Tentamen Inleiding programmeren voor LS&Ters
Vrijdag 14 maart 2003, 14.00–17.00 uur
Universiteit Leiden — Informatica

Bij de 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!
Cijfers te zijner tijd: `http://www.liacs.nl/home/kosters/lst/`

1. **a.** Schrijf een C++-functie `int rondaf (double x)` die het positieve "reële" getal `x` afrond op het dichtstbijgelegen gehele getal.
b. Schrijf een C++-functie `int wortel (double x)` die de wortel uit het positieve getal `x` als volgt benadert. Het grootste gehele getal waarvan het kwadraat nog kleiner dan of gelijk aan `x` is moet worden geretourneerd. Gebruik een `while`-loop.
c. Leg in woorden *binair zoeken* uit.
d. Gegeven twee arrays `A` en `B`, beide met 10 verschillende gehele getallen. Schrijf een C++-functie `int verschil (int A[10], int B[10])` om te bepalen hoeveel verschillende getallen er in totaal voorkomen.

2. Deze opgave gaat over array's met `n` integers: `int A[n];`, met `const int n = 15;` bijvoorbeeld. De arrays bevatten verschillende positieve getallen. Een voorbeeldrij:

```
3 6 9 2 1 12 45 77 8 5 19 88 55 40 43
```

a. Een array-element dat groter is dan zijn beide directe buren (voor de getallen aan het uiteinde: groter dan de ene directe buur) noemen we een *piek*. Schrijf een C++-functie `bool ispiek (int A[ ], int i)` die `true` teruggeeft precies als `A[i]` een piek is. Voor de voorbeeldrij zouden aanroepen `ispiek (A,14)` en `ispiek (A,7)` `true` opleveren (want `43` en `77` zijn piek), en `ispiek (A,1)` `false` (want `6` is geen piek).
b. Maak een C++-functie `int pieken (int A[ ], int n)` die het aantal pieken in `A` geeft. Gebruik **a**. In het voorbeeld is het antwoord `4` (wegens pieken `9`, `77`, `88` en `43`).
c. Analoog aan pieken zijn *dalen* gedefinieerd.
Geef een C++-functie `int dalen (int A[ ], int n)` die het aantal dalen in `A` teruggeeft. Het array `A` mag alleen via de functies van **a** en **b** benaderd worden. Voor het voorbeeld zijn het er `4`. Hint: tussen elke twee opeenvolgende pieken zit één dal.
d. Schrijf een C++-functie `bool gesorteerd (int A[ ], int n)` die `true` teruggeeft precies als `A` (op- of aflopend) gesorteerd is. Nu mag `A` alleen via de functies van **a**, **b** en **c** benaderd worden. Lukt het niet op die manier, dan mag je een eigen methode gebruiken.

3. **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:

```
void jules (int & keer, int & a) {
  a = a + 2; if ( a == r ) keer = 0;
  cout << "Jules " << keer << a << endl;
}  // jules
int wallace (int & keer, int & x) {
  int i; for ( i = 1; i <= keer; i++ ) jules (keer,x);
  cout << "Wallace " << keer << x << endl; return i;
}  // wallace
```

Gegeven de volgende programma-code:

```
r = 10; s = 4;                    // (*)
cout << wallace (r,s) << endl;    // (*)
cout << r << s << endl;
```

Wat wordt er afgedrukt? Geef hierbij uiteraard uitleg. De variabelen `r` en `s` zijn globaal en van type `int`.
c. Als **b**, maar nu met alle vier `&`'s weggelaten.
d. Weer terug naar de situatie van **b**, met de vier `&`'s erbij. Stel dat regels `(*)` vervangen worden door

```
r = 11; s = 4;
cout << wallace (s,s) << endl;
```

Leg uit waarom er nu een oneindige loop optreedt.
e. Mag een statement als `wallace (wallace (r,r),wallace (r,r));` ergens in het programma staan? En zo ja, is er dan sprake van recursie?

4. We hebben een 2-dimensionaal vierkant array `char bord[10][10]` met karakters. Het stelt een bord voor waarop een "slang" ligt. Lege vakjes worden aangegeven met een spatie (`' '`), de kop van de slang met een `'K'`, het uiteinde van de staart met een `'S'` en de overige lichaamsdelen met een `'L'`. Een voorbeeld:

```
  0 1 2 3 4 5 6 7 8 9
0     L L L
1     L   L L
2     L     L
3     L     L L
4     L       L
5     L L     S
6       L
7       L         K
8       L L L   L L
9           L L L
```

a. Schrijf een C++-functie `int lengte (char bord[10][10])` die de lengte van de slang berekend (inclusief kop en uiteinde staart). In ons voorbeeld: `27`.
b. Schrijf een functie `void zoek (char bord[10][10], char X, int & i, int & j)` die de coordinaten van `X` in `i` en `j` oplevert. In het voorbeeld, met `X` gelijk aan `'K'`, zou `i` `7` en `j` `8` moeten worden. Neem aan dat `X` precies één keer voorkomt.
c. Schrijf een functie `bool looprechts (char bord[10][10])` die de kop van de slang één vakje naar *rechts* laat gaan. Als dit niet kan, omdat je of uit het array loopt, of in de slang zelf terecht komt, wordt er niets veranderd en retourneert de functie `false`, en anders `true`. In het voorbeeld mag het, en wordt de `'K'` een vakje naar rechts verplaatst (op diens oude plek komt een `'L'`). De rest van de slang blijft onveranderd. Gebruik **b**.
d. Maak een functie `void kortin (char bord[10][10])` die bij de staart de slang één korter maakt, en een `S` zet in het nieuwe uiteinde. Neem aan dat het oude uiteinde precies één (horizontale of verticale) buur met een `'L'` erin had. Gebruik **b**.