Tentamen Programmeermethoden
Maandag 30 juli 2012, 14.00–17.00 uur
Universiteit Leiden — Informatica

Bij alle functies moeten de variabelen (constanten eventueel uitgezonderd) in de heading of als locale variabele voorkomen; vul zelf headings goed in. De opgaven tellen alle vier even zwaar mee. Succes! Cijfers: `http://www.liacs.nl/home/kosters/pm/cijf/res.html`.

1. In een array `double A[n]` staan `n` (een `const` > 1) verschillende double's ≥ 0.
NB Geef steeds (ook in de andere opgaven) de compleet ingevulde heading van de functie!
a. Schrijf een C++-functie `int telme (A,X,n)` die teruggeeft hoe vaak de double `X` in `A` voorkomt.
b. Schrijf een C++-functie `double gr (A,n)` die het (niet-afgeronde) grootste absolute verschil tussen directe buren uit `A` teruggeeft.
c. Schrijf een C++-functie `int wisseling (A,n)` die teruggeeft hoe vaak het rijtje `A` van dalend overgaat in stijgend, of andersom.
d. Schrijf een C++-functie `buso (A,n)` die het array `A` met behulp van *bubblesort aflopend* sorteert.
e. Hoeveel vergelijkingen tussen array-elementen doet de methode van **d**, en waarom is het een slechte methode?

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 onderscheiden we ook nog *formele* en *actuele* parameters. Leg deze zes begrippen duidelijk uit.
b. Gegeven een C++-programma met daarin de volgende twee functies:

```
bool laurel (bool twee, int x) {
  int i = 42; twee = ! twee;
  for ( i = x; i <= z; i++ ) { y++; x++; i++; z--; }//for
  cout << i << "," << x << "," << y << "," << z << "," << (int)twee << endl;
  return ( twee && ( i > z ) ); }//laurel
int hardy (bool een, int x) {
  int y = 42; een = ! een;
  if ( een || ! laurel (een,x) ) { een = ( y > z ); if ( een ) z = 42; }//if
  else { y = z; x = x + y; z = x - y; een = ( y > x ); }//else
  return z; }//hardy
```

Verder zijn de globale variabelen `x`, `y` en `z` gegeven (alle van type `int`) en een `bool` `b`. Voordat de functie `hardy` wordt aangeroepen hebben zij de waarde `1`, `98`, `6` en `true`, respectievelijk. Wat is dan de uitvoer van het volgende stukje programma (leg je antwoord duidelijk uit; let op het gebruik van *casting*: `(int)b`, oftewel `static_cast<int>(b)`):

```
cout << hardy (b,x) << endl; cout << x << y << z << (int)b << endl;
```

c. Als **b**, maar nu met een `&` bij de vier parameters van de functies.
d. Als **b**, dus zonder de vier `&`'s, maar nu met aanroep

```
cout << hardy ((bool)(y-98),x) << endl; cout << x << y << z << endl;
```

e. Als **d**, maar nu weer met de vier `&`'s erbij.

3. Gegeven zijn twee `n` bij `n` (een `const` > 1) arrays `Q` en `K`, met gehele getallen. Hierbij geeft `Q[i][j]` de kwaliteit van een hotel op locatie $(\texttt{i}, \texttt{j})$ aan, en `K[i][j]` de bijbehorende kosten (die alle verschillen). Rechts staat een voorbeeld met $\texttt{n} = 4$.

3	3	4	2	20	10	40	16
8	1	7	8	90	21	70	71
6	2	5	1	44	32	30	18
1	4	9	2	9	37	77	17

a. Schrijf een C++-functie `goed (Q,K,min,i,j)` die in `i` en `j` de locatie van het goedkoopste hotel met kwaliteit ten minste gelijk aan `min` oplevert. Als er geen enkel hotel met minimaal deze kwaliteit is, moeten `i` en `j` beide -1 worden. In het voorbeeld, met $\texttt{min} = 8$: $\texttt{i} = 1$ en $\texttt{j} = 3$ (kosten zijn dan 71).

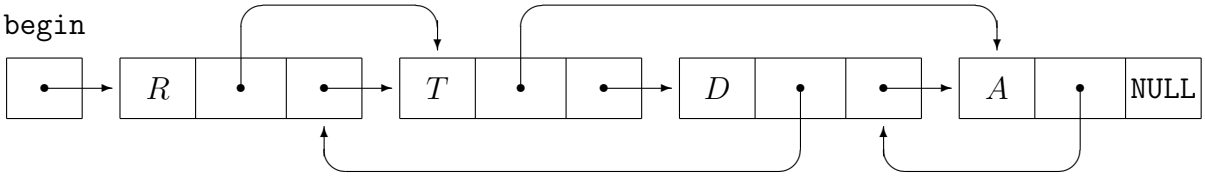
b. Iemand vermoedt dat de kosten van een hotel met kwaliteit `q` gelijk zijn aan $10 \cdot \texttt{q}$. Schrijf een C++-functie `double ver (Q,K)` die de gemiddelde absolute afwijking van deze waarde uitrekent.

c. We maken een reis, die aan de volgende eigenschappen moet voldoen. Iedere dag moet je naar een ander hotel, waarbij de kwaliteit beter moet worden; als dit niet meer kan, stopt de reis. We mogen alleen horizontaal één stap naar rechts, en als dat niet kan (omdat de kwaliteit niet beter wordt, of we uit het array vallen) verticaal één stap naar beneden. Schrijf een C++-functie `int kosten (Q,K,i,j)` die de kosten van een dergelijke reis, te beginnen op locatie $(\texttt{i}, \texttt{j})$ (met $0 \leq \texttt{i}, \texttt{j} < \texttt{n}$), uitrekent. Beginnend in $(2, 1)$ kost dat $32 + 30 + 77 = 139$.

4. Gegeven is het volgende type:

```
class duo { public: char naam; duo* alfa; duo* volg; };
```

Met behulp hiervan worden lijstjes met (verschillende) namen opgebouwd. Het veld `volg` bevat een pointer naar het volgende `duo`-object, het `alfa`-veld wijst naar het eerstvolgende alfabetisch (in ASCII-volgorde) grotere object wat `naam` betreft (vanuit het alfabetisch grootste weer naar het kleinste). Een voorbeeld (`begin` van type `duo*`):



a. Schrijf een C++-functie `verwijder (begin)` die het eerste `duo`-object uit de lijst (met `begin` van type `duo*` als ingang) netjes verwijdert, mits dit bestaat én geen cijfer als naam bevat. Er hoeft niets met de `alfa`-velden gedaan te worden.

b. Schrijf een C++-functie `voegtoe (begin,letter)` die een nieuw `duo`-object met naam `letter` erin vooraan de lijst met ingang `begin` toevoegt. Als de naam een kleine letter is, moet de bijbehorende hoofdletter erin gestopt worden. Er hoeft weer niets met de `alfa`-velden gedaan te worden.

c. Schrijf een C++-functie `verwissel (begin)` die de namen uit de eerste twee `duo`-objecten uit de lijst met ingang `begin` verwisselt, mits deze objecten bestaan. Let op: verwissel de namen, niet de objecten! Zet wel de twee uitgaande `alfa`-pointers goed.

d. In de functies bij **a**, **b** en **c** staat in de heading de pointer `begin`. 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 duidelijk uit. Zou het uitmaken als bij **c** de twee objecten verwisseld moesten worden?

e. Schrijf een C++-functie `verwijdergoed (begin)` die de functie van **a** aanvult met het in orde maken van `alfa`-pointers. Geef alleen de extra code.