Derde programmeeropdracht: Bloemen plukken
Algoritmiek voorjaar 2016, Universiteit Leiden

Inleiding

Melania is bloemen aan het plukken in het veld, verschillende soorten, meerdere bloemen van eenzelfde soort, een heel boeket. Een beetje spannend is het wel, want als ze klaar is met plukken, zal ze haar vriend Donald een bloemensoort laten uitkiezen. Vervolgens zal Melania met 'hij houdt van me, hij houdt niet van me' één voor één de bloemen van die soort opzij leggen. Als het bij de laatste bloem van die soort 'hij houdt van me' is (als er een oneven aantal bloemen van die soort was), zullen Donald en Melania trouwen. Zo niet, dan gaan ze uit elkaar. Melania hoopt op het eerste, dus ze doet haar best om van alle soorten een oneven aantal exemplaren in haar boeket te krijgen. Kun jij haar helpen?

Specificatie

Het veld waarin Melania bloemen plukt is rechthoekig van vorm, en is opgedeeld in vierkante vakken die allemaal even groot zijn. Het veld is hoogstens 100 vakken hoog en hoogstens 100 vakken breed. Melania begint in de linker bovenhoek van het veld en gaat iedere keer óf naar het rechterbuurvak óf naar het benedenbuurvak. Ze verlaat het veld wanneer ze in de rechter benedenhoek is aangekomen. In elk vak staat één bloem. De soort waartoe een bloem behoort, wordt aangegeven met een cijfer s met $0 \leq s \leq 7$. In elk vak waar Melania komt, plukt ze de betreffende bloem. Het gaat er dus om een route te bepalen van de linker bovenhoek naar de rechter benedenhoek, waarop van zoveel mogelijk bloemen een oneven aantal exemplaren voorkomt. Merk op dat het getal 0 (nul) even is.

Voorbeeld

Beschouw het volgende veld, waarbij de route van Melania loopt via de roodgedrukte cijfers.

0	3	6	1	4	7	0	1	2	3	4	5
1	4	7	2	5	6	7	0	1	2	3	4
2	5	0	3	6	5	6	7	0	1	2	3
3	6	1	4	7	4	5	6	7	0	1	2
4	7	2	5	0	3	4	5	6	7	0	1
5	0	3	6	1	7	5	3	1	7	5	3
6	1	4	7	2	6	4	2	0	6	4	2
7	2	5	0	3	5	3	1	7	5	3	1
0	3	6	1	4	4	2	0	6	4	2	0
1	4	7	2	5	3	1	7	5	3	1	7
2	5	0	3	6	2	0	6	4	2	0	6

In dit geval zou Melania op het volgende boeket komen: drie keer bloem 0, twee keer bloem 1, vier keer bloem 2, twee keer bloem 3, twee keer bloem 4, twee keer bloem 5, drie keer bloem 6, vier keer bloem 7. Dit is geen optimaal boeket.

C++-programma

Schrijf een C++-programma dat

- de gebruiker vraagt om de naam van een invoer-tekstbestand, en de naam van een uitvoer-tekstbestand,
- uit het invoer-tekstbestand de beschrijving van een veld met bloemen inleest,
- met behulp van *bottom-up* dynamisch programmeren een optimale route voor Melania bepaalt van de linker bovenhoek naar de rechter benedenhoek in het veld, dat wil zeggen: een route waarop ze van zoveel mogelijk soorten bloemen een oneven aantal exemplaren kan plukken,
- deze optimale route met bijbehorende aantallen naar het uitvoer-tekstbestand schrijft.

Voor een precieze specificatie van invoer en uitvoer, zie verderop.

Bij het dynamisch programmeren is het de bedoeling om per vak in het veld bij te houden welke aantallen bloemen je van de verschillende soorten al in je boeket kunt hebben. De exacte aantallen zijn niet van belang; het is voldoende om van elk soort bloemen bij te houden of je er een even aantal of een oneven aantal van kunt hebben. Acht bits zijn daarvoor voldoende, en acht bits komen overeen met de getallen 0 tot en met 255. Houd voor elk vak en voor elke mogelijke bitstring (getal tussen 0 en 255) bij of je in het vak een boeket kunt hebben dat met die bitstring overeenkomt. **Gebruik daarvoor een driedimensionaal array (hoogte $\times$ breedte $\times$ 256) van booleans**. Bij 'In het verslag' gaan we in meer detail in op dit array.

Je programma moet onder Linux bij LIACS getest zijn en werken.

Invoer en uitvoer

De invoer-tekstbestanden zijn van de volgende vorm:

- een regel met twee getallen h en b, die voldoen aan $1 \leq h \leq 100$ en $1 \leq b \leq 100$: respectievelijk de hoogte en de breedte van het veld,
- h regels met elk b getallen s, die voldoen aan $0 \leq s \leq 7$: de bloemsoorten van links naar rechts in de betreffende rij van het veld. De eerste regel correspondeert met de bovenste rij, de laatste regel met de onderste rij.

De getallen die op een zelfde regel staan, worden steeds gescheiden door een spatie. Op

```
http://www.liacs.leidenuniv.nl/~graafjmde/ALGO/
```

staat ter illustratie een invoerbestand dat met bovenstaand voorbeeld overeenkomt.

Wellicht ten overvloede: als je in C++ `ifstream fin` gebruikt voor je invoer-tekstbestand, dan kun je heel eenvoudig een getal inlezen, b.v. met

```
fin >> getal;
```

De uitvoer-tekstbestanden moeten van de volgende vorm zijn (zodat ze door een controle-programma kunnen worden ingelezen):

- een regel met een getal m, dat voldoet aan $0 \leq m \leq 8$: het maximum aantal soorten bloemen waarvan Melania een oneven aantal exemplaren kan plukken,

- $h+b-1$ regels, waarbij elke regel (in volgorde) overeenkomt met een vak in een optimale route van Melania.[1] Elk van deze regels bevat twee getallen i en j, die voldoen aan $1 \leq i \leq h$ en $1 \leq j \leq b$, overeenkomend met het j-de vak (vanaf links) in de i-de rij (vanaf boven). De eerste regel bevat dus altijd de getallen 1 en 1, de laatste regel de getallen h en b,

- een regel met acht getallen die het boeket beschrijven dat Melania op de bovenbeschreven route samenstelt: het aantal bloemen van soort 0, het aantal bloemen van soort 1, ..., het aantal bloemen van soort 7 (in die volgorde). Het gaat nu dus niet om de bits van even/oneven, maar om de echte aantallen.

Ook in de uitvoer worden getallen op een zelfde regel steeds gescheiden door (alleen) een spatie.

Wellicht ten overvloede: als je in C++ `ofstream fuit` gebruikt voor je uitvoer-tekstbestand, dan kun je heel eenvoudig een getal wegschrijven, b.v. met

```
fuit << getal;
```

In het verslag

Je verslag bevat

- (uiteraard) een introductie, met de probleemstelling,

- een beschrijving en toelichting van de recurrente betrekking die je hebt gebruikt bij het dynamisch programmeren. Laat mogelijk$(i, j, s_7, s_6, \ldots, s_1, s_0)$ weergeven of het mogelijk is om in vak (i, j) van het veld een boeket te hebben dat met de bitstring $s_7 s_6 \ldots s_1 s_0$ overeenkomt. [2] Hoe bereken je mogelijk$(i, j, s_7, s_6, \ldots, s_1, s_0)$ dan (uit de waardes in andere vakken)?

- uitleg hoe je uit de waardes van mogelijk$(i, j, s_7, s_6, \ldots, s_1, s_0)$ het maximale aantal soorten bloemen met een oneven aantal aan het eind van de route haalt,

- uitleg hoe je vervolgens een optimale route bepaalt,

- een analyse van zowel de tijdcomplexiteit als de ruimtecomplexiteit van het gebruikte algoritme voor dynamisch programmeren. Allereerst voor de eigenlijke situatie, waarin er acht soorten bloemen zijn, maar vervolgens ook voor een situatie dat er $n \geq 1$ soorten bloemen zouden zijn,

- een (voor Melania) zo slecht mogelijk veld met hoogstens 10×10 vakken met bloemen. Dat wil zeggen: een $h \times b$ veld met $1 \leq h \leq 10$ en $1 \leq b \leq 10$, waarin elk van de acht soorten bloemen minstens één keer voorkomt, waarvoor Melania (zelfs bij een optimale route) van zo min mogelijk soorten een oneven aantal kan plukken. De exacte waardes van h en b mag je zelf bepalen. Beredeneer ook *waarom* dit een zo slecht mogelijk veld is. Om hoeveel soorten met een oneven aantal gaat het dan?

[1] Het is niet zo moeilijk in te zien dat elke toegestane route inderdaad bestaat uit $h + b - 1$ vakken.

[2] In je programma wordt dit een array-element `mogelijk[i][j][k]`, waarbij k het getal is dat met de bitstring $s_7 s_6 \ldots s_1 s_0$ gerepresenteerd wordt.

Voor u beschikbaar ...

Om je te helpen bij deze opdracht worden op de website van het vak,

`http://www.liacs.leidenuniv.nl/~graafjmde/ALGO/`

de volgende vier bestanden gezet:

`veld1.in`, een invoer-tekstbestand dat met het voorbeeldveld met bloemen hierboven overeenkomt,

`veld1.uit`, een mogelijk uitvoer-tekstbestand bij `veld1.in`, [3]

`testbits.cc`, een programma om te spelen met bitrepresentaties van getallen, inclusief twee functies `getBit` en `setBit` die je zo mag kopiëren.

`genererereerbloemen.cc`, een programma om een random veld met bloemen te genereren, waarbij je waardes voor h en b kunt opgeven.

Eventuele verdere aanvullingen of tips bij de programmeeropdracht komen ook op de hierboven vermelde website te staan. Er is dit keer geen skeletprogramma beschikbaar. Je zult je programma dus zelf netjes moeten opzetten. Op de website komen te zijner tijd ook de behaalde cijfers te staan.

In te leveren

Uiterste (!) inleverdatum: dinsdag 17 mei 2016, uiterlijk 11.00. Het programma (inclusief, indien van toepassing, `Makefile`) per e-mail sturen naar: `j.m.de.graaf@liacs.leidenuniv.nl`. Zorg dat het onderwerp van je mail begint met "[ALGO]", dat scheelt de docent een hoop gezoek. Het verslag (inclusief het programma!) moet op papier worden ingeleverd in de daartoe bestemde doos met opschrift Algoritmiek in de postkamer van Informatica, kamer 156. Voor elke week te laat inleveren gaat er een punt van het cijfer af. Vermeld overal duidelijk de namen van de makers.

Normering: verslag 3 punten; commentaar en layout 1 punt; modulaire opbouw en OOP 1 punt; werking 5 punten.

[3] Merk op dat er verschillende mogelijke optimale routes voor Melania kunnen zijn, zodat er ook verschillende mogelijke uitvoer-tekstbestanden kunnen zijn. Verschillende routes kunnen verschillende boeketten opleveren, maar het aantal soorten bloemen met een oneven aantal is natuurlijk bij al die optimale boeketten hetzelfde.