EXAMEN: **Computergebruik**

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

woensdag 20-01-2010, 8:30
academiejaar 2009-2010
eerste zittijd

Opgave 1

Het bestand `padnamen.txt` bevat een lijst van padnamen naar bestanden in een unix directorystructuur. Hierbij staat elke padnaam op een afzonderlijke regel, en worden padnamen opgebouwd volgens het formaat

```
[/]<directorynaam_1>/<directorynaam_2>/.../<directorynaam_n>/<bestandsnaam>
```

Een padnaam bestaat hierbij uit n directorynamen, waarbij de slash vooraan optioneel is (om onderscheid te maken tussen absolute en relatieve padnamen). Directorynamen bestaan uitsluitend uit cijfers en kleine letters en bestandsnamen bestaan uitsluitend uit kleine letters. Gevraagd wordt:

1. Bepaal reguliere expressies voor elk van de onderstaande verzamelingen, waarbij $\mathcal{P}$ de verzameling van alle geldige padnamen voorstelt in het formaat dat hierboven werd beschreven. Probeer deze reguliere expressies zo kort mogelijk te houden.

   (a) $\alpha = \{p \in \mathcal{P} \mid$ alle directorynamen van p zijn even lang en bevatten 3, 4 of 5 karakters$\}$

   (b) $\beta = \{p \in \mathcal{P} \mid p$ bestaat uit drie directorynamen, waarbij het laatste karakter van elke directorynaam gelijk is aan het eerste karakter van de volgende directory- of bestandsnaam$\}$
   voorbeeld: `/abc/cx2w4/456w/woord` $\in \beta$

   (c) $\gamma = \{p \in \mathcal{P} \mid$ geen enkele directorynaam van p is juist 4 karakters lang$\}$

   (d) $\delta = \{p \in \mathcal{P} \mid$ elke directorynaam van p bestaat enkel uit letters of enkel uit cijfers$\}$
   voorbeelden: `/abc/1234/def/567/woord` $\in \delta$, `\abc/1t34/d9f/567/woord` $\notin \delta$

   Gebruik een commando uit de `grep` familie om enkel die regels van het bestand `padnamen.txt` te selecteren die padnamen bevatten die behoren tot de opgegeven verzameling. Vermeld in je antwoordbestand voor elke verzameling het gebruikte selectiecommando, en geef 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) eerste woord is bestandsnaam op unieke regel met padnaam uit verzameling $\alpha \cap \beta$

   (b) tweede woord is bestandsnaam op unieke regel met padnaam uit verzameling $\beta \cap \gamma$

   (c) derde woord is bestandsnaam op unieke regel met padnaam uit verzameling $\gamma \cap \delta$

   (d) vierde woord is bestandsnaam op unieke regel met padnaam uit verzameling $\delta \cap \alpha$

   Vermeld in je antwoordbestand de gevonden woorden, samen met het unix commando (of de commandosequentie) dat je gebruikt hebt om elke van deze woorden te vinden. Elk commando of elke commandosequentie moet dus als resultaat één van de gezochte woorden naar standaard uitvoer schrijven (niet de volledige padnaam waar dit woord deel van uitmaakt).

Opgave 2

1. Geef LaTeX-code die een reconstructie maakt van onderstaande tabel. Zorg er daarbij voor dat de opmaak zo getrouw mogelijk behouden blijft.

De tabel

	£	€	$
£	1	**0,89**	0,62
€	**1,13**	**1**	**0,70**
$	1,62	**1,43**	1

vergelijkt enkele belangrijke munteenheden.

2. Geef LaTeX-code die precies hetzelfde resultaat oplevert als het tekstfragment in onderstaand kader. Zorg er daarbij voor dat formules en eigen omgevingen automatisch genummerd worden, en gebruik waar mogelijk verwijzingen naar deze nummeringen.

We leiden enkele functies af volgens de volgende recursieve relaties:

$$\eta_1(Z) = [\xi(Z) - \eta_0(Z)]/Z, \tag{1}$$
$$\eta_m(Z) = [\eta_{m-2}(Z) - (2m-1)\eta_{m-1}(Z)]/Z, \quad m = 2, 3, \ldots. \tag{2}$$

Merk op dat elke η_m functie een lineaire combinatie is van de reference propagators $\bar{u} = \xi$ en $\bar{v} = \delta\eta_0$. De functies $\xi(Z), \eta_0(Z), \eta_1(Z), \ldots$ werden al beschreven in een vorig artikel (en daar voorgesteld als $\bar{\xi}, \bar{\eta}_0, \bar{\eta}_1, \ldots$). Ze voldoen aan een aantal basiseigenschappen die we hieronder kort samenvatten.

`Eigenschap` **1 (Serie expansie)**

$$\eta_m(Z) = 2^m \sum_{q=0}^{\infty} \frac{g_{mq} Z^q}{(2q+2m+1)!},$$

met

$$g_{mq} = \begin{cases} 1 & \text{if } m = 0, \\ (q+1)(q+2)\ldots(q+m) & \text{if } m > 0. \end{cases}$$

In het bijzonder

$$\eta_m(0) = \frac{1}{(2m+1)!!} = \frac{1}{(2m+1)(2m-1)(2m-3)\ldots 1}.$$

`Eigenschap` **2 (Differentiatie met betrekking tot Z)**

$$\xi'(Z) = \frac{1}{2}\eta_0(Z), \quad \eta_m'(Z) = \frac{1}{2}\eta_{m+1}(Z), \quad m = 0, 1, 2, \ldots. \tag{3}$$

Waarom hebben de formules in Eigenschap 1 geen nummer? Ik verkies genummerde formules zoals bij Eigenschap 2.

Plaats een PDF bestand met daarin de gecompileerde LaTeX fragmenten in het ZIP-bestand dat je indient via Indianio.

Opgave 3

Gegeven is een tekstbestand `vulkanen.csv`, waarvan de regels de volgende vorm hebben:

```
<nummer><naam><plaats><datering><breedtegraad><NZ-breedte><lengtegraad><OW-lengte><hoogte><type>
```

Elke regel bevat gegevens over een bepaalde vulkaan. Het bestand is opgeslagen in CSV-formaat, m.a.w. de velden worden van elkaar gescheiden door puntkomma's. Het bestand bevat bovendien een hoofdingsregel en een voorbeeldregel die beginnen met een hekje (`#`). Gevraagd wordt om, gebruik makend van de teksteditor `vi` (of `vim`), een reeks substitutiecommando's op te stellen die achtereenvolgens de volgende opdrachten uitvoeren:

1. Vervang het huidige veldscheidingsteken door een tabteken.

2. Sommige vulkaannummers eindigen niet op een cijfer. Vang dit op door het laatste teken van die nummers te vervangen door een nul (`0`). Zo moet de regel

   ```
   0101-01= Campi Flegrei Italy Historical 40,827 N 14,14 E 458 Caldera
   ```

   worden vervangen door de regel

   ```
   0101-010 Campi Flegrei Italy Historical 40,827 N 14,14 E 458 Caldera
   ```

3. Zet de coördinaten van elke vulkaan om naar het formaat `EW/lengtegraad-NS/breedtegraad` en verwijder de kolom met de plaatsnaam. Zo moet de regel

   ```
   0101-010 Campi Flegrei Italy Historical 40,827 N 14,14 E 458 Caldera
   ```

   worden omgezet naar

   ```
   0101-010 Campi Flegrei Historical E/14,14-N/40,827 458 Caldera
   ```

4. Ga op zoek naar alle vulkanen waarvan de naam met dezelfde letter begint als het type, en duid deze aan door voor het type een sterretje te plaatsen, bv.:

   ```
   0101-040 Stromboli Historical E/15,21-N/38,789 924 *Stratovolcano
   ```

5. Markeer alle vulkanen die onder water zitten (hoogte < 0) door achteraan deze regels een extra veld toe te voegen met als mededeling '`LOW`'. Bijvoorbeeld:

   ```
   0101-070 Campi Flegrei Mar Sicilia Historical E/12,70-N/37,1 -8 Submarine volcano LOW
   ```

6. Herorganiseer het bestand zodanig dat de regels alfabetisch worden gerangschikt op dateringsmethode, en waarbij elke kleine letter wordt omgezet in een hoofdletter, bv.:

   ```
   0101-010 CAMPI FLEGREI HISTORICAL E/14,14-N/40,827 458 *CALDERA
   ```

Probeer voor elke opdracht zo weinig mogelijk commando's te gebruiken en zorg ervoor dat elk van deze commando's bestaat uit zo weinig mogelijk tekens. De commentaarregels mogen door je commando's niet gewijzigd worden, zelfs niet als er bijkomende voorbeeldregels aan het bestand worden toegevoegd. Alle wijzigingen moeten na elkaar uitgevoerd worden.

Opgave 4

1. *Pluggable Authentication Modules* (kortweg `PAM`) vormen een mechanisme om verschillende authenticaschema's op systeemniveau op een hoger niveau te integreren tot een gemeenschappelijke programmeerinterface voor het ontwikkelen van toepassingen. Dit laat toe om programma's te schrijven die gebruik willen maken van authenticatie, op een manier die onafhankelijk is van het onderliggende authenticatieschema. PAM is gebaseerd op een reeks bibliotheekmodules, waarvan sommige afhangen van het configuratiebestand `/etc/pam.conf`. Gevraagd wordt om

onderstaande pipelines aan te vullen zodat de corresponderende uitvoer gegenereerd wordt. Bestudeer hiervoor zelf de inhoud van het tekstbestand `/etc/pam.conf` (een kopie van dit bestand werd meegeleverd als hulpbestand, voor het geval je computersysteem geen gebruik maakt van dit configuratiebestand).

(a)
```
$ cat /etc/pam.conf | ...
   4 binding
  27 required
   7 requisite
   2 sufficient
```

(b)
```
$ cat /etc/pam.conf | ...
  27 required
   7 requisite
   4 binding
   2 sufficient
```

2. Gegeven is een `perl` script dat de cijfers $1, 2, \ldots, 9$ op afzonderlijke regels uitschrijft naar standaard uitvoer. Gevraagd wordt om de onderstaande pipelines op een correcte manier aan te vullen zodat de uitvoer van het `perl` script wordt omgevormd tot het formaat zoals aangegeven. Je moet bij elk van deze drie pipelines tekstregels samenvoegen in kolommen. Merk ook op dat $7 = 1 + 2 \times 3$ en dat $120 = 7 + 34 + 79$.

(a)
```
$ perl -e 'for ($i=1;$i<10;$i++) { print "$i\n"; }' | ...
1    2    3
4    5    6
7    8    9
```

(b)
```
$ perl -e 'for ($i=1;$i<10;$i++) { print "$i\n"; }' | ...
7
34
79
```

(c)
```
$ perl -e 'for ($i=1;$i<10;$i++) { print "$i\n"; }' | ...
120
```

Opgave 5

De *Markup Validator* (`validator.w3.org`) is een online dienstverlening die gratis wordt aangeboden door het `W3C` om de geldigheid van webdocumenten te helpen controleren. De meeste webdocumenten maken immers gebruik van markeertalen zoals `HTML` of `XHTML`. Omdat dergelijke markeertalen worden vastgelegd in technische specificaties die meestal formele grammaticaregels (en een vocabularium) bevatten, kunnen deze ook door machines geïnterpreteerd worden. Het proces waarbij de opbouw van een document gecontroleerd wordt ten opzichte van een stel grammaticaregels wordt *validatie* genoemd, en dat is precies wat de Markup Validator op een geautomatiseerde manier uitvoert.
De Markup Validator laat echter alleen maar toe om individuele webdocumenten te valideren. Vaak wordt echter gevraagd om de geldigheid van een volledige lijst van webdocumenten te controleren. Je opdracht bestaat erin om ook dit proces volledig te automatiseren. Hiervoor ga je als volgt te werk:

1. Schrijf een `bash` shell script `validate_html` dat aangeeft of een `HTML` of `XHTML` webdocument — dat online beschikbaar is — geldig is of niet. De URL van het webdocument moet als parameter aan het shell script doorgegeven worden. Op basis daarvan moet het shell script de Markup

Validator aanroepen via de web service

```
http://validator.w3.org/check?uri=<URL>
```

Binnen deze template-URL vul je op de plaats van `<URL>` de URL van een gegeven online webdocument in. De Markup Validator geeft dan als resultaat een XHTML bestand terug, waaruit het shell script de inhoud van het `h2` element moet filteren en als boodschap moet uitschrijven naar standaard uitvoer. Gebruik hierbij `ed` of `ex` om binnen het shell script de nodige bestandsmanipulaties uit te voeren op het bestand dat door de web service teruggegeven wordt. Je mag er hierbij van uitgaan dat het bestand dat door de web service wordt teruggegeven slechts één enkel `h2` element bevat. Houd er echter wel rekening mee dat de inhoud van dit `h2` element over verschillende regels kan gesplitst zijn. Onderstaande sessie illustreert het gebruik van het shell script `validate_html`.

```
$ validate_html www.w3c.org
This document was successfully checked as XHTML 1.0 Strict!

$ validate_html www.ugent.be
Errors found while checking this document as XHTML 1.0 Transitional!

$
```

2. Gebruik het shell script `validate_html` om een `bash` shell script `validate_multiple_html` te schrijven waarmee een lijst van URLs kan gevalideerd worden. Aan dit shell script moet een bestandsnaam als argument meegegeven worden. Dit bestand bevat een lijst van URLs, waarbij elke URL op een afzonderlijke regel staat. Om elke URL uit de lijst te valideren, moet het shell script als volgt te werk gaan:

   (a) kopieer de lijst van URLs naar een tijdelijk bestand

   (b) vorm het tijdelijk bestand om naar een shell script waarin het validatiescript afzonderlijk wordt aangeroepen voor elk van de opgelijste URLs

   (c) voer het omgevormde tijdelijk bestand uit als shell script

   (d) verwijder het tijdelijk bestand

   Zorg er hierbij voor dat het shell script `validate_multiple_html` het commando `ed` of `ex` gebruikt om de nodige bestandsmanipulaties uit te voeren in de stappen die hierboven beschreven worden. Onderstaande sessie illustreert het gebruik van shell script `validate_multiple_html`.

```
$ cat html_test_files
www.w3c.org
www.ugent.be
www.google.com
www.vrtnieuws.net
www.xxx.yyy

$ validate_multiple_html html_test_files
www.w3c.org: This document was successfully checked as XHTML 1.0 Strict!
www.ugent.be: Errors found while checking this document as XHTML 1.0 Transitional!
www.google.com: Errors found while checking this document as HTML5!
www.vrtnieuws.net: Errors found while checking this document as XHTML 1.0 Transitional!
www.xxx.yyy: Sorry! This document can not be checked.

$
```

   Het testbestand `html_test_files` werd ook als hulpbestand meegeleverd met het examen.

EXAMEN: **Computergebruik**

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

woensdag 20-01-2010, 14:00
academiejaar 2009-2010
eerste zittijd

Opgave 1

Het bestand `padnamen.txt` bevat een lijst van padnamen naar bestanden in een unix directorystructuur. Hierbij staat elke padnaam op een afzonderlijke regel, en worden padnamen opgebouwd volgens het formaat

```
[/]<directorynaam_1>/<directorynaam_2>/.../<directorynaam_n>/<bestandsnaam>
```

Een padnaam bestaat hierbij uit n directorynamen, waarbij de slash vooraan optioneel is (om onderscheid te maken tussen absolute en relatieve padnamen). Directorynamen bestaan uitsluitend uit cijfers en kleine letters en bestandsnamen bestaan uitsluitend uit kleine letters. Gevraagd wordt:

1. Bepaal reguliere expressies voor elk van de onderstaande verzamelingen, waarbij $\mathcal{P}$ de verzameling van alle geldige padnamen voorstelt in het formaat dat hierboven werd beschreven. Probeer deze reguliere expressies zo kort mogelijk te houden.

   (a) $\alpha = \{p \in \mathcal{P} \mid p$ is relatieve padnaam, waarbij elke directorynaam exact twee cijfers bevat$\}$
   voorbeeld: `bo28/ze2zti9/drl1au9f/woord` $\in \alpha$

   (b) $\beta = \{p \in \mathcal{P} \mid$ er bestaat een paar opeenvolgende karakters dat in minstens drie opeenvolgende directory- of bestandsnamen voorkomt binnen $p\}$
   voorbeeld: `/abc/d4o47wip/9d4/oud4v/woord` $\in \beta$

   (c) $\gamma = \{p \in \mathcal{P} \mid$ elke directorynaam van p is alternerende reeks van cijfers en letters$\}$
   voorbeeld: `/8o4/b9z2/f7i2/woord` $\in \gamma$

   (d) $\delta = \{p \in \mathcal{P} \mid$ laatste karakter van elke directorynaam in p is verschillend van eerste karakter van volgende directory- of bestandsnaam$\}$

   Gebruik een commando uit de `grep` familie om enkel die regels van het bestand `padnamen.txt` te selecteren die padnamen bevatten die behoren tot de opgegeven verzameling. Vermeld in je antwoordbestand voor elke verzameling het gebruikte selectiecommando, en geef 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) eerste woord is bestandsnaam op unieke regel met padnaam uit verzameling $\alpha \cap \beta$

   (b) tweede woord is bestandsnaam op unieke regel met padnaam uit verzameling $\beta \cap \gamma$

   (c) derde woord is bestandsnaam op unieke regel met padnaam uit verzameling $\gamma \cap \delta$

   (d) vierde woord is bestandsnaam op unieke regel met padnaam uit verzameling $\delta \cap \alpha$

   Vermeld in je antwoordbestand de gevonden woorden, samen met het unix commando (of de commandosequentie) dat je gebruikt hebt om elke van deze woorden te vinden. Elk commando of elke commandosequentie moet dus als resultaat één van de gezochte woorden naar standaard uitvoer schrijven (niet de volledige padnaam waar dit woord deel van uitmaakt).

Opgave 2

1. Geef LaTeX-code die een reconstructie maakt van onderstaande tabel. Zorg er daarbij voor dat de opmaak zo getrouw mogelijk behouden blijft.

Het tabelletje

	£	€	$
£	10	8,9	6,2
€	11,3	10	7,0
$	16,2	14,3	10

zou nuttig kunnen zijn voor mensen die graag reizen.

2. Geef LaTeX-code die precies hetzelfde resultaat oplevert als het tekstfragment in onderstaand kader. Zorg er daarbij voor dat formules en eigen omgevingen automatisch genummerd worden, en gebruik waar mogelijk verwijzingen naar deze nummeringen.

Definitie 1 (Tralie $\mathcal{L}^I$) *De $\mathcal{L}^I$ tralie is gedefinieerd door*

$$\begin{aligned} L^I &= \{[x_1, x_2] \mid (x_1, x_2) \in [0,1]^2 \text{ and } x_1 \leq x_2\} \\ [x_1, x_2] \sqcap [y_1, y_2] &= [\min(x_1, y_1), \min(x_2, y_2)] \\ [x_1, x_2] \sqcup [y_1, y_2] &= [\max(x_1, y_1), \max(x_2, y_2)] \end{aligned}$$

Concept 2 *De componentgewijze uitbreiding van $\leq$ wordt gegeven door*

$$[x_1, x_2] \leq_{L^I} [y_1, y_2] \iff x_1 \leq y_1 \text{ and } x_2 \leq y_2.$$

Men heeft aangetoond dat er geen MTL-algebra bestaat voor de tralie $\mathcal{L}^I = (L^I, \sqcap, \sqcup)$, gedefineerd in Def. 1 en wiens partiële ordening $\leq_{L^I}$ gegeven wordt door de componentsgewijze uitbreiding van $\leq$ (zie Concept 2).

Het testsysteem leest

$$\begin{bmatrix} y_1'' \\ y_2'' \\ y_3'' \end{bmatrix} = \begin{bmatrix} 3-2x-E & -x & 1+x \\ -x & -1-2x-E & 1-x \\ 1+x & 1-x & 1-2x-E \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}. \tag{1}$$

De exacte oplossing is

$$y_1(x) = (1+x)e^x, \quad y_2(x) = (1-x)e^x, \quad y_3(x) = xe^x.$$

$$\Delta\tilde{y}_i(10) = \left| \frac{\tilde{y}_i(10) - y_i(10)}{y_i(10)} \right| \tag{2}$$

Plaats een PDF bestand met daarin de gecompileerde LaTeX fragmenten in het ZIP-bestand dat je indient via Indianio.

Opgave 3

Gegeven is een bestand `impactfactoren.txt`, waarvan de regels de volgende vorm hebben:

```
<naam><ISSN><# citaties><impactfactor><5-jarige impactfactor><immediacy index><# artikels>
<half-life citatie><eigenfactor score><article influence score><nummer>
```

Elke regel bevat gegevens over een bepaald wetenschappelijk tijdschrift. Het bestand bevat een hoofdingsregel die begint met een hekje (`#`). Gevraagd wordt om, gebruik makend van de teksteditor `vi` (of `vim`), een reeks substitutiecommando's op te stellen die achtereenvolgens de volgende opdrachten uitvoeren:

1. Vervang het huidige veldscheidingsteken door een tabteken.

2. De naam van enkele tijdschriften begint met de afkorting `IEEE T`, wat voluit geschreven staat voor `IEEE TRANSACTIONS`. Vul deze tijdschriftnamen op deze manier aan. Zo moet bijvoorbeeld de regel

   ```
   IEEE T PATTERN ANAL 0162-8828 24674 5,9 7,981 0,669 181 9,0 0,04964 2,617 2
   ```

   worden vervangen door

   ```
   IEEE TRANSACTIONS PATTERN ANAL 0162-8828 24674 5,9 7,981 0,669 181 9,0 0,04964 2,617 2
   ```

3. Verwijder de *immediacy index*, het aantal artikels, de *half-life citatie*, de *eigenfactor score* en de *article influence* kolommen, en wissel de *impactfactor* en de *5-jarige impactfactor* kolommen om. Zo moet de regel

   ```
   IEEE TRANSACTIONS PATTERN ANAL 0162-8828 24674 5,9 7,981 0,669 181 9,0 0,04964 2,617 2
   ```

   worden omgezet naar

   ```
   IEEE TRANSACTIONS PATTERN ANAL 0162-8828 24674 7,981 5,9 2
   ```

4. Vervang de huidige getalnotatie naar een wetenschappelijk formaat waarin een komma wordt vervangen door een punt, en kap af zodat er telkens maar 1 cijfer na de komma komt te staan, bijvoorbeeld:

   ```
   IEEE TRANSACTIONS PATTERN ANAL 0162-8828 24674 7.9 5.9 2
   ```

5. Markeer de regels waarbij de impactfactoren dezelfde grootteorde (eenheid of tiental) hebben door achteraan deze regels een extra veld toe te voegen met als mededeling '`ORDE`'. Bijvoorbeeld:

   ```
   EVOL COMPUT 1063-6560 1816 3.8 3.0 14 ORDE
   ```

6. Herorganiseer het bestand zodat je twee lijsten krijgt, één met de impactfactoren en één met de 5-jarige impactfactoren. Met andere woorden, elke van deze lijsten moet slechts één impactfactortype bevatten, terwijl alle andere velden behouden blijven. Binnen elke lijst moeten de tijdschriften geordend staan op impactfactor (de grootste impactfactor eerst). In hetzelfde bestand moeten dus de twee lijsten finaal vlak na elkaar geplaatst worden, waarbij de eerste lijst de impactfactorlijst is, onmiddellijk gevolgd door de 5-jarige impactfactorlijst. Het bestand moet er na herorganisatie als volgt uitzien:

   ```
   # journal_title ISSN     Total_Cites     Impact_Factor   5-Year_Impact_Factor Immediacy_Index...
   INT J COMPUT VISION     0920-5691       8660    10.3    3
   IEEE TRANSACTIONS PATTERN ANAL  0162-8828       24674   7.9     2
   IEEE TRANSACTIONS EVOLUT COMPUT 1089-778X       3849    6.6     4
   J MACH LEARN RES        1532-4435       3328    5.8     12
   ...
   INT J INNOV COMPUT I    1349-4198       907     2.6     15      ORDE
   J WEB SEMANT    1570-8268       438     0.4     13
   IEEE TRANSACTIONS PATTERN ANAL  0162-8828       24674   5.9     2
   INT J COMPUT VISION     0920-5691       8660    5.3     3
   ...
   ```

```
IEEE COMPUT INTELL M    1556-603X       123     2.5     19      ORDE
DATA MIN KNOWL DISC     1384-5810       1470    2.4     20
NEURAL COMPUT   0899-7667       6730    2.3     21
```

Probeer voor elke opdracht zo weinig mogelijk commando's te gebruiken en zorg ervoor dat elk van deze commando's bestaat uit zo weinig mogelijk tekens. De commentaarregels mogen door je commando's niet gewijzigd worden, zelfs niet als er bovenaan bijkomende commentaarregels aan het bestand worden toegevoegd. Alle wijzigingen moeten na elkaar uitgevoerd worden.

Opgave 4

1. RFC *Assigned Numbers* definieert een lijst van poortnummers waarop standaard netwerkdiensten aangeboden worden. Een kopie van deze lijst van deze gestandaardiseerde netwerkdiensten wordt op elke host bijgehouden in het bestand `/etc/services`, waardoor server- en clientprogramma's makkelijk servicenamen kunnen omzetten in de daarmee corresponderende poortnummers. Enkel de *root* gebruiker kan wijzigingen aanbrengen aan deze lijst, maar elke gebruiker kan de informatie uit het tekstbestand `/etc/services` lezen. Gevraagd wordt om onderstaande pipelines aan te vullen zodat de informatie van de beschikbare standaard netwerkdiensten op de aangegeven manier wordt samengevat. Bestudeer hiervoor zelf de inhoud van het tekstbestand `/etc/services` (een kopie van dit bestand werd meegeleverd als hulpbestand, voor het geval je computersysteem geen gebruik maakt van dit configuratiebestand).

   (a)
   ```
   $ cat /etc/services | ...
   70 tcp
   41 udp
   ```

   (b)
   ```
   $ cat /etc/services | ...
   70 TCP
   41 UDP
   ```

2. Gegeven is een `perl` script dat de cijfers $9, 8, \ldots, 1$ op afzonderlijke regels uitschrijft naar standaard uitvoer. Gevraagd wordt om de onderstaande pipelines op een correcte manier aan te vullen zodat de uitvoer van het `perl` script wordt omgevormd tot het formaat zoals aangegeven. Je moet bij elk van deze drie pipelines tekstregels samenvoegen in kolommen. Merk ook op dat $10 = 9 + 8 - 7$ en dat $280 = 10 \times 7 \times 4$.

   (a)
   ```
   $ perl -e 'for ($k=9;$k>0;$k--) { print "$k\n"; }' | ...
   9    8    7
   6    5    4
   3    2    1
   ```

   (b)
   ```
   $ perl -e 'for ($k=9;$k>0;$k--) { print "$k\n"; }' | ...
   10
   7
   4
   ```

   (c)
   ```
   $ perl -e 'for ($k=9;$k>0;$k--) { print "$k\n"; }' | ...
   280
   ```

Opgave 5

De *CSS Validation Service* (`jigsaw.w3.org/css-validator`) is een online dienstverlening die gratis wordt aangeboden door het `W3C` om webdesigners te helpen bij het controleren van de geldigheid van hun Cascading Style Sheets (`CSS`). De structuur van stijlbladen werd immers vastgelegd in technische

CSS specificaties, die formele grammaticaregels (en een vocabularium) bevatten waardoor ze ook door machines kunnen geïnterpreteerd en gecontroleerd worden. Het proces waarbij de opbouw van een document gecontroleerd wordt ten opzichte van een stel grammaticaregels wordt *validatie* genoemd, en dat is precies wat de CSS Validation Service op een geautomatiseerde manier uitvoert.
De CSS Validation Service laat echter alleen maar toe om individuele stijlbladen te valideren. Vaak wordt echter gevraagd om de geldigheid van een volledige lijst van stijlbladen te controleren. Je opdracht bestaat erin om ook dit proces volledig te automatiseren. Hiervoor ga je als volgt te werk:

1. Schrijf een `bash` shell script `validate_css` dat aangeeft of een gegeven CSS stijlblad — dat online beschikbaar is — geldig is of niet. De URL van het CSS stijlblad moet als parameter aan het shell script doorgegeven worden. Op basis daarvan moet het shell script de CSS Validation Service aanroepen via de web service

   ```
   http://jigsaw.w3.org/css-validator/validator?uri=<URL>
   ```

   Binnen deze template-URL vul je op de plaats van `<URL>` de URL van een gegeven CSS stijlblad in. De CSS Validation Service geeft dan als resultaat een XHTML bestand terug, waaruit het shell script de inhoud van het **eerste** `h3` element moet filteren en als boodschap moet uitschrijven naar standaard uitvoer. Gebruik hierbij `ed` of `ex` om binnen het shell script de nodige bestandsmanipulaties uit te voeren op het bestand dat door de web service teruggegeven wordt. Indien het bestand dat door de web service wordt teruggegeven geen `h3` elementen bevat, dan moet het shell script een lege regel naar standaard uitvoer schrijven. Hou er rekening mee dat de inhoud van dit `h3` element over verschillende regels kan gesplitst zijn. Onderstaande sessie illustreert het gebruik van het shell script `validate_css`.

   ```
   $ validate_css www.ugent.be/screen.css
   Sorry!  We found the following errors (3)

   $ validate_css www.ugent.be/print.css
   Congratulations!  No Error Found.

   $ validate_css www.xxx.yyy/zzz.css


   $
   ```

2. Gebruik het shell script `validate_css` om een `bash` shell script `validate_multiple_css` te schrijven waarmee een lijst van URLs van CSS stijlbladen kan gevalideerd worden. Aan dit shell script moet een bestandsnaam als argument meegegeven worden. Dit bestand bevat een lijst van URLs van CSS stijlbladen, waarbij elke URL op een afzonderlijke regel staat. Om elke URL uit de lijst te valideren, moet het shell script als volgt te werk gaan:

   (a) kopieer de lijst van URLs naar een tijdelijk bestand

   (b) vorm het tijdelijk bestand om naar een shell script waarin het validatiescript `validate_css` afzonderlijk wordt aangeroepen voor elk van de opgelijste URLs; voer hierbij telkens ook volgende conversies uit op het resultaat dat door het validatiescript wordt teruggegeven

   - geef de tekst "`CSS is correct.`" terug indien het validatiescript een tekst teruggeeft die begint met het woord `Congratulations`.
   - geef de tekst "`CSS bevat fouten !!`" terug indien het validatiescript een tekst teruggeeft die begint met het woord `Sorry`.
   - geef de tekst "`CSS is niet beschikbaar !!`" terug indien het validatiescript een lege regel teruggeeft.

   (c) voer het omgevormde tijdelijk bestand uit als shell script

   (d) verwijder het tijdelijk bestand

Zorg er hierbij voor dat het shell script `validate_multiple_css` het commando `ed` of `ex` gebruikt om de nodige bestandsmanipulaties uit te voeren in de stappen die hierboven beschreven worden. Onderstaande sessie illustreert het gebruik van shell script `validate_multiple_css`.

```
$ cat css_test_files
www.ugent.be/screen.css
www.ugent.be/print.css
www.sporza.be/html/css/sporza.css
www.csszengarden.com/zengarden-sample.css
www.xxx.yyy/zzz.css

$ validate_multiple_css css_test_files
www.ugent.be/screen.css: CSS bevat fouten !!
www.ugent.be/print.css: CSS is correct.
www.sporza.be/html/css/sporza.css: CSS bevat fouten !!
www.csszengarden.com/zengarden-sample.css: CSS is correct.
www.xxx.yyy/zzz.css: CSS is niet beschikbaar !!

$
```

Het testbestand `css_test_files` werd ook als hulpbestand meegeleverd met het examen.