Programmeermethoden

Oude tentamens

Walter Kosters en Jonathan Vis

week 14: 11-15 december 2023

www.liacs.leidenuniv.nl/~kosterswa/pm/

Opgave 1 van het tentamen van 6 januari 2014:

In een array `int A[n]` staan `n` (een `const` > 0) gehele getallen.

a. Schrijf een C++-functie `hoevaak (A,X,n)` die teruggeeft hoe vaak het gehele getal `X` in het array `A` voorkomt.

b. Schrijf een Booleaanse C++-functie `uniek (A,n)` die precies dan `true` teruggeeft als geen enkel getal twee maal (of vaker) voorkomt in `A`, en anders `false`. Hierbij moet de functie van **a** *zinvol* gebruikt worden (hoe vaak komt `A[i]` voor?).

c. Schrijf een C++-functie `meest (A,n)` die het meest voorkomende getal uit `A` teruggeeft. Als er verschillende kandidaten zijn (bijvoorbeeld voor het array `17 12 30 12 42 30`) moet het kleinste getal dat het meest voorkomt worden geretourneerd. In het voorbeeld is dit `12` (dat even vaak voorkomt als `30`). Maak opnieuw gebruik van de functie van **a**.

d. Schrijf een C++-functie `sorteer (A,n)` die de getallen in `A` zodanig ordent dat voor alle getallen (behalve het laatste) geldt dat ze hooguit even vaak voorkomen als hun rechter buurman. Tip: pas de C++-code voor *bubblesort* eenvoudig aan; gebruik **a**.

e. Hoe vaak wordt de functie `hoevaak` aangeroepen in **d**?

soms dus ook iets over "complexiteit": $O(n^2)$ ...

Uitwerking Opgave 1

```
a. int hoevaak (int A[ ], int X, int n) {
     int i, teller = 0;
     for ( i = 0; i < n; i++ ) if ( X == A[i] ) teller++;
     return teller;
   }//hoevaak
b. bool uniek (int A[ ], int n) {
     int i;
     for ( i = 0; i < n; i++ )
       if ( hoevaak (A,A[i],n) > 1 ) return false;
     return true;
   }//uniek
c. int meest (int A[ ], int n) {
     int i, tel, vaak = A[0], aantal = hoevaak (A,A[0],n);
     for ( i = 1; i < n; i++ ) {
       tel = hoevaak (A,A[i],n);
       if ( tel > aantal || ( tel == aantal && A[i] < vaak ) ) {
         vaak = A[i]; aantal = tel; }//if
     }//for
     return vaak;
   }//meest
d. void sorteer (int A[ ], int n) {
     int i, j, temp;
     for ( i = 1; i < n; i++ )
       for ( j = 0; j < n-i; j++ )
         if ( hoevaak (A,A[j],n) > hoevaak (A,A[j+1],n) ) {
           temp = A[j]; A[j] = A[j+1]; A[j+1] = temp; }//if
   }//sorteer
e. 2 ( 1+2+...+n-1) = n (n-1) keer
```

Opgave 2 van het tentamen van 6 januari 2015:

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 `&`.

Uitwerking Opgave 2b

"main"		"jeanine"		"ludo"				
x	y	a	b	a	b	n	i	
2	2							
		2	2					aanroep `jeanine`
				2	2	2	42	eerste aanroep `ludo`
							0	
					4		1	
					6		2	geeft 6 terug
		6						`cout`: 6,2
				6	2	6	42	tweede aanroep `ludo`
							0	
					8		1	
					⋮		⋮	
					38		6	geeft 38 terug
		38						`cout`: 38,2; geeft 38 terug
	38							`cout`: 2,38

tijd ↓

Uitwerking Opgave 2c

"main"		"ludo"	tijd ↓
x = a uit jeanine = a uit ludo = n uit ludo	y = b uit jeanine = b uit ludo	i	
2	2		
			aanroep jeanine
		42	eerste aanroep ludo
		0	
	4	1	
	6	2	geeft 6 terug
6			cout: 6,6
		42	tweede aanroep ludo
		0	
	12	1	
	⋮	⋮	
	42	6	geeft 42 terug
42			cout: 42,42; geeft 42 terug
	42		cout: 42,42

d. $\mathtt{n} \cdot \mathtt{a} + \mathtt{b}$ (als $\mathtt{n} \geq 0$) **e**. $(\mathtt{a}^2 + \mathtt{b})^2 + \mathtt{a}^2 + \mathtt{b}$

f. Dan moet de tweede parameter van `jeanine` call by value zijn, en een prototype van `jeanine` moet boven `ludo` staan.

Opgave 3 van het tentamen van 16 maart 2017:

Gegeven is het twee-dimensionale array `int afstand[n][n];`, met zekere `const int n` ≥ 2. Er geldt dat `afstand[i][j]` > 0 de afstand is tussen de plaatsen `i` en `j` met `i` $\neq$ `j`, waarbij `afstand[i][i]` `0` is, en de afstand tussen `i` en `j` even groot is als die tussen `j` en `i`. Een voorbeeld met `n = 4` staat hiernaast.

0	3	7	9
3	0	4	14
7	4	0	8
9	14	8	0

a. Schrijf een C++-functie `bool reis (afstand,km)` die kijkt of er een rondreis van `i` naar `j` naar `k` naar `i` is (voor willekeurige onderling verschillende plaatsen `i`, `j` en `k`) die in totaal precies afstand `km` heeft. In het voorbeeld zou voor `26` het antwoord `true` zijn: van `0` naar `1` naar `3` naar `0` (dat is `3 + 14 + 9` $=$ `26`).

b. Schrijf een C++-functie `int verste (afstand,i)` die het nummer van de verst van `i` afgelegen plaats oplevert. In het voorbeeld, met `i` $=$ `3`, is dat `1` (wegens afstand `14`). Als er meerdere plaatsen voldoen: die met de grootste index. Neem aan dat `0` $\leq$ `i` $<$ `n`.

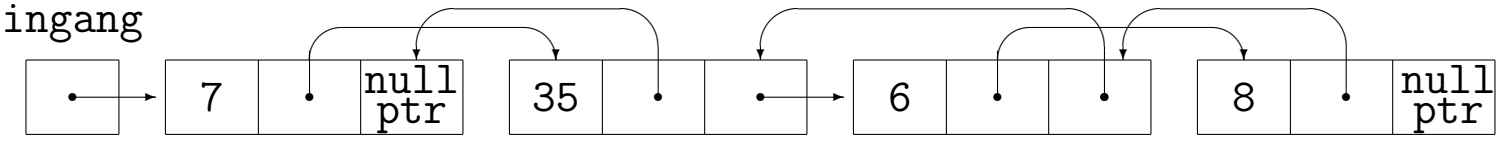
c. Schrijf een C++-functie `int hoever (afstand,i)` die bepaalt hoeveel je in totaal reist als je begint in `i`, dan steeds naar de verst gelegen plaats gaat, en stopt zodra je ergens komt waar je al eerder geweest was. In het voorbeeld, met `i` $=$ `0`, is het antwoord `9 + 14 + 14` $=$ `37` (van `0` naar `3` naar `1` naar `3`). Neem aan dat `0` $\leq$ `i` $<$ `n`. Hint: gebruik een Booleaans hulparray.

Opgave 4 van het tentamen van 3 januari 2019:

Gegeven is het volgende type:

```
class object { public: int info; object* volg1; object* volg2; };
```

Met behulp hiervan kan een lijst van objecten worden opgebouwd, bestaande uit vakjes met een getal, en twee pointers. Precies een van deze twee wijst naar het volgende object, de andere naar het vorige — maar je weet niet welke. Een voorbeeld, met `ingang` van type `object*`:



a. (6(*)) Schrijf een C++-functie `voegtoe (ingang,getal)` die een nieuw object met `getal` erin netjes vooraan de lijst met ingang `ingang` toevoegt. Je mag zelf kiezen welke van de twee pointers in het nieuwe object naar vorige en volgende object wijst. Zet in het oude eerste object (als dat bestaat) de terugwijzende pointer goed.

(*) maximaal aantal punten voor dit onderdeel; per som 25; tentamen $4 \cdot 25 = 100$

Opgave 4 van het tentamen van 3 januari 2019, vervolg:

b. (6) Schrijf een C++-functie `verwijder (ingang)` die het eerste object uit de lijst met ingang `ingang` netjes verwijdert, indien dit bestaat.

c. (4) Schrijf een C++-functie `hoogop (ingang)` die als er minstens twee objecten zijn en als het `info`-veld van het eerste object oneven is, dit ophoogt met het `info`-veld van het tweede object. In het voorbeeld: 7 wordt 42.

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.

e. (6) Schrijf een C++-functie `repareer (ingang)` die ervoor zorgt dat na afloop alle `volg1`-pointers naar het volgende, en alle `volg2`-pointers naar het vorige object wijzen.

www.liacs.leidenuniv.nl/~kosterswa/pm/tentamens.php

En: video's

Oud tentamen: 5.1.2016

Werkcollege Programmeermethoden 14 december 2023

Tentamen 5 januari 2016

1. In een array `int A[n]` staan `n` (een `const` ≥ 1) gehele getallen > 0.
a. (5) Schrijf een Booleaanse C++-functie `hoe (A,X,n)` die bepaalt of er een getal dat *hooguit* 1 van het gehele getal `X` verschilt in het array `A` voorkomt. Gebruik geen `(f)abs`.
b. (8) Schrijf een C++-functie `int langste (A,n,gem)` die de lengte uitrekent van een langste stijgende (of beter: niet-dalende) aaneengesloten deelrij in het array `A`. Hierbij moet `gem` het op de gebruikelijke manier naar een geheel getal afgeronde gemiddelde zijn van de getallen in de betreffende deelrij. Voor het array `1 7 1 1 2 7 6 3` zou het antwoord 4 zijn, namelijk de lengte van de deelrij `1 1 2 7`. En `gem` moet 3 worden (via 11/4). Als er meer deelrijen dezelfde maximale lengte realiseren: het gemiddelde van de eerste.
c. (8) Schrijf een C++-functie `busort (A,n)` die het array `A` met de volgende variant van *bubblesort* oplopend sorteert. In de eerste ronde wordt het gehele array van links naar rechts doorlopen, in de tweede ronde van rechts naar links, in de derde van links naar rechts, etcetera. Stop als er tijdens een ronde geen verwisselingen waren.
d. (4) Hoeveel vergelijkingen tussen array-elementen worden minimaal/maximaal uitgevoerd in de functie van **c**? Geef voor beide situaties een voorbeeld.

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. (6) Gegeven een C++-programma met daarin de volgende twee functies:

```
int hillary (int n, int m) {
  n--; return n+m-1; }//hillary
int donald (int n, int m) {
  int a = 6; m--; n += 2; a++;
  m = n + hillary (n,m) + hillary (m,n) + a;
  cout << a << "," << m << "," << n << endl; return a+n+m; }//donald
```

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

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

c. (5) We voegen nu vier maal een `&` toe bij de parameters in de heading van `hillary` en `donald`. Beantwoord opnieuw vraag **b**; leg uit waarom verschillende uitkomsten mogelijk zijn, en geef deze.
d. (4) Als in de functie `hillary` ergens `a = donald (42,a-a);` staat, compileert het programma dan nog? Onderscheid gevallen met en zonder `&`.
e. (4) Geef een eenvoudige uitdrukking voor de functiewaarde van `donald (r,s)`, uitgedrukt in `r` en `s`, voor de situatie zonder `&`'s (net zoals bij **b**).

3. (25) Gegeven is een `m` bij `n` (beide `const` > 0) array `M` met gehele getallen ≥ 0. Hierbij geeft `M[i][j]` ≥ 0 het aantal mensen ter plaatse `(i,j)` aan (met $0 \leq$ `i` $<$ `m` en $0 \leq$ `j` $<$ `n`). Zie hiernaast voor een voorbeeld met `m` $= 4$ en `n` $= 5$. De constantes `m` en `n` hoeven bij deze opgave niet doorgegeven te worden als parameter.

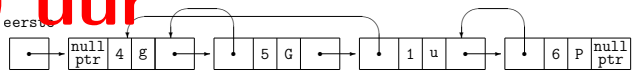
3	0	4	2	0
7	2	4	4	2
1	2	0	4	1
6	9	3	1	0

a. (7) Schrijf een C++-functie `int druk (M,som)` die de index van de drukste kolom, dat wil zeggen de kolom met de grootste som, geeft. In `som` moet de desbetreffende som komen. In het voorbeeld kolom 0, met `som` gelijk aan 17. Als er meerdere kolommen dit maximum realiseren, geef degene met de hoogste kolomindex.
b. (8) Schrijf een C++-functie `int duos (M)` die het aantal horizontaal of verticaal direct aan elkaar grenzende (eventueel overlappende) paren geeft, waarbij de een precies het dubbele van de ander is. In het voorbeeld: 4–2, 2–4, 4–2, 1–2, 2–4, 2–1: 6 stuks.
c. (10) We lopen als volgt door het array `M`. Start bij `(i,j)`. De waarde van dit array-element geeft aan hoeveel plaatsen naar rechts je gaat; de waarde van de nieuwe plek bepaalt hoeveel plaatsen je omlaag gaat, enzovoorts. Als je rechts het array uitloopt kom je in dezelfde rij links weer binnen, en analoog voor de verticale richting. De wandeling stopt als je ergens komt waar je al eerder bent geweest (in het bijzonder bij een verplaatsing van 0). Schrijf een C++-functie `int aantal (M,i,j)` die het aantal stappen uitrekent; in `i` en `j` moeten de coördinaten van het laatst bezochte punt komen. In het voorbeeld, beginnend bij `(0,0)` eerst 3 naar rechts, dan 2 omlaag, dan 4 naar rechts, dan 0 omlaag en klaar: antwoord 4, en `(i,j)` $= (2,2)$. Hint: gebruik een Booleaans hulparray.

4. (25) Gegeven is het volgende type:

```
class kamer { public: kamer* vorig; int nr; char naam; kamer* volg; };
```

Hiermee wordt een lijst van kamers gemaakt (`naam` is een kleine letter of hoofdletter). Het veld `volg` bevat een pointer naar het volgende `kamer`-object, en `vorig` naar het vorige *of* voor-vorige, *of* `nullptr` als die er beide niet zijn. Een voorbeeld (`eerste` van type `kamer*`):



a. (5) Schrijf een C++-functie `voegtoe (eerste,kamernr,kamernm)` die een nieuw `kamer`-object met `kamernr` en `kamernm` erin vooraan in de lijst met ingang `eerste` toevoegt. Zet de `vorig`-pointer van het oude voorste object (als dat bestond) ook goed.
b. (5) Schrijf een C++-functie `verwijder (eerste)` die het voorste `kamer`-object uit de structuur die door `eerste` wordt aangewezen, netjes verwijdert — mits het bestaat. Zet eventuele `vorig`-pointers die er naar wijzen goed.
c. (5) Schrijf een C++-functie `wissel (eerste)` die de kamer-nummers van de twee voorste kamers omwisselt, mits de ene letter de hoofdletter van de andere is (zoals in het voorbeeld; 4 en 5 worden verwisseld). Controleer of de lijst minstens twee objecten heeft.
d. (4) 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. (6) Schrijf een C++-functie `herstel (eerste)` die alle `vorig`-pointers, behalve die van de voorste twee objecten (zo die al bestaan), naar het voor-vorige object laat wijzen. In het voorbeeld zou de pointer bij het object met `P` erin naar het object met `G` erin moeten gaan wijzen, verder verandert er niets.

Zie www.liacs.leidenuniv.nl/~kosterswa/pm/tentamens.php voor meer vragen en antwoorden!

Opgave 2 van het tentamen van 4 januari 2013:

b. Gegeven een C++-programma met daarin de volgende twee functies:

```
bool mark (int a, int b) {
  int z;
  a = a + b; b = a - b; a = a - b; cout << "M" << a << "," << b << endl;
  z += 10; return ( a < b );
}//mark
int diederik (int b, int a) {
  bool temp; if ( mark (b,a) ) a += 2;
  while ( a > 0 ) { temp = mark (a,b); a--; cout << a << "," << b << endl; }
  z += 10; return ( a + b + 2 );
}//diederik
```

Verder zijn de globale variabelen `x`, `y` en `z` gegeven (van type `int`). Wat is dan de uitvoer van het volgende stukje programma (leg je antwoord duidelijk uit; tip: 9 komma's):

```
x = 1; y = 3; z = 1; x = diederik (y,z); cout << x << "," << y << "," << z;
```

c. Als **b**, maar nu met een `&` ("ampersand") bij de vier parameters van de functies.

d. Als **c**, dus met vier `&`'s erbij, voor:

```
x = 1; y = 3; z = 1; y = diederik (y,y); cout << x << "," << y << "," << z;
```

e. Als in `mark` ergens `a = diederik (static_cast<int>(mark (a,b)),y);` staat, compileert het programma dan nog? Onderscheid gevallen met en zonder `&`.

Uitwerking Opgave 2b

tijd ↓

"main"			"diederik"			"mark"		
x	y	z	a	b	temp	a	b	
1	3	1						
			1	3	?			aanroep `diederik`
						3	1	eerste aanroep `mark` (`z` doet er niet toe)
						1	3	cout: M1,3; geeft `true` terug
			3					"`a` += 2;" uit `diederik`
						3	3	tweede aanroep `mark`
						3	3	cout: M3,3; geeft `false` terug
			2		`false`			cout: 2,3
						2	3	derde aanroep `mark`
						3	2	cout: M3,2; geeft `false` terug
			1		`false`			cout: 1,3
						1	3	vierde aanroep `mark`
						3	1	cout: M3,1; geeft `false` terug
			0		`false`			cout: 0,3
		11						geeft 5 terug aan `x`
5								cout: 5,3,11

Uitwerking Opgave 2c

"main"			"mark"	
x	y = b uit diederik	z = a uit diederik	z	tijd ↓
1	3	1		
				aanroep diederik
	= a uit mark	= b uit mark	?	eerste aanroep mark
	1	3	10	cout: M1,3; geeft true terug
		5		"a += 2;" uit diederik
	= b uit mark	= a uit mark	?	tweede aanroep mark
	5	1	10	cout: M1,5; geeft true terug
		0		cout: 0,5
		10		geeft 17 terug aan x
17				cout: 17,5,10

"main"			"mark"	
x	y = a uit diederik = b uit diederik	z	z	tijd ↓
1	3	1		
				aanroep diederik
	= a uit mark		?	aanroep mark
	= b uit mark			
	0		10	cout: M0,0; geeft false terug
		11		geeft 2 terug aan y
	2			cout: 1,2,11

e. Dit mag alleen als er geen & staat voor de eerste parameter van diederik, en er boven mark een prototype van diederik wordt gezet.

Tentamen

- laatste werkcollege: donderdag 14 december 2023, 13:15 uur, Snellius 174: oude tentamens

- tentamen:
  woensdag 17 januari 2024, 13:00–16:00 uur,
  Universitair Sportcentrum; aanmelden!
  hertentamen:
  donderdag 28 maart 2024, 13:00–16:00 uur

- www.liacs.leidenuniv.nl/~kosterswa/pm/