EXAMEN: **Computergebruik**

1e Bachelor Informatica
prof. dr. Peter Dawyndt
groep 3

maandag 20-08-2012, 08:30
academiejaar 2011-2012
tweede zittijd

Opgave 1

Eenvoudig rekenwerk met de Unix commandlijn? Het kan! Gebruik filters, I/O redirection en pipes om telkens een commando samen te stellen dat uitvoer genereert conform onderstaande beschrijvingen. Hierbij is het toegelaten om gebruik te maken van `sed`, maar niet van andere programmeerbare filters zoals `awk`, `perl`, ... Vermijd dat je commando's (tijdelijke) bestanden aanmaken binnen het bestandssysteem, tenzij dat expliciet gevraagd wordt.

1. Veronderstel dat je beschikt over 4 zwarte en 6 blauwe blokken, die los van hun kleur onderling niet van elkaar verschillen. Hoeveel verschillende manieren bestaan er dan om deze blokken naast elkaar te zetten? Hoeveel van deze rijen hebben geen twee zwarte blokken naast elkaar? Er bestaan eenvoudige wiskundige oplossingen voor deze vragen, maar het is ook mogelijk om via de Unix commandlijn alle oplossingen te genereren en te tellen. Als vertrekpunt voor het genereren van alle oplossingen kan je gebruik maken van het *brace expansion* mechanisme van de `bash` commandlijn.

   (a) Geef een commando dat met behulp van brace expansion alle verschillende mogelijkheden genereert om 4 zwarte en 6 blauwe blokken naast elkaar te zetten, en het aantal mogelijkheden uitschrijft naar standaard uitvoer.

   (b) Geef een commando dat met behulp van brace expansion alle verschillende mogelijkheden genereert om 4 zwarte en 6 blauwe blokken naast elkaar te zetten, zonder dat daarbij twee zwarte blokken naast elkaar staan, en het aantal mogelijkheden uitschrijft naar standaard uitvoer.

2. De formule van Leibniz stelt dat

$$\sum_{n=0}^{\infty} \frac{(-1)^n}{2n+1} = \frac{\pi}{4}$$

   Hiermee kunnen we dus de waarde van π benaderen door bijvoorbeeld de volgende partieelsom te berekenen

$$\frac{4}{1} - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \ldots + \frac{4}{997} - \frac{4}{999} \approx \pi$$

   (a) Het tekstbestand `oneven.txt` bevat alle oneven getallen tussen 1 en 999, elk op een afzonderlijk regel. Vul onderstaand commando aan, zodat het resultaat van bovenstaande partieelsom naar standaard uitvoer wordt uitgeschreven

```
$ cat oneven.txt | ...
3.13959265558978323856
```

   (b) Vervang in bovenstaand commando het fragment `cat oneven.txt` door een commando dat de oneven getallen tussen 1 en 999 elk op een afzonderlijke naar standaard uitvoer uitschrijft, zonder daarbij evenwel gebruik te maken van het bestand `oneven.txt`.

3. Geef een commando dat, als het in een tekstbestand `coderegels` geplaatst wordt, resulteert in een `bash` shell script dat het totaal aantal regels uitschrijft van alle gewone bestanden uit een gegeven directory en alle subdirectories daarvan, en waarvan de bestandsnaam eindigt op een gegeven bestandsextensie (die in de bestandsnaam dus telkens wordt voorafgegaan door een punt). De directory en bestandextensie moeten als twee afzonderlijke argumenten aan het shell script doorgegeven worden. Het shell script mag nooit iets uitschrijven naar standard error. Onderstaande sessie geeft voorbeelduitvoer wanneer het shell script wordt uitgevoerd op `helios`. Daaruit leiden we bijvoorbeeld af dat de directory `/usr/share` op `helios` 1156503 regels Pythoncode en slechts 5985 regels Javacode bevat.

```
$ bash coderegels /usr py
1560098
$ bash coderegels /usr/bin py
863
$ bash coderegels /usr/share py
1156503
$ bash coderegels /usr/share pl
111178
$ bash coderegels /usr/share java
5985
```

4. Het tekstbestand `raven.txt` bevat de tekst van het bekende gedicht *The Raven* van Edgar Allan Poe. Het tekstbestand `stopwoorden.txt` bevat een alfabetisch gerangschikte lijst van woorden die elk op een afzonderlijke regel staan. Vul onderstaand commando aan, zodat een lijst van de 30 meest voorkomende woorden in het gedicht wordt uitgeschreven. Deze woorden staan elk op een afzonderlijke regel, gerangschikt volgens afnemend aantal voorkomens. Woorden die even vaak voorkomen moeten in alfabetisch volgorde opgelijst worden. Naast het woord zelf, bevat elke regel ook nog het aantal voorkomens van dat woord, voorafgegaan door een dubbelpunt (`:`). De woorden van het gedicht bestaan uit de langst mogelijke reeksen van letters, enkele aanhalingstekens (`'`) en koppeltekens (`-`), waarbij vooraan en achteraan een woord nooit aanhalingstekens of koppeltekens voorkomen. Hoofdletters moeten omgezet worden naar kleine letters. Woorden die bestaan uit minder dan drie letters of die voorkomen in het bestand `stopwoorden.txt` mogen niet opgelijst worden.

```
$ cat raven.txt | ...
door:14
chamber:11
bird:10
nevermore:10
raven:10
lenore:8
soul:7
thy:7
bust:6
word:6
quoth:5
tapping:5
angels:4
floor:4
prophet:4
thee:4
bore:3
darkness:3
fancy:3
heart:3
implore:3
lamp-light:3
leave:3
```

```
maiden:3
named:3
o'er:3
perched:3
rapping:3
sad:3
sat:3
```

Opgave 2

Gegeven is een tekstbestand `athletes.txt` dat de persoonsgegevens bevat van een aantal atleten die deelnamen aan de olympische spelen van Londen 2012. Elke regel van dit bestand bevat informatie over één atleet. Deze informatie bestaat uit de volgende velden, die van elkaar worden gescheiden door een tab: *i*) familienaam, *ii*) voornaam, *iii*) lengte (in cm), *iv*) gewicht (in kg), *v*) land, *vi*) landcode, *vii*) geslacht (`M` voor mannen en `W` voor vrouwen), *viii*) discipline en *ix*) wedstrijden. Het laatste veld bevat een lijst van één of meer wedstrijden waaraan de atleet deelnam. Deze worden van elkaar gescheiden door een tab. De overige informatievelden bevatten zelf geen tabs of komma's. Bovenaan het tekstbestand staan optioneel een aantal commentaarregels, die worden voorafgegaan door een hekje (`#`).
Gevraagd wordt om, gebruik makend van de teksteditor `vi` (of `vim`), een reeks (substitutie)commando's op te stellen die achtereenvolgens de volgende opdrachten uitvoeren. Probeer voor elke opdracht zo weinig mogelijk commando's te gebruiken en zorg er voor dat elk van deze commando's bestaat uit zo weinig mogelijk tekens. De commando's mogen ook geen wijzigingen aanbrengen aan de commentaarregels. Alle wijzigingen moeten na elkaar uitgevoerd worden.

1. Zorg er voor dat de lijst van wedstrijden gescheiden wordt door een komma (`,`) en een spatie, in plaats van een tab. Zo moeten de regels

   ```
   Pipes   Ben      204   ...   Volleyball    Men's Volleyball
   ADAMS   ANTOINE  180   ...   Athletics     Men's 200m      Men's 100m      Men's 4 x 100m Relay
   ```

   omgezet worden naar

   ```
   Pipes   Ben      204   ...   Volleyball    Men's Volleyball
   ADAMS   ANTOINE  180   ...   Athletics     Men's 200m, Men's 100m, Men's 4 x 100m Relay
   ```

   Opmerking: In bovenstaande voorbeelden werden de velden *iv*) tot en met *vii*) vervangen door drie puntjes (`...`) om de weergave van de regels niet te breed te maken. De inhoud van deze velden moet uiteraard behouden blijven.

2. Zorg er voor dat de velden gescheiden worden door een puntkomma (`;`) in plaats van een tab, en verwijder alle spaties die vooraan en achteraan een veld staan. Vervang ook overal het land `Team GB` door `Great Britain` en het land `Team USA` door `United States of America`. De twee regels uit het voorgaande voorbeeld moeten dus omgezet worden naar

   ```
   Pipes;Ben;204;90;Great Britain;GBR;M;Volleyball;Men's Volleyball
   ADAMS;ANTOINE;180;79;Saint Kitts and Nevis;SKN;M;Athletics;Men's 200m, Men's 100m, Men's 4 x 100m Relay
   ```

3. Voeg de velden met de voor- en familienaam samen tot één enkel veld. In dit samengestelde veld komt eerst de voornaam, gevolgd door een spatie en daarna de familienaam. Zorg er voor dat enkel de eerste letter van de voornaam en de eerste letter van de familienaam met een hoofdletter geschreven worden, de rest met kleine letter. Dit levert het volgende resultaat op

   ```
   Ben Pipes;204;90;Great Britain;GBR;M;Volleyball;Men's Volleyball
   Antoine Adams;180;79;Saint Kitts and Nevis;SKN;M;Athletics;Men's 200m, Men's 100m, Men's 4 x 100m Relay
   ```

4. Verwijder alle regels waarvan het land een oneven aantal klinkers bevat. Dat betekent dus onder meer dat de voorbeeldregel voor de atleet Ben Pipes moet verwijderd worden.

5. Sorteer vanuit de teksteditor `vi` of `vim` de overblijvende atleten volgens oplopende lengte. Bij gelijke lengte komen de zwaarste atleten eerst.

Opgave 3

In tegenstelling tot het decimaal talstelsel gebruikt het hexadecimaal talstelsel geen tien maar zestien cijfers. De cijfers `0` tot en met `9` worden verder uitgebreid met de letters `a` (waarde tien) tot en met `f` (waarde vijftien), die in deze context dus ook cijfers voorstellen. Om verwarring tussen de termen *cijfers* en *letters* te verkomen, zullen we daarom in het vervolg van deze opgave gebruik maken van de terminologie zoals die hieronder expliciet in Backus-Naur vorm gedefinieerd wordt.

```
<hexletter>           ::= a | b | c | d | e | f
<niethexletter>       ::= g | h | i | j | k | l | m | n | o | p | q | r | s |
                          t | u | v | w | x | y | z
<kleine letter>       ::= <hexletter> | <niethexletter>
<hoofdletter>         ::= A | B | C | D | E | F | G | H | I | J | K | L | M |
                          N | O | P | Q | R | S | T | U | V | W | X | Y | Z
<woord>               ::= (<hoofdletter> | <kleine letter>)+
<arabisch cijfer>     ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
<hexadecimaal cijfer> ::= <arabisch cijfer> | <hexletter>
<hexadecimaal getal>  ::= <hexadecimaal cijfer>+
```

Elke regel van het tekstbestand `hexdump.txt` start met een lijst van zestien hexadecimale getallen die elk bestaan uit twee hexadecimale cijfers. Deze hexadecimale getallen worden telkens van elkaar gescheiden door één enkele spatie. Daarna eindigt elke regel met nog een spatie, gevolgd door een woord dat tussen verticale strepen staat (`|`). Merk op dat uit voorgaande definitie volgt dat alle woorden enkel bestaan uit hoofdletters en kleine letters. Gevraagd wordt:

1. Bepaal reguliere expressies voor elk van onderstaande verzamelingen, waarbij $\mathcal{H}$ de verzameling van alle regels tekst voorstelt die voldoen aan het voorschrift van de regels uit het tekstbestand `hexdump.txt`. Probeer de reguliere expressies bovendien zo kort mogelijk te houden.

   (a) $\alpha = \{h \in \mathcal{H} \mid$ alle hexadecimale getallen bestaan uit één arabisch cijfer en één hexletter$\}$

   voorbeelden: `0b 5a 3f 5a 7d d0 5d e6 2b c4 7e 7d c2 c0 e6 9a |duivenkot|` $\in \alpha$,
   `84 bd aa 74 f3 85 da 9d ac b6 e0 b6 62 0f b5 d5 |Zuidwestkaap|` $\notin \alpha$

   (b) $\beta = \{h \in \mathcal{H} \mid$ tweede hexadecimaal cijfer van hexadecimaal getal is telkens gelijk aan eerste hexadecimaal cijfer van volgende hexadecimaal getal$\}$

   voorbeelden: `72 2c c0 00 06 66 6b bf fb bf fb ba ae e2 2d d8 |afgevreten|` $\in \beta$,
   `55 6f ae f7 80 ef 0c 01 a2 f8 b7 7d 3f 61 f8 2e |leemlagen|` $\notin \beta$

   (c) $\gamma = \{h \in \mathcal{H} \mid$ elk hexadecimaal cijfer komt twee keer voor in lijst van hexadecimale getallen$\}$

   voorbeelden: `33 e0 f1 76 9c 4f a8 6c 01 5d 45 9e 28 db 7b 2a |etalageraam|` $\in \gamma$,
   `c0 b0 f5 60 02 8b 1c a4 41 7c 53 f2 85 20 a0 d1 |topklasse|` $\notin \gamma$

   (d) $\delta = \{h \in \mathcal{H} \mid$ woord bevat geen enkele hexletter uit de lijst van hexadecimale getallen$\}$

   opmerking: ook hoofdletters die in het woord voorkomen mogen in hun kleine-letter-variant niet voorkomen als hexletter in de lijst van hexadecimale getallen

   voorbeelden: `64 52 68 23 18 08 78 43 44 11 70 72 6c 76 1c 3b |drijfriemen|` $\in \delta$,
   `70 06 61 17 7b b6 67 7d de e3 3d dc c4 45 57 77 |kromneus|` $\notin \delta$

Gebruik een commando uit de `grep` familie om enkel die regels van het bestand `hexdump.txt` te selecteren die behoren tot de opgegeven verzameling. Vermeld in je antwoordbestand voor elke verzameling het gebruikte selectiecommando, en geef telkens ook aan hoeveel regels je gevonden hebt.

2. Beschouw de verzamelingen α, β, γ en δ zoals hierboven gedefinieerd. Gebruik nu deze verzamelingen om op de volgende manier een boodschap bestaande uit vier woorden te achterhalen:

   (a) het eerste woord staat op de unieke regel met programmacode uit de verzameling $\alpha \cap \beta$

   (b) het tweede woord staat op de unieke regel met programmacode uit de verzameling $\beta \cap \gamma$

   (c) het derde woord staat op de unieke regel met programmacode uit de verzameling $\gamma \cap \delta$

   (d) het vierde woord staat op de unieke regel met programmacode uit de verzameling $\delta \cap \alpha$

   Vermeld in je antwoordbestand de gevonden woorden, samen met het Unix commando (of de commandosequentie) dat je gebruikt hebt om elk van deze woorden te vinden. Elk commando of elke commandosequentie moet dus als resultaat één van de gezochte woorden naar standaard uitvoer schrijven, zonder de hexadecimale getallen die eraan voorafgaan en de verticale strepen die errond staan.

Opgave 4

Geef LaTeX-code die precies hetzelfde resultaat oplevert als het tekstfragment in onderstaand kader. Zorg er daarbij voor dat de opmaak zo getrouw mogelijk behouden blijft. Zorg er ook voor dat formules automatisch genummerd worden, en gebruik waar mogelijk verwijzingen naar deze nummeringen. Plaats een PDF bestand met daarin het gecompileerde LaTeX-fragment in het ZIP-bestand dat je indient via Indianio.

Elke mogelijke functie f zal een goede oplossing zijn indien ze de hoofdinvariant inv_m dicht benadert in de ongelijkheid. Verschillende ongelijkheidsoperatoren kunnen opgegeven worden. Nemen we hier bijvoorbeeld $\leq$ dan moet gelden dat:

$$inv_m(G) \leq f(inv_1(G), inv_2(G), \ldots) \tag{1}$$

Een mogelijke functie f is bijvoorbeeld $f(inv_1, inv_2, inv_3, \ldots) = inv_1 + inv_2$. Dan is de vraag of voor elke graaf G geldt dat:

$$inv_m(G) \leq inv_1(G) + inv_2(G)$$

En in welke mate de rechterkant van de vergelijking de linkerkant benadert.
We nemen als fout "oneindig" indien de functie f de ongelijkheid niet respecteert, en het verschil met de invariant indien deze dat wel doet. Als je dus een graaf G en een functie van de invarianten f hebt dan is $\text{fout}(G, f)$ in ons voorbeeld gelijk aan:

$$\text{fout}(G, f) = \begin{cases} \infty & inv_m(G) \not\leq f(inv_1(G), inv_2(G), \ldots) \\ f(inv_1(G), inv_2(G), \ldots) - inv_m(G) & \text{anders} \end{cases} \tag{2}$$

Los hiervan kunnen we ook stellen dat

$$0 \leq i - (2 * h(r)|_{a_b} + 1)2^{a-a_b-1} \leq 2^{a-a_b} - 1$$
$$\Leftrightarrow i \geq (2 * h(r)|_{a_b} + 1) * 2^{a-a_b-1} \wedge i \leq 2^{a-a_b} - 1 + (2 * h(r)|_{a_b} + 1) * 2^{a-a_b-1}$$

Ook onderstaande tabel is compleet ongerelateerd aan vergelijking 1.

algoritme	slechtste geval	gemiddeld
bubblesort	$O(n^2)$	$\theta(n^2)$
quicksort	$O(n^2)$	$\theta(n \log n)$
mergesort	$O(n \log n)$	$\theta(n \log n)$