Tentamen Programmeermethoden
Dinsdag 6 januari 2015, 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 `int A[n]` staan `n` (een `const` $>$ `1`) gehele getallen $>$ `0`.
a. Een rijtje getallen heet *bijna-gesorteerd* als er *exact één buur-paar* `(A[i],A[i+1])` is dat verkeerd om staat, dat wil zeggen: `A[i]` is groter dan `A[i+1]`. Schrijf een Booleaanse C++-functie `bijna (A,n,i)` die precies dan `true` teruggeeft als `A` bijna-gesorteerd is, waarbij `i` de waarde `-1` krijgt als het rijtje niet bijna-gesorteerd is en anders de index van het linker element van het falende buur-paar.
b. Neem nu aan dat in een oplopend gesorteerd rijtje een willekeurig element is vervangen door de som van de oorspronkelijke getallen. Schrijf een C++-functie `effisort (A,n)` die `A` efficiënt oplopend sorteert. Gebruik één maal de functie van **a**, en zet de elementen zonder verdere vergelijkingen tussen array-elementen op hun juiste plek.
c. Schrijf een C++-functie `busort (A,n)` die het array `A` met behulp van *bubblesort* oplopend sorteert, waarbij het algoritme stopt zodra er in een ronde geen verwisselingen waren.
d. Hoeveel vergelijkingen tussen array-elementen doet de methode van **c** minimaal en maximaal in de situatie van **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 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:

```
int ludo (int a, int b, int n) {
  int i = 42; for ( i = 0; i < n; i += 2 ) { b += a; i--; }//for
  return b;
}//ludo
int jeanine (int a, int b) {
  a = ludo (a,b,a); cout << a << "," << b << endl;
  a = ludo (a,b,a); cout << a << "," << b << endl;
  return a;
}//jeanine
```

Verder zijn de globale variabelen `x` en `y` gegeven (van type `int`). Wat is dan de uitvoer van het volgende stukje programma (leg je antwoord duidelijk uit):

```
x = 2; y = 2; y = jeanine (x,y); cout << x << "," << y << endl;
```

c. Als **b**, maar nu met een `&` ("ampersand") bij de vijf parameters van de functies.
d. Geef een eenvoudige uitdrukking voor de return-waarde van een aanroep `ludo (a,b,n)`, uitgedrukt in diens parameters. Het maakt niet uit of er `&`'s bij de parameters staan.
e. Als **d**, maar nu voor `jeanine (a,b)`. Neem aan dat er bij alle vijf parameters een `&` staat, net als bij **c**.
f. Als in de functie `ludo` ergens `a = jeanine (a,2*n);` staat, compileert het programma dan nog? Onderscheid gevallen met en zonder `&`.

3. Gegeven is een `m` bij `n` (beide `const` $>$ `1`) array `HL` met hoofdletters. Er is precies één rij in `HL` met exact twee klinkers (`A/E/I/O/U`), de overige rijen bevatten exact één klinker. Zie hiernaast voor een voorbeeld. De constantes `m` en `n` hoeven bij deze opgave niet doorgegeven te worden als parameter.

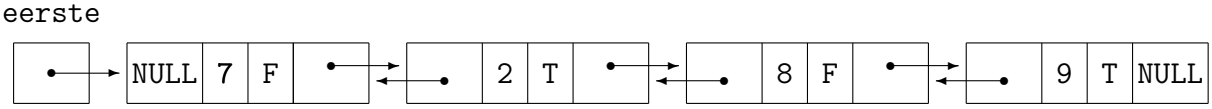
T	A	B	C
I	E	X	X
A	B	V	W
A	V	V	V

a. Schrijf een Booleaanse C++-functie `klinker (c)` die bepaalt of het karakter `c` een klinker is. Neem aan dat `c` een hoofdletter is. De functie is handig voor **b**, **c** en **d**.
b. Schrijf een C++-functie `aantal (HL)` die teruggeeft hoeveel klinkers er in `HL` direct horizontaal naast elkaar staan. Laat de functie stoppen zodra het antwoord bekend is. Het aantal is `0` of `2`, in het voorbeeld: `2`.
c. Schrijf een Booleaanse C++-functie `kliko (HL,i1,i2,j)` die precies dan `true` teruggeeft als in de `j`-de kolom van `HL` tussen rijen `i1` en `i2` alleen maar klinkers staan, de grenzen inbegrepen. Neem aan dat $0 \leq$ `i1` $\leq$ `i2` $<$ `m` en $0 \leq$ `j` $<$ `n`. In het voorbeeld levert `kliko (HL,1,3,0) true` op, en `kliko (HL,0,2,1)` geeft `false`.
d. Schrijf een Booleaanse C++-functie `verti (HL)` die precies dan `true` teruggeeft als je vanaf de eerste rij de laatste rij van `HL` kunt bereiken, beginnend en eindigend met een klinker, door herhaald horizontaal of verticaal direct aangrenzende buur-klinkers te bezoeken. In het voorbeeld: `true`, via `A-E-I-A-A`. Gebruik de functie van **c**.

4. Gegeven is het volgende type:

```
class mens { public: mens* vorig; int nr; bool weg; mens* volg; };
```

Hiermee wordt een dubbel-verbonden lijst van mensen gemaakt. Het veld `volg` bevat een pointer naar het volgende `mens`-object, en `vorig` naar het vorige. Een voorbeeld (`eerste` van type `mens*`), waarbij `F` voor `false` en `T` voor `true` staat:



a. Schrijf een C++-functie `verwijder (eerste)` die het voorste `mens`-object uit de structuur dat door `eerste` van type `mens*` wordt aangewezen, netjes verwijdert — mits het bestaat.
b. Schrijf een C++-functie `voegtoe (eerste,mensnr)` die een nieuw `mens`-object met `mensnr` erin vooraan in de lijst met ingang `eerste` toevoegt. De waarde van `weg` moet `false` worden.
c. Schrijf een C++-functie `wissel (eerste)` die de mens-nummers van de twee voorste mensen omwisselt, mits het eerste getal groter is dan het tweede (zoals in het voorbeeld: `7` $>$ `2`). Controleer of de lijst wel minstens twee objecten heeft.
d. In de functies bij **a**, **b** en **c** staat in de heading een pointer. 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? Mag het, moet het? Leg duidelijk uit.
e. Schrijf een C++-functie `opschonen (eerste)` die alle `mens`-objecten waarvoor `weg` op `true` staat verwijdert. In het voorbeeld worden tweede en vierde object verwijderd.

Gelukkig `ludo (20,15,100)`!