naam `titel` langer duurt dan eerst was aangenomen. Gegeven is dat deze film de eerste of de tweede in de lijst is. Schrijf een C++-functie `void verbeter (marathon,titel,duur)`, die het lengteveld van de betreffende film verandert in de juiste lengte (namelijk `duur`).
d. In de functies bij **a**, **b** en **c** staat in de heading de parameter `marathon`. Deze heb je call by value of call by reference doorgegeven (met een `&`). Maakt het voor de werking van deze functies verschil uit of die `&` erbij staat? Leg ook uit wat er bij deze twee situaties precies gebeurt tijdens executie van de betreffende functies.
e. Schrijf een C++-functie `int hoeveel (marathon)`, die het aantal films bepaalt die langer duren dan de gemiddelde lengte van de films uit de lijst. Hint: 2 keer lopen.

Zie voor cijfers `http://www.liacs.nl/home/kosters/pm/res03.txt`, omstreeks 26 januari.