Tentamen Programmeermethoden
Donderdag 25 maart 2021
13:00–16:00 uur Informatica

Universiteit Leiden
The Netherlands

Bij alle functies moeten de variabelen (constanten eventueel uitgezonderd) in de heading of locaal voorkomen; vul zelf headings goed in. De te behalen punten (totaal 100) staan tussen haakjes bij de opgaven. Succes! Cijfers: te zijner tijd via Brightspace/uSis.

1. (25) In het array `int A[n]` staan `n` (een `const int` ≥ 1) verschillende gehele getallen.
a. (4) Schrijf een efficiënte Booleaanse C++-functie `pos (A,n)` die `true` geeft als alle elementen in array `A` niet negatief zijn (dus ≥ 0), en anders `false`.
b. (5) Schrijf een C++-functie `int tel (A,n,v)` die bepaalt hoeveel elementen in array `A` strikt kleiner zijn dan het gegeven gehele getal `v`.
c. (8) Schrijf een C++-functie `sorteer (A,n)` die het array `A` oplopend sorteert door voor ieder array-element te kijken op welke plek dat element moet komen te staan en zonodig te verwisselen. Ga naar het volgende array-element zodra het vorige goed staat. Maak gebruik van de functie van **b**.
d. (3) Leg met een voorbeeld uit waarom het belangrijk is dat alle array-elementen van elkaar verschillen voor de functie van **c**.
e. (5) Hoeveel vergelijkingen tussen array-elementen worden minimaal/maximaal uitgevoerd in de functie van **c**? Tel de vergelijkingen in de aanroepen van **b** mee. Geef voor beide situaties een voorbeeld met `n` $= 10$.

2. (25) **a**. (6) 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. (5) Gegeven een C++-programma met daarin de volgende twee functies:

```
int thomas (bool a, int b, int c) {
  int t = 42;
  if ( a ) { t = b; b = c; c = t; t++; }//if
  b--; c--; t = b + c;
  cout << b << "," << c << "," << t + 3 << endl; return t + 3; }//thomas
int guy (int x, int y) {
  int i, totaal = 0;
  for ( i = 1; i <= x; i++ ) totaal += thomas (x < y,x,y);
  cout << i << "," << x << "," << y << endl; return totaal; }//guy
```

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

```
p = 3; q = 5; cout << guy (p,q) << ","; cout << p << "," << q << endl;
```

c. (3) Wat merkt een gebruiker in het algemeen bij de werking van de functie `thomas` als het `if`-statement wordt weggelaten?
d. (7) Als vraag **b**, maar nu met een `&` toegevoegd bij de parameters `b` en `c` van `thomas` en bij de parameters `x` en `y` van `guy`.
e. (4) Compileert de code als in `thomas` ergens `b = guy (thomas (b < c,c - 1,b),c);` staat? Onderscheid situaties met/zonder `&` bij de betrokken parameters.

3. (25) Gegeven is een `m` bij `n` (beide `const int` > 0; ze hoeven bij deze opgave niet te worden doorgegeven als parameter) array `Z` met gehele getallen. Een 0 staat voor water, een getal > 0 voor land. Hiernaast een voorbeeld met $\texttt{m} = 4$ en $\texttt{n} = 5$.

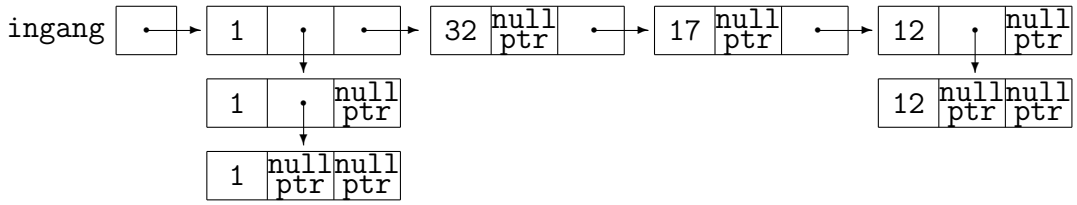
```
18 17  0 33 99
 0 11  0 77  3
 0  5  0  0 12
87  8 19 44  0
```

a. (7) Schrijf een Booleaanse C++-functie `meer (Z,i,j)` die bepaalt of er een meer in `Z` is. Een *meer* bestaat uit drie of meer locaties die onderling horizontaal en/of verticaal verbonden zijn. Het voorbeeld bevat één meer, ter grootte 4. In `i` en `j` moeten de coordinaten van een locatie in een meer worden teruggegeven. Als er geen meer is: beide -1. In het voorbeeld: `i` gelijk aan 2 en `j` gelijk aan 3 (bijvoorbeeld).
b. (10) Schrijf een C++-functie `vul (Z,i,j,waarde)` die de nullen uit het meer waarin het punt $(\texttt{i}, \texttt{j})$ ligt door het gehele getal `waarde` < 0 vervangt. Neem aan dat $(\texttt{i}, \texttt{j})$ inderdaad deel uitmaakt van een meer.
c. (8) Schrijf een C++-functie `allemeren (Z)` die de verschillende meren in `Z` geheel vult met -1, -2, enzovoorts. In het voorbeeld alleen 4 keer -1 voor het bij **a** bedoelde meer.

4. (25) Gegeven is het volgende type:

```
class object { public: int info; object* zelf; object* volg; };
```

Met behulp hiervan houden we een lijst bij. Dubbele objecten worden opgeslagen in een tweede (verticale) lijst per object, toegankelijk via de pointer `zelf`. Een voorbeeld, met `ingang` van type `object*`:



a. (7) Schrijf een C++-functie `voegtoe (ingang,x)` die een nieuw object met getal `x` erin vooraan toevoegt aan de lijst met ingang `ingang`. Als `x` gelijk is aan het eerste getal uit de lijst, voeg dan het nieuwe object ergens toe aan de eerste verticale lijst.
b. (7) Schrijf een C++-functie `verwijder (ingang)` die één van de getallen uit de eerste verticale lijst met objecten uit de lijst met ingang `ingang` netjes verwijdert, indien dit bestaat. In het voorbeeld: één van de drie 1-en, kies zelf welke. Denk aan de lege lijst en aan de situatie waarin de eerste verticale lijst precies één getal heeft.
c. (8) Schrijf een C++-functie `int dubbel (ingang)` die het totaal aantal dubbele objecten in de lijst met ingang `ingang` oplevert. Voor het voorbeeld: $2 + 0 + 0 + 1 = 3$.
d. (3) 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.