
EXAMEN: Computergebruik

1^e Bachelor Informatica
prof. dr. Peter Dawyndt
groep 2

woensdag 08-01-2014, 14:00
academiejaar 2013-2014
eerste zittijd

Opgave 1

(10 pt)

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 de commando's (tijdelijke) bestanden aanmaken binnen het bestandssysteem, tenzij dat expliciet gevraagd wordt.

1. Spelfouten zijn snel gemaakt en komen voor in verschillende vormen. Sommige spelfouten kunnen bovendien de betekenis van een zin grondig wijzigen. Voor deze opgave vragen we je voor één keer niet om spelfouten op te sporen en te verwijderen, maar net om spelfouten te introduceren in een aantal gegeven zinnen. De zinnen staan elk op een afzonderlijke regel in een gegeven tekstbestand (**spelout.txt** in onderstaand voorbeeld), en worden telkens voorafgegaan door een getal $n \in \mathbb{N}_0$ en een spatie. Vul onderstaande commandolijn aan, zodat elk zin zonder de letter op de n -de positie wordt uitgeschreven naar standaard uitvoer. Indien de zin geen n -de letter heeft, dan moeten geen letters verwijderd worden uit de zin.

```
$ cat spelout.txt
1 star wars
4 munich
5 casino royale
7 the fast and the furious
52 a fish called wanda
$ cat spelout.txt | ...
tar wars
munch
casio royale
the fat and the furious
a fish called wanda
```

2. Een **acrostichon** is een gedicht waarvan de eerste letters van iedere regel achter elkaar gelezen zelf ook een woord of zin vormen. Een bekend voorbeeld is de tekst van het Nederlandse volkslied — het Wilhelmus — waarvan de eerste letters van de coupletten in de originele spelling samen de naam Willem van Nassov vormen. Ook de beginletters van de eerste 15 zinnen van de Nederlandse troonrede van 2010 (zie tekstbestand **troonrede.txt**) vormen de naam Willem van Nassov. Gevraagd wordt:
 - (a) Geef een commando dat de eerste karakters van alle regels van een tekstbestand achter elkaar uitschrijft naar standaard uitvoer (in onderstaand voorbeeld geïllustreerd voor de bestanden **troonrede.txt** en **wakker.txt**). Als het eerste karakter van een regel geen letter is (of als de regel geen karakters bevat), dan moet dit karakter niet uitgeschreven worden.
 - (b) Schrijf een **bash** shell script **acrostichon** waaraan de padnaam van een tekstbestand als argument moet doorgegeven worden. Enkel als de eerste karakters van alle regels van het tekstbestand (uitgezonderd diegene die geen letter voorstellen) een woord vormen dat voorkomt in **/usr/share/dict/words** (op **helios**; of een andere woordenlijst waarvan de

locatie vast in het script zit ingebakken als je op een ander computersysteem zou werken), moet het script dat woord met kleine letters uitschrijven naar standaard uitvoer.

```
$ ... (commando met troonrede.txt)
WILLEMVANNASSOV
$ ... (commando met wakker.txt)
WAKKER
$ acrostichon troonrede.txt
$ acrostichon wakker.txt
wakker
```

3. Met een paar tandenstokers of lucifers kan je eenvoudige wiskundige uitdrukkingen neerschrijven. Natuurlijke getallen worden gevormd door evenveel tandenstokers (voorgesteld door `|`) verticaal naast elkaar te plaatsen. Het getal drie wordt dan bijvoorbeeld geschreven als `|||`. We kunnen ook twee tandenstokers gebruiken om een plusteken te vormen (`+`), en als we dat 45 graden draaien dan krijgen we de vermenigvuldigingsoperator (`x`). De uitdrukking `|| x |||| + |` vormt dan bijvoorbeeld een alternatieve manier om het getal negen weer te geven, gebruik makend van elf tandenstokers.



Schrijf een **bash** shell script **tandenstokers** waaraan als argument (of reeks argumenten) een wiskundige uitdrukking wordt doorgegeven, die uitgeschreven wordt aan de hand van een aantal tandenstokers. Voor het neerschrijven van deze uitdrukking wordt in stringvoorstelling enkel gebruik gemaakt van de karakters `|`, `+` en `x` (een kleine letter). Voorts kunnen er in de stringnotatie voor de duidelijkheid overal spaties geplaatst worden, die verder geen betekenis hebben. Het script moet naar standaard uitvoer één enkele regel uitschrijven met de vorm

uitdrukking = *n* (*m* tandenstokers)

Hierbij moeten de cursieve fragmenten ingevuld worden op basis van gegeven of berekende waarden. Op de plaats van *uitdrukking* moet de stringnotatie van de uitdrukking ingevuld worden, zoals die als argument werd doorgegeven. De uitdrukking moet echter opgemaakt worden met de volgende stijlregels

- vooraan en achteraan de uitdrukking staan geen spaties
- tussen opeenvolgende verticale strepen (`|`) staan geen spaties
- voor en achter elke operator (`+`, `x`) staat juist één spatie

De waarde *n* stelt de numerieke waarde van de uitdrukking voor, en de waarde *m* stelt het aantal tandenstokers voor dat nodig is om de uitdrukking te vormen. Vergeet bij deze laatste waarde ook niet de tandenstokers mee te tellen die nodig zijn om de wiskundige operatoren te vormen!

```
$ tandenstokers "| | x || | |+|"
|| x |||| + | = 9 (11 tandenstokers)
$ tandenstokers "\| \| x\\|| \| \\|+\\|
|| x |||| + | = 9 (11 tandenstokers)
```

4. Bij grote softwareprojecten willen we vaak achterhalen in welke broncodebestanden een bepaalde variabele of constante gedefinieerd en/of gebruikt wordt. Schrijf een **bash** shell script **broncode** waaraan drie argumenten moeten doorgegeven worden: *i*) de padnaam van een directory, *ii*)

een bestandextensie (het gedeelte na het laatste punt in de naam van het bestand) en *iii*) de naam van een variabele of constante. Het script moet opzoeken waar en in welke tekstbestanden regels aangetroffen worden die de naam van de variabele of de constante vermelden. Enkel de tekstbestanden in de opgegeven directory en de onderliggende subdirectories die een naam hebben die eindigt op de opgegeven extensie moeten doorzocht worden. Voor elke broncoderegel waarop de naam gevonden wordt, moet het script een regel naar standaard uitvoer schrijven die de volgende drie informatievelden bevat (van elkaar gescheiden door een dubbelpunt): *i*) de naam van het bestand waarin de broncoderegel voorkomt, *ii*) het regelnummer binnen dat bestand en *iii*) de broncoderegel zelf.

```
$ broncode /usr/lib py spam
/usr/lib/python2.5/site.py:36:with three subdirectories, foo, bar and spam, a...
/usr/lib/python2.5/site.py:56:because bar.pth comes alphabetically before foo...
/usr/lib/python2.5/cgitb.py:32:      return '''<!--: spam
$ broncode /usr/share pl foobar
/usr/share/doc/swig2.0-examples/test-suite/perl5/exception_order_runme.pl:18:...
/usr/share/doc/swig2.0-examples/perl5/multiple_inheritance/runme.pl:15:print ...
/usr/share/doc/swig2.0-examples/perl5/multiple_inheritance/runme.pl:16:$foo_B...
```

Opgave 2

(10 pt)

Het tekstbestand `stormen.txt` bevat informatie over alle tropische stormen die de afgelopen 5 jaar werden waargenomen. De eerste regel van dit bestand bevat de hoofding. Alle andere regels bevatten de volgende informatievelden van één tropische storm: *i*) jaar, *ii*) rangnummer, *iii*) naam, *iv*) periode, *v*) windsnelheid, *vi*) luchtdruk en *vii*) categorie. De informatievelden worden telkens van elkaar gescheiden door één enkele verticale streep (|). Gevraagd wordt om — gebruik makend van de teksteditoren `vi` of `vim` — een reeks 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. Alle opdrachten moeten na elkaar uitgevoerd worden. Ter controle kan je gebruik maken van de meegeleverde bestanden `stormen.i` ($1 \leq i \leq 5$), die telkens de inhoud van het bestand bevatten nadat de *i*-de opdracht werd uitgevoerd.

1. Verwijder de verticale strepen aan het begin en einde van elke regel en vervang alle overblijvende verticale strepen door puntkomma's. Eventuele spaties rond de verticale strepen moeten ook verwijderd worden. Verwijder ook de hoofdingsregel. Bijvoorbeeld:

```
1 | Year | # | Name | Date | Wind | Pres | Cat |
2 | 2013 | 8 | Tropical Depression EIGHT | 06-07 SEP | 30 | | - |
3 | 2013 | 9 | Hurricane-1 HUMBERTO | 08-19 SEP | 75 | 982 | 1 |
4 | 2013 | 11 | Tropical Storm JERRY | 29 SEP-03 OCT | 45 | 1005 | - |
```

wordt omgezet naar

```
1 | 2013;8;Tropical Depression EIGHT;06-07 SEP;30;;-
2 | 2013;9;Hurricane-1 HUMBERTO;08-19 SEP;75;982;1
3 | 2013;11;Tropical Storm JERRY;29 SEP-03 OCT;45;1005;-
```

2. In het veld dat de categorie van de storm vermeldt, staat enkel informatie ingevuld als het gaat om een orkaan (*Hurricane*). In het geval van tropische stormen (*Tropical Storm*) of tropische depressies (*Tropical Depression*) staat er enkel een koppelteken (-) in het veld. Pas het bestand aan zodat in plaats van het streepje respectievelijk TS of TD komt te staan op basis van de informatie in het veld met de naam van de storm. Toegepast op het vorige voorbeeld wordt dit

```
1 | 2013;8;Tropical Depression EIGHT;06-07 SEP;30;;TD
2 | 2013;9;Hurricane-1 HUMBERTO;08-19 SEP;75;982;1
3 | 2013;11;Tropical Storm JERRY;29 SEP-03 OCT;45;1005;TS
```

3. Wijzig de volgorde van de velden naar: *i*) periode, *ii*) jaar, *iii*) categorie, *iv*) naam, *v*) windsnelheid en *vi*) luchtdruk. Het veld met het rangnummer moet dus weggelaten worden. Toegepast op het vorige voorbeeld wordt dit

```

1 | 06-07 SEP;2013;TD;Tropical Depression EIGHT;30;
2 | 08-19 SEP;2013;1;Hurricane-1 HUMBERTO;75;982
3 | 29 SEP-03 OCT;2013;TS;Tropical Storm JERRY;45;1005

```

4. Laat in het veld met de naam telkens de classificatie (*Tropical Storm*, *Tropical Depression*, *Hurricane-1* of *Hurricane-2*, ...) weg. Schrijf van het overgebleven deel **enkel** de eerste letter in hoofdletters, en zet de rest om naar kleine letters. Toegepast op het vorige voorbeeld wordt dit

```

1 | 06-07 SEP;2013;TD;Eight;30;
2 | 08-19 SEP;2013;1;Humberto;75;982
3 | 29 SEP-03 OCT;2013;TS;Jerry;45;1005

```

5. Vervang de twee velden met de periode en het jaar waarin de storm werd waargenomen door twee nieuwe velden die de volledige startdatum en einddatum van de waarneming vermelden. Het formaat van de nieuwe velden is DD MMM YYYY, waarbij DD het volgnummer van de dag van de maand is, MMM de drie-letterafkorting van de maand is en YYYY het jaar (vier cijfers) is. Haal deze informatie uit de oorspronkelijke velden met de periode en het jaar waarin de storm werd waargenomen. Toegepast op het vorige voorbeeld wordt dit

```

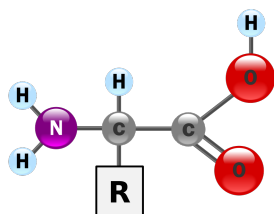
1 | 06 SEP 2013;07 SEP 2013;TD;Eight;30;
2 | 08 SEP 2013;19 SEP 2013;1;Humberto;75;982
3 | 29 SEP 2013;03 OCT 2013;TS;Jerry;45;1005

```

Opgave 3

(10 pt)

Menselijke eiwitten worden opgebouwd uit 20 verschillende aminozuren. Dit zijn organische verbindingen die allemaal dezelfde generische chemische structuur hebben die hieronder staat weergegeven. De zijketen *R* is voor elk van de aminozuren verschillend en de polariteit van deze zijketen bepaalt in grote mate de oplosbaarheid van de aminozuren. De gangbare indeling van de aminozuren in zes grote groepen wordt weergegeven in onderstaande tabel. Deze indeling gebeurt op basis van de algemene structuur van de aminozuren en de chemische eigenschappen van hun *R*-zijketens. Elk aminozuur wordt doorgaans aangeduid met een hoofdletter (hierbij worden alle letters van het alfabet gebruikt, behalve B, J, O, U, X en Z). In de tabel hebben we dan ook zowel de namen als de corresponderende hoofdletters van de aminozuren opgenomen.



KLASSE	NAAM VAN DE AMINOZUREN	LETTERS
alifatisch	glycine, alanine, valine, leucine, isoleucine	G, A, L, V, I
hydroxyl	serine, cysteïne, threonine, methionine	S, C, T, M
cyclisch	proline	P
aromatisch	fenylalanine, tyrosine, tryptofaan	F, Y, W
basisch	histidine, lysine, arginine	H, K, R
zuur	asparaginezuur, glutaminezuur, asparagine, glutamine	D, E, N, Q

Elke regel van het tekstbestand **eiwitten.txt** bevat de stringvoorstelling van een eiwitsequentie, die bestaat uit een reeks hoofdletters die de aminozuren voorstellen. Daarna volgt telkens één enkele spatie en een woord dat enkel uit hoofdletters bestaat. Gevraagd wordt:

- Bepaal reguliere expressies voor elk van onderstaande verzamelingen. Daarbij staat \mathcal{P} voor de verzameling van alle eiwitsequenties. Probeer de reguliere expressies bovendien zo kort mogelijk te houden.

(a) $\alpha = \{p \in \mathcal{P} \mid \text{in } p \text{ staat na elk alifatisch aminozuur een basisch aminozuur}\}$

voorbeelden: $\text{DGHEHVMVHQHGRHDQSLHNR} \in \alpha$,

$\text{CLMNI\textbf{MR}NKENKYTRCNDW\textbf{IN}WNTMQ} \notin \alpha$

- (b) $\beta = \{p \in \mathcal{P} \mid \text{als er in } p \text{ tussen twee zure aminozuren één ander aminozuur staat, dan moet dit een basisch aminozuur zijn} \}$

voorbeelden: $\text{ALMQRNTVKYHCYFNHNRGTMDRQFYAK} \in \beta$,
 $\text{LLVRKFDIDHHQFSVLWDHDQEHAIAKIKCVQKNIIYVMPM} \notin \beta$

- (c) $\gamma = \{p \in \mathcal{P} \mid \text{in } p \text{ staat voor en na elk aminozuur met een hydroxylgroep altijd hetzelfde aromatisch aminozuur} \}$

voorbeelden: $\text{HRVRAHGHD FN IYMYIKQHFLFSFNKWWNEIWTWVRHGKHIKFARRGFCFHFYE} \in \gamma$,
 $\text{EDARAIDPKRSMRIHNKNRWCGYRIHTQALFLKYAFMKRFWVGI} \notin \gamma$

- (d) $\delta = \{p \in \mathcal{P} \mid p \text{ bevat minstens één triplet dat meer dan één keer voorkomt}^1 \}$

¹ de herhaalde tripletten mogen elkaar ook gedeeltelijk overlappen

voorbeelden: $\text{EDGFYSFPCPQILQDVGEINAFCHLNMKEEKSRYVFPCRAVSECREFFKKLCKEWMG} \in \delta$,
 $\text{GCYTFTFQTKALDDNKH MNVFCIFQDCFH YFEFRSMADVYDRANAINCLDPDFDSV} \notin \delta$

Gebruik een commando uit de **grep** familie om enkel die regels van het bestand **eiwitten.txt** te selecteren, waarvan de eiwitsequentie behoort 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:

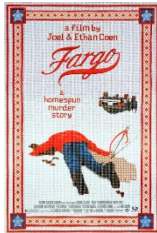
- (a) het eerste woord staat op de unieke regel met een eiwitsequentie uit de verzameling $\alpha \cap \beta$
- (b) het tweede woord staat op de unieke regel met een eiwitsequentie uit de verzameling $\beta \cap \gamma$
- (c) het derde woord staat op de unieke regel met een eiwitsequentie uit de verzameling $\gamma \cap \delta$
- (d) het vierde woord staat op de unieke regel met een eiwitsequentie 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.

Opgave 4

(6 pt)

Het doel van deze opgave is om een **bash** shell script te schrijven dat automatisch informatie over een gegeven film ophaalt uit de Open Movie Database (<http://www.omdbapi.com/>) en deze in een propere opmaak weergeeft in een PDF document. Voor de film Fargo moet het PDF bestand dan bijvoorbeeld de volgende informatie weergeven in de aangegeven opmaak.

<div>Fargo</div> <div>  </div>	
1996	William H. Macy, Steve Buscemi, Peter Stormare, Kristin Rudrüd
Jerry Lundegaard's inept crime falls apart due to his and his henchmen's bungling and the persistent police work of the quite pregnant Marge Gunderson.	

Hiervoor ga je als volgt te werk:

1. Maak een $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ -bestand `template_film.tex` dat gebruikt kan worden als template om informatie over een film weer te geven in tabelvorm. Voor elk informatieveld bevat de template een plaatshouder die achteraf kan vervangen worden door de eigenlijke inhoud van het veld. Plaatshouders worden aangeduid door een apestaartje (@) gevolgd door de naam van een informatieveld (namen bestaan enkel uit letters). De volgende informatie over een film zit vervat in de template: *i*) de titel van de film (@`titel`), *ii*) de locatie van een afbeelding van de poster (@`poster`), *iii*) een korte omschrijving van de plot (@`plot`), *iv*) het jaar waarin de film werd uitgebracht (@`jaar`) en *v*) een lijst van de belangrijkste acteurs die in de film meespelen (@`acteurs`). Na compilatie van het $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ -bestand moet de informatie in een tabel met de volgende opmaak weergegeven worden (voor deze weergave hebben we @`poster` reeds vervangen door de locatie van een afbeeldingsbestand). Zorg er voor dat je template de opmaak van de tabel zo getrouw mogelijk weergeeft.

@ <code>titel</code>	
	
@ <code>jaar</code>	@ <code>acteurs</code>
@ <code>plot</code>	

2. Schrijf een `bash` shell script `vervang` dat alle tekstregels die binnenkomen via standaard invoer terug uitschrijft naar standaard uitvoer. Hierbij moeten alle voorkomens van @`titel` in de invoer vervangen worden door het eerste argument dat aan het script wordt doorgegeven, en analoog ook alle voorkomens van respectievelijk @`poster`, @`plot`, @`jaar` en @`acteurs` door de volgende vier argumenten die aan het script doorgegeven worden. Het script moet dus bijvoorbeeld op de volgende manier kunnen gebruikt worden.

```
$ cat template_film.txt
titel: @titel
poster: @poster
plot: @plot
jaar: @jaar
acteurs: @acteurs
$ vervang aaa bbb ccc ddd eee < template_film.txt
titel: aaa
poster: bbb
plot: ccc
jaar: ddd
acteurs: eee
```

3. Schrijf een `bash` shell script `film` waaraan de naam van een film als argument moet doorgegeven worden. Het script moet in de eerste plaats de informatie over de opgegeven film opvragen via de restful API van de Open Movie Database. Hiervoor moet het script gebruik maken van de volgende URL template:

`http://www.omdbapi.com?t=titel`

Als je hierin *titel* vervangt door de titel van een film, dan krijg je een JSON-bestand terug in het volgende formaat (hier weergegeven voor de film *Fargo*).

```
1 {
2   "Title": "Fargo",
3   "Year": "1996",
4   "Rated": "R",
5   "Released": "05 Apr 1996",
```

```

6   "Runtime":"98 min",
7   "Genre":"Crime, Drama, Thriller",
8   "Director":"Joel Coen, Ethan Coen",
9   "Writer":"Ethan Coen, Joel Coen",
10  "Actors":"William H. Macy, Steve Buscemi, Peter Stormare, Kristin Rudrüd",
11  "Plot":"Jerry Lundegaard's inept crime falls apart due to his and his...",
12  "Language":"N/A",
13  "Country":"N/A",
14  "Awards":"N/A",
15  "Poster":"http://ia.media-imdb.com/images/M/MV5BMTgxNzY3M._V1_SX178_.jpg",
16  "Metascore":"85",
17  "imdbRating":"8.2",
18  "imdbVotes":"294,567",
19  "imdbID":"tt0116282",
20  "Type":"movie",
21  "Response":"True"
22 }

```

Het JSON-bestand dat door de Open Movie Database webservice wordt teruggegeven, bevat steeds de informatie van één enkele film. Deze informatie wordt weergegeven als een verzameling naam-waarde paren. De elementen van de verzameling zitten ingesloten tussen accolades, en worden van elkaar gescheiden door een komma. Zowel de naam als de waarde staan tussen dubbele aanhalingstekens, en worden van elkaar gescheiden door een dubbelpunt (:). Buiten de inhoud van de naam-waarde paren kan in het JSON-formaat naar willekeur witruimte toegevoegd worden (we hebben in bovenstaand voorbeeld dan ook de naam-waarde paren iets overzichtelijker weergegeven; de webservice geeft het volledige JSON-bestand op één enkele regel terug). Het script moet de volgende informatie over een film uit het JSON-bestand halen: de titel (**Title**), de locatie van een afbeelding van de filmposter (**Poster**), de korte omschrijving van de plot (**Plot**), het jaar waarin de film werd uitgebracht (**Year**) en een lijst van de belangrijkste acteurs die in de film meespelen (**Actors**). De locatie waarop een afbeelding van de filmposter te vinden is, wordt weergegeven als een absolute URL.

Het script moet daarna een L^AT_EX-bestand aanmaken door het **bash** shell script **vervang** uit te voeren om de informatie die uit het JSON-bestand geëxtraheerd werd op de juiste plaatsen in te vullen in het L^AT_EX-bestand **template.film.tex**. Dit L^AT_EX-bestand moet vervolgens gecompileerd worden tot een PDF document. Enkel dit laatste PDF document mag in de huidige directory opgeslaan worden onder de naam **titel.pdf**, waarbij **titel** de titel van de film is die aan het shell script **film** werd doorgegeven. Alle andere bestanden die tijdens het uitvoeren van het script aangemaakt worden, moeten opgeslaan worden in de directory **tempdir** die hiervoor tijdelijk wordt aangemaakt in de huidige directory. Na het uitvoeren van het shell script moet deze tijdelijke directory met zijn volledige inhoud terug verwijderd zijn. Zorg er ook voor dat het shell script geen enkele uitvoer uitschrijft naar het standaard uitvoerkanaal en naar het standaard errorkanaal.

Belangrijk: Genereer de PDF bestanden **fargo.pdf** en **batman returns.pdf** door het shell script **film** als volgt aan te roepen. Plaats deze PDF bestanden in het ZIP-bestand dat je indient via Indianio.

```

$ film fargo
$ film "batman returns"

```