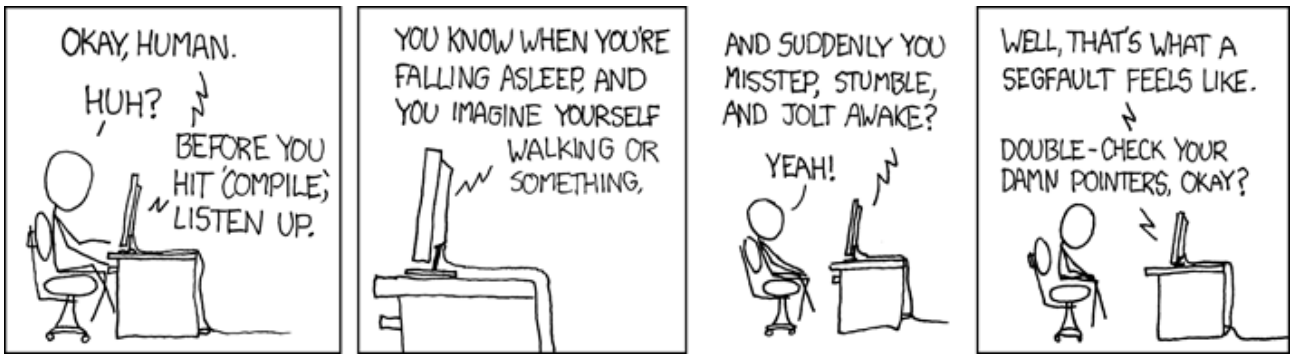
DLF Pointerworkshop

Opgaven

2 maart 2016



XKCD # 371

In dit document staan een aantal oude tentamenopgaven om te oefenen voor het hertentamen PR. Bij de selectie is rekening gehouden met de opgaven die het meeste fout zijn gegaan op het tentamen, en waarbij de meeste vooruitgang te behalen is. Zo blijken vooral de opgaven over 2D-arrays en pointers lastig, traditioneel opgave **3** en **4** van het tentamen. De opgaven op het tentamen hebben vaak meerdere onderdelen, aangegeven met de bekende letters (a, b, c, d, e). De laatste van deze onderdelen zijn het moeilijkst en bouwen voort op de eerste paar. Ze zijn dan ook niet per se nodig om een voldoende te halen, maar dienen om een 'topcijfer' te halen. In deze opgavenbundel ligt de focus op de eerstgenoemde onderdelen.

2D-arrays

Opgave 1

Gegeven is een `m` bij `n` (beide `const` > 0) array `puzzel`, gevuld met *verschillende kleine letters*. In het voorbeeld geldt `m` = 3 en `n` = 6.

q	w	e	r	t	y
a	s	d	f	g	h
z	x	c	v	b	n

Voorbeeld

a. Schrijf een C++-functie `opvolger (puzzel,letter,p,q)` die het coördinaten-paar `(p,q)` oplevert waarin de alfabetisch eerste letter uit `puzzel` zit die op de `char letter` volgt. Mocht `letter` de alfabetisch laatste zijn, dan moeten `p` en `q` beide -1 worden. In het voorbeeld: `letter = 't'` zou ze op `p = 2` en `q = 3` moeten zetten[1].

b. Schrijf een C++-functie `alpha (puzzel)` die de letters uit de `puzzel` in alfabetische volgorde afdrukt. In het voorbeeld: `abcdefghnqrstvwxyz`.

c. Schrijf een Booleaanse C++-functie `komtvoor (puzzel,een,twee)` die bepaalt of de `char`'s `een` en `twee` als horizontale of verticale buren voorkomen in `puzzel`. In het voorbeeld: `true` voor `'d'` en `'e'`, maar `false` voor `'q'` en `'y'`.

Opgave 2

Gegeven is een `m` bij `n` (beide `const` > 0) array `temper`; `temper[i][j]` stelt de temperatuur (tussen -50 en $+50$) op punt `(i,j)` voor, waarbij 99 staat voor "onbekend".

12	-1	-3	-3	-6	9
99	-1	2	2	2	7
99	-1	-1	99	99	99

Voorbeeld

a. Schrijf een C++-functie `int tel (temper)` die telt hoeveel rijen van de matrix zowel een temperatuur > 0 als een temperatuur < 0 bevatten. De waarde 99 telt niet mee. In het voorbeeld: 2.

b. Schrijf een C++-functie `bool schaatsbaan (temper,i,j,p,q)` die kijkt of je vanuit punt `(i,j)` in punt (p,q) kunt komen, waarbij je herhaald naar een verticaal aangrenzend punt mag gaan, of herhaald naar een horizontaal aangrenzend punt (maar niet gemengd).
Alle gebruikte punten moeten een temperatuur < 0 hebben. Als `(i,j)` = `(p,q)` is het geen schaatsbaan. In het voorbeeld is er een schaatsbaan van (0,2) naar (0,4), maar niet van (0,4) naar (1,1), en ook niet van (0,2) naar (2,2). Neem aan dat $0 \leq$ `i,p` $<$ `m` en $0 \leq$ `j,q` $<$ `n`.

c. Schrijf een C++-functie `bool kruis (temper)` die bepaalt of er een horizontale en een verticale schaatsbaan zijn die elkaar *kruisen*, dat wil zeggen precies één punt gemeenschappelijk hebben. Gebruik **b.**. In het voorbeeld is het resultaat `true`.

[1] Hier wegens onduidelijkheden weggelaten uit originele opgave: `(de 'v')`. Zie `http://www.liacs.nl/~kosters/pm/pmtenfeb2011.pdf`

Opgave 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 `(i,j)` aan, en `K[i][j]` de bijbehorende kosten (die alle verschillen). Zie het voorbeeld met `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
		`Q`				`K`	

Voorbeeld

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: `min = 8` resulteert in `i = 1` en `j = 3` (kosten zijn dan 71).

b. Tycho vermoedt dat de kosten van een hotel met kwaliteit `q` gelijk zijn aan $10 \cdot$ `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 `(i,j)` (met $0 \leq$ `i,j` $\leq$ `n`), uitrekent. Beginnend in `(2,1)` kost dat $32 + 30 + 77 = 139$.

Pointers

XKCD # 379

Opgave 1

Bekijk het volgende programma:

```
1  #include <iostream>
2  using namespace std;
3
4  void tjatja (int* & r, int* & s) {
5      r = new int();
6      *r = 1;
7      *s = 96;
8  } //tjatja
9
10 int main ( ) {
11     int* p; int* q;
12     p = new int();
13     *p = 3;
14     q = new int();
15     *q = 4;
16     cout << *p << *q << endl;
17     tjatja (p,q);
18     cout << *p << *q << endl;
19     return 0;
20 } //main
```

a. Wat stellen `p`, `*p` en `&p` voor?
b. Wat is de uitvoer van dit programma (met uitleg/schetsen)?
c. Idem als in de functie-heading van `tjatja` het symbool & tweemaal wordt weggelaten.
d. Hoe kun je het programma (met en zonder de &'s van **c**) aanpassen zodat er na afloop geen loze vakjes meer in het geheugen rondhangen? (gebruik `delete`)

Opgave 2

Gegeven:

```
class mens {
    public:
        char naam[30];
        mens* kind;
}; //mens
```

Deze "datastructuur" is bedoeld om een ouder-kind-kleinkind-...-relatie te representeren, een soort stamboom dus.

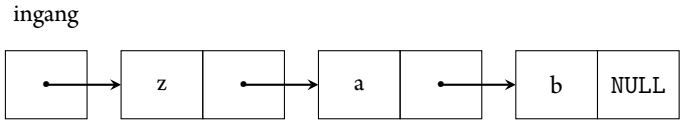
a. Creëer een variabele van type `mens`, die een persoon geheten Ellen voorstelt.
b. Voeg haar dochter Vera toe.
c. Voeg haar kleindochter Minke toe.
d. Voeg de moeder van Ellen, Tineke, toe. Doe eventueel **a** opnieuw. Maak een tekening van de tot hier gemaakte structuur.
e. Hoe kun je de naam van het kleinkind van Tineke vinden?

Opgave 3

Gegeven:

```
class vakje {
    public:
        char info;
        vakje* volgende;
};
```

Voorbeeld:



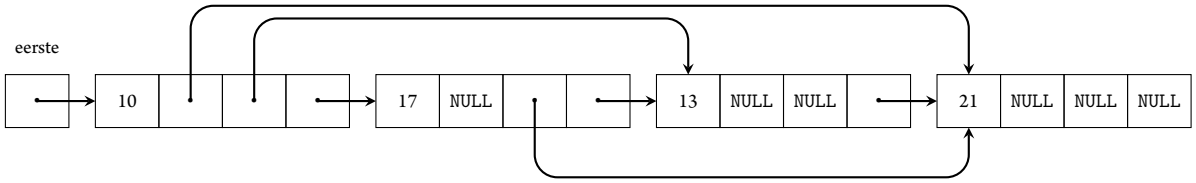
a. Schrijf een functie die de inhoud van de vakjeslijst afdrukt, gescheiden door spaties. In het voorbeeld: `z a b`.
b. Schrijf een functie die als input een letter heeft, en die een nieuw vakje achteraan toevoegt met deze letter.
c. Schrijf een functie die als input een letter heeft, en die een nieuw vakje vooraan toevoegt met deze letter.
d. Schrijf een functie die de eerst letter verwisselt met de tweede letter: (1) door de waarden van de infovelden aan te passen en (2) door de lijstvolgorde aan te passen.
e. Schrijf een functie die het laatste vakje in de lijst verwijdert, als dit er is.
f. Schrijf een functie die als input een letter heeft, en het eerste vakje met deze letter uit de lijst verwijdert. Als het vakje gevonden en verwijderd is, geeft de functie `true` terug. Als de letter niet in de lijst zit, geeft hij `false` terug.
g. Schrijf een functie die als input een letter heeft, en middels een `int` teruggeeft hoe vaak de letter in de lijst voorkomt.

Opgave 4

Gegeven is het volgende type:

```
class hetgetal {
    public:
        int info;
        int point;
        hetgetal* volg;
};
```

Met behulp hiervan worden lijstjes met getallen opgebouwd. Het veld `volg` bevat een pointer naar het volgende `hetgetal`-object, het `point`-veld bevat hetzelfde getal als het `info`-veld van het door deze pointer aangewezen `hetgetal`-object (0 als dat `NULL` is).
Een voorbeeld (`eerste` van type `hetgetal*`):



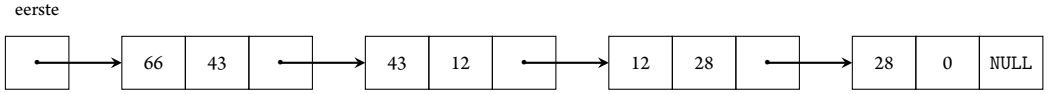
a. Schrijf een C++-functie `verwissel (eerste)` die de twee eerste *objecten* --- indien aanwezig --- van de lijst (met `eerste` van type `hetgetal*` als ingang) verwisselt. De `point`-velden moeten zonodig aangepast worden.
b. Schrijf een C++-functie `voegtoe (eerste,get)` die een nieuw `hetgetal`-object met `get` in het `info`-veld vooraan de lijst (met `eerste` van type `hetgetal*` als ingang) toevoegt. Het `point`-veld moet op de juiste manier gevuld worden.
c. Schrijf een C++-functie `verwijder (eerste)` die het *tweede* `hetgetal`-object uit de lijst (met `eerste` van type `hetgetal*` als ingang) verwijdert, mits dat er is. Denk dus aan de lege lijst en aan een lijst met één element. Let ook weer op de `point`-velden.

Opgave 5

Gegeven is het volgende type:

```
class pr {
    public:
        int prijs;
        pr* volg1;
        pr* volg2;
        pr* volg3;
};
```

Met behulp hiervan worden lijstjes met prijzen opgebouwd. Het veld `volg1` bevat een pointer naar het volgende `pr`-object, `volg2` wijst naar het daarop volgende object, en `volg3` naar het daar weer op volgende object (soms `NULL`).
Een voorbeeld (`begin` van type `pr*`; in de objecten staan de pointers in volgorde `volg3`,`volg2`,`volg1` getekend):



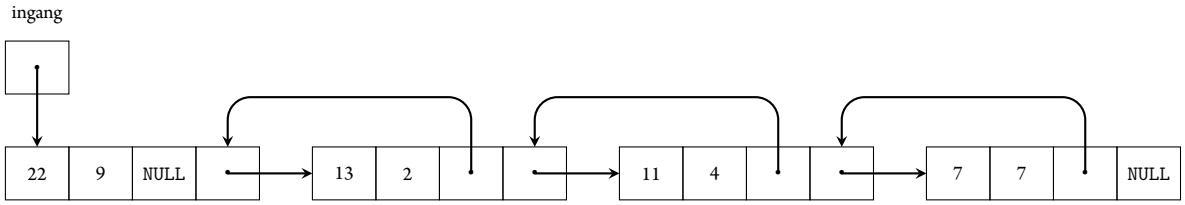
a. Schrijf een C++-functie `verwijder (begin)` die het eerste `pr`-object uit de lijst (met `begin` van type `pr*` als ingang) netjes verwijdert, mits dat object bestaat en de prijs erin *even* is.
b. Schrijf een C++-functie `voegtoe (begin,prijsje)` die een nieuw `pr`-object met prijs `prijsje` erin vooraan de lijst met ingang `begin` toevoegt. Zet wederom alle pointers goed.
c. Schrijf een C++-functie `verwissel (begin)` die de eerste twee `pr`-objecten uit de lijst met ingang `begin` verwisselt, mits deze objecten bestaan. Let op: verwissel de objecten, niet de inhouden! Zet uiteraard wel alle pointers goed.

Opgave 6

Gegeven is het volgende type:

```
class info {
    public:
        info* volg;
        info* vorig;
        int som;
        int getal;
};
```

Met behulp hiervan worden rijtjes (lijstjes) met getal-getal combinaties opgebouwd. Het veld `volg` bevat een pointer naar het *volgende* object in de lijst (of `NULL`), `vorig` bevat een pointer naar het *vorige* object (of, bij het eerste object, `NULL`). Het `som`-veld moet de som van alle `getal`-velden vanaf (en inclusief) het huidige object bevatten. Een voorbeeld (`ingang` van type `info*`), waarbij `volg` de meest rechtse pointer in ieder object is (bijvoorbeeld, $13 = 2 + 4 + 7$):



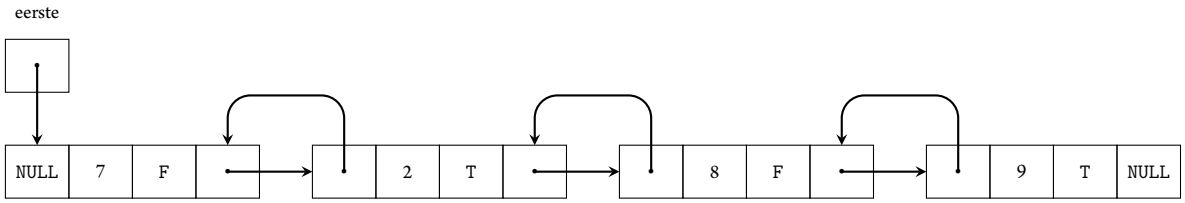
a. Schrijf een C++-functie `voegtoe (ingang,get)` die een nieuw object met het getal `get` erin vooraan de structuur (met `ingang` van type `info*` als ingang) toevoegt. Denk ook aan de `vorig`-pointers (mits de originele lijst minstens één object had). En geef het `som`-veld de juiste waarde.
b. Schrijf een C++-functie `verwijder (ingang)` die het eerste object uit de lijst (met `ingang` van type `info *` als ingang) verwijdert indien in dat object alleen oneven getallen zitten (in `som`- en `getal`-veld). Denk aan de lege lijst, en een eventuele `vorig`-pointer die `NULL` moet worden.
c. Schrijf een C++-functie `verwissel (ingang)` die de `getal`-velden uit het eerste en tweee object verwisselt (dus ook de inhouden), indien deze bestaan, en anders niets doet. De `som`-waarden moeten ook in orde gemaakt worden.

Opgave 7

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.

Herkomst van opgaven

2D-arrays

1: 23 februari 2011
2: 24 februari 2010
3: 30 juli 2012

Pointers

1, 2 en 3: Op basis van opgaven van voorgaande pointerworkshop
4: 24 februari 2010
5: 28 maart 2012
6: 4 augustus 2008
7: 6 januari 2015