

Environnement de développement (2I012)

Partiel – Durée : 2h

Valérie Ménissier-Morain & Christoph Lauter

15 mars 2017

Conditions générales

Documents autorisés

Tous les documents sont autorisés. L'usage des clefs USB en lecture seule est autorisé.

Le sujet est disponible depuis la page Web du module <https://www-licence.ufr-info-p6.jussieu.fr/lmd/licence/2016/ue/2I012-2017fev/>.

La version HTML directement disponible sur la page Web est particulièrement avantageuse puisque les lignes très longues y peuvent être mieux vues.

Traitement général du sujet

Parmi les utilitaires Unix qui peuvent être employés à bon escient dans cet examen, citons `sort`, `uniq`, `cut`, `head`, `tail`, `grep`, `sed`, `mktemp` (et bien sûr `bash`).

Comme l'indique le barème, certaines questions sont difficiles. Pensez d'abord à traiter les questions faciles puis correctement les cas simples des questions difficiles plutôt que de vous acharner à traiter tous les cas particuliers.

Il est de loin préférable d'avoir moins de scripts mais qui marchent, plutôt que d'avoir essayé de toucher à tout et de n'avoir rien qui marche à la fin.

N'oubliez pas de tester vos scripts ! Vous avez des fichiers d'exemples à votre disposition, vous avez des exemples textuels dans le sujet que vous pouvez copier/coller, et vous pouvez si ça ne suffit pas créer vos propres exemples de toutes pièces.

Le barème est indicatif et correspond à un total de 28. La note finale est la somme des points à chacune des questions tronquée à 20. Par exemple, une note de 18/28 est ramenée à 18/20 et une note de 22/28 est ramenée à 20/20.

Fichiers à rendre

Tous les fichiers que vous aurez à écrire doivent être placés dans un répertoire nommé `envdev` placé à la racine de votre répertoire personnel. Le contenu du répertoire `envdev` (et seulement celui-là) sera récupéré par l'équipe système de la PPTI dans vos *HOME* à l'issue de l'épreuve.

Les scripts que vous aurez à écrire ne doivent pas dépendre du répertoire où ils sont placés ; ils ne doivent pas contenir de référence à des chemins absolus et doivent pouvoir être exécutés sans problème depuis n'importe quel répertoire. Pour vos tests vous exécuterez au préalable

```
PATH=~ /envdev : $PATH
```

Édition et nom des scripts

Un script s'édite, sous Emacs, en mode *shell-script*, ce que l'on peut obtenir en forçant le mode avec `C-c b` si vous avez créé ce raccourci ou `M-x sh-mode` sinon. Dans ce mode la commande `C-c :` place la convention du *shebang* au début de votre script et le rend exécutable.

Même si vous n'utilisez pas Emacs, n'oubliez pas de rendre vos scripts exécutables par `chmod +x script` et en mettant un *shebang* fonctionnel au début du fichier.

De plus, quand il vous est demandé un script *script*, vous devez rendre le script *script* et non un script *script.sh*.

Fichiers de données

Vous trouverez un certain nombre de fichiers de données dans le répertoire `/Infos/lmd/2016/licence/ue/2I012-2017fev/Partiel`.

Vous n'avez pas besoin de modifier ces fichiers de données, alors ne les copiez pas ! Déclarez plutôt dans votre terminal de test une variable `DONNEES` par

DONNEES=/Infos/lmd/2016/licence/ue/2I012-2017fev/Partiel

et utilisez ensuite \$DONNEES/*fichier*.

Nous aurons recours à ces différents fichiers dans les exemples ci-dessous.

Exécution et tests

Vos scripts doivent être **testés**. Pour cela nous vous conseillons fortement de copier-coller les exemples depuis le sujet au format HTML présent sur le site Web de l'UE.

CES TRACES D'EXÉCUTION SONT CONTRACTUELLES : SUR LE MÊME JEU DE DONNÉES VOTRE SCRIPT DOIT RENDRE EXACTEMENT CE RÉSULTAT !

Fichiers temporaires

Aucun de vos scripts ne doit laisser après son exécution de fichier non prévu et tout fichier temporaire devra être créé avec la commande `mktemp`, par exemple par

```
fichier_temporaire=`mktemp`
```

et supprimé à la fin de vos scripts par

```
rm -f $fichier_temporaire
```


Mais la plupart du temps l'utilisation d'un fichier temporaire n'est qu'une solution de facilité qui masque une mauvaise maîtrise de la composition de commandes.

Messages d'erreur et codes de retour

Lorsque cela est demandé, les scripts signaleront les erreurs avec un code de retour différent de zéro, les messages d'erreur seront produits sur le flux d'erreur standard. Inversement, les scripts signaleront leur bon achèvement par un code de retour égal à zéro.

Conventions typographiques

Nous utilisons dans ce sujet la convention typographique utilisée dans le cours pour les exemples : les lignes en gras commençant par % correspondent à des commandes, le % ne fait pas partie de la commande (c'est le prompt du shell dans notre convention), les lignes qui suivent (jusqu'à une nouvelle ligne en gras ou la fin du paragraphe) sont le résultat de l'exécution de la commande.

Par ailleurs ce sujet comporte des lignes dont la longueur excède la largeur d'une feuille de papier. Dans la version papier les longues lignes sont typographiées en plusieurs lignes, une petite main  indique que la ligne n'est pas terminée et continue sur celle qui suit. La version HTML ne présente pas la même contrainte de longueur de lignes et si ces coupures artificielles vous troublent ou si vous avez besoin de faire des copier-coller vous vous y reporterez avec profit.

Remarques générales

Lorsqu'on manipule des données avec un nombre de mots variable (pour le séparateur usuel, l'espace) (nom d'opérateur, ville ou région) et qu'on veut les considérer comme une donnée unique et non mot à mot lors d'un parcours, il faut utiliser la variable `IFS` avec un autre séparateur que l'espace.

Si votre script ne fonctionne pas, pensez à l'option `-x` de `bash` qui permet de tracer l'exécution du script (préfixez votre script par `bash -x`) et si vous souhaitez visualiser cet affichage verbeux en page par page alors

```
bash -x script arguments 2>&1 | less
```

Lire les consignes et conseils

(feuille à part distribuée avant le début de l'épreuve)

et le sujet entièrement n'est pas une perte de temps !

Il est impératif de tester vos scripts : le sujet est disponible depuis la page Web du module, pensez-y pour *copier-coller* les tests de l'énoncé.

Lignes téléphoniques russes

Un numéro de téléphone russe est composé de 10 chiffres. Il y a évidemment des centaines de millions de numéros de téléphones attribués et des millions de lignes sont réservées par des opérateurs en attente d'attribution à un usager. Ce qui est un numéro de téléphone pour l'usager est une ligne téléphonique pour l'opérateur et nous utiliserons l'un ou l'autre terme suivant le point de vue où l'on se place.

Le fichier `$DONNEES/codes_fuseau.csv` contient la description de toutes les lignes téléphoniques russes. Ce fichier est au format CSV (*Comma Separated Values*), les informations sont séparées par des `;`. On peut voir le contenu de ce fichier comme un tableau à 7 colonnes :

- un *préfixe* à 3 chiffres,
- un *début de zone* à 7 chiffres,
- une *quantité*, qui est un nombre non nul,
- l'*opérateur* de la ligne,
- la *ville du domicile* des usagers,
- la *région du domicile* des usagers,
- le fuseau horaire de ces domiciles sous la forme soit d'une chaîne vide, soit au format `UTC+nb` où *nb* est un nombre compris entre 2 et 11.

L'opérateur, la ville et la région sont des chaînes de caractères translittérées de l'alphabet cyrillique. Elles peuvent contenir notamment des apostrophes et des espaces. De plus une partie du nom de l'opérateur peut se trouver entre guillemets.

Voici un extrait du fichier :

```
% sed -n '38487, 38496 p' $DONNEES/codes_fuseau.csv
347;9836821;179;PAO "Bashinformsvyaz";Duvanskiy;Respublika Bashkortostan;UTC+5
347;9837300;164;PAO "Bashinformsvyaz";Duvanskiy;Respublika Bashkortostan;UTC+5
347;9837500;20;PAO "Bashinformsvyaz";Duvanskiy;Respublika Bashkortostan;UTC+5
347;9837520;1;PAO "Rostelekom";Duvanskiy;Respublika Bashkortostan;UTC+5
347;9837521;427;PAO "Bashinformsvyaz";Duvanskiy;Respublika Bashkortostan;UTC+5
349;2220000;1000;OAO "YAmaltelemek";Salekhard;YAmalo-Nenetskiy avtonomnyj okrug;
UTC+5
349;2221000;1000;OAO "YAmaltelemek";Salekhard;YAmalo-Nenetskiy avtonomnyj okrug;
UTC+5
349;2222000;1000;PAO "Rostelekom";Salekhard;YAmalo-Nenetskiy avtonomnyj okrug;UTC
+5
349;2223000;10;OOO "T2 Mobajl";Salekhard;YAmalo-Nenetskiy avtonomnyj okrug;UTC+5
349;2223999;2001;PAO "Rostelekom";Salekhard;YAmalo-Nenetskiy avtonomnyj okrug;UTC
+5
```

La première ligne se lit : *Tous les numéros de 3479836821 à 3479836821+179-1 sont attribuées à l'opérateur PAO "Bashinformsvyaz" dans la ville de Duvanskiy de la région Respublika Bashkortostan qui est dans le fuseau horaire UTC+5.* Par exemple le numéro 3479836900 fait partie de cette plage de numéros. Le fuseau horaire UTC+5 signifie que par exemple lorsqu'il est 12h à cet endroit, il est $12 - 5 = 7$ h en temps universel (à Londres).

Le fichier `$DONNEES/mobiles.csv` contient la partie qui se rapporte aux lignes de téléphones mobiles.

Le fichier `$DONNEES/codes50000.csv` contient les 50000 premières lignes du fichier `$DONNEES/codes_fuseau.csv` pour avoir un autre fichier de taille raisonnable pour tester certains scripts.

Un tel fichier sera appelé dans la suite un *fichier de lignes téléphoniques*. Ces fichiers sont triés par numéro croissant.

Question 1 – liste_regions – 2 points

Écrire un script `liste_regions` qui prend en paramètre un fichier de lignes téléphoniques, et qui renvoie la liste des régions triée par ordre alphabétique.

Par exemple

```
% liste_regions $DONNEES/codes_fuseau.csv | head -n3
Altajskij kraj
Amurskaya oblast'
Arkhangel'skaya oblast'

% liste_regions $DONNEES/mobiles.csv | tail -n3
YAmalo-Nenetskiy avtonomnyj okrug
YAroslavskaya oblast'
Zabajkal'skiy kraj
```

Question 2 – prefixes_region – 3 points

Écrire un script `prefixes_region` qui prend en paramètre une région et un fichier de lignes téléphoniques, et qui renvoie les préfixes correspondant à cette région dans ce fichier.

Par exemple

```
% prefixes_region "Leningradskaya oblast'" $DONNEES/codes_fuseau.csv | head -n3
800
805
812

% prefixes_region "Permskiy kraj" $DONNEES/mobiles.csv | tail -n3
992
996
999
```

Question 3 – nombre_operateurs – 2 points

Écrire un script `nombre_operateurs` qui prend sur son entrée standard un ensemble de descriptions de lignes téléphoniques et qui affiche le nombre d'opérateurs distincts dans cet ensemble.

Par exemple

```
% nombre_operateurs < $DONNEES/codes_fuseau.csv
2892

% head -n1000 $DONNEES/mobiles.csv | nombre_operateurs
28
```

Question 4 – nombre_lignes_operateur – 3 points

Écrire un script `nombre_lignes_operateur` qui prend en paramètre un nom d'opérateur téléphonique et un fichier de lignes téléphoniques, et qui renvoie le nombre de lignes gérées par cet opérateur dans ce fichier.

Par exemple

```
% nombre_lignes_operateur 'AO "Kordiant"' $DONNEES/codes_fuseau.csv
1000

% nombre_lignes_operateur 'PAO "Rostelekom"' $DONNEES/codes_fuseau.csv
58665326
```

Question 5 – plus_gros_operateur – 4 points

Écrire un script `plus_gros_operateur` qui prend en paramètre un nom de fichier de lignes téléphoniques et renvoie le nom de l'opérateur qui a le plus de lignes téléphoniques.

Par exemple

```
% plus_gros_operateur $DONNEES/mobiles.csv
Opérateur PAO "Mobil'nye TeleSistemy" avec 170884000 lignes.
```

```
% plus_gros_opérateur $DONNEES/codes50000.csv
```

Opérateur PAO "Rostelekom" avec 13457948 lignes.

Question 6 – operateurs_locaux – 2+2 points

Écrire un script `opérateurs_locaux` qui prend en paramètre une ville et un fichier de lignes téléphoniques, et qui renvoie les noms des opérateurs se rapportant à cette ville dans ce fichier dans l'ordre alphabétique..

Certains noms de villes apparaissent dans plusieurs régions. Pour indiquer de quelle ville il s'agit elle pourra être désignée sous la forme `<ville>/<region>`.

La version simple vaut 2 points. La version complète vaut 4 points.

Par exemple

```
% operateurs_locaux Moskva $DONNEES/mobiles.csv | head -n3
```

```
AO "GLONASS"
```

```
AO "Kompaniya TransTeleKom"
```

```
OA "ASVT"
```

```
% operateurs_locaux "Tul'skaya oblast'" $DONNEES/codes_fuseau.csv
```

```
% operateurs_locaux Leninskiy $DONNEES/codes_fuseau.csv | head -n3
```

```
AO "TK TEL"
```

```
GUP RK "Krymtelekom"
```

```
OA "ASVT"
```

```
% operateurs_locaux "Leninskiy/Tul'skaya oblast'" $DONNEES/codes_fuseau.csv
```

```
PAO "Rostelekom"
```

```
ZAO "Kontakt"
```

Question 7 – localisation_numero – 5 points

Écrire un script `localisation_numero` qui prend en paramètre un numéro à 10 chiffres et un fichier de lignes téléphoniques, et qui indique la ville, la région et le fuseau horaire correspondant à ce numéro s'ils existent et affiche un message d'erreur sinon.

Par exemple

```
% localisation_numero 8137899950 $DONNEES/codes_fuseau.csv
```

```
8137899950;Vyborg;Leningradskaya oblast';UTC+3
```

```
% localisation_numero 8137899999 $DONNEES/codes_fuseau.csv
```

```
8137899999;Vyborg;Leningradskaya oblast';UTC+3
```

```
% localisation_numero 8137900000 $DONNEES/codes_fuseau.csv
```

```
8137900000;Priozerskiy;Leningradskaya oblast';UTC+3
```

```
% localisation_numero 3011000000 $DONNEES/codes_fuseau.csv
```

```
Numéro non attribué
```

```
% localisation_numero 3412971600 $DONNEES/codes_fuseau.csv
```

```
Numéro non attribué
```

Question 8 – heure_acceptable – 5 points

Écrire un script `heure_acceptable` qui prend 6 paramètres :

- le numéro de l'appelant et l'heure locale d'appel
- le numéro appelé, l'heure de début et de fin de la plage où on veut l'appeler selon son heure locale
- le fichier des lignes téléphoniques où puiser l'information

et qui renvoie un code de retour nul si et seulement si l'heure locale pour l'appelant se trouve dans la plage horaire prévue pour l'appelé.

Certains numéros n'ont pas de fuseau horaire associé (essentiellement les numéros gratuits, de préfixe 800, associés à la région Rossijskaya Federatsiya mais peu importe). Il s'agit de numéros appelables à toute heure. On rendra donc toujours un code de retour nul si le numéro appelé n'a pas de fuseau horaire associé.

De même si un usager appelle d'un tel numéro on supposera qu'il appelle localement son correspondant et on rendra un code de retour nul si l'heure d'appel est située dans la plage horaire prévue.

Par exemple

```
% heure_acceptable 8633220200 10 3012480900 9 17 $DONNEES/codes_fuseau.csv && 📡  
📡echo "Horaire de bureau"  
Horaire de bureau
```

```
% heure_acceptable 8633220200 15 3012480900 9 17 $DONNEES/codes_fuseau.csv || 📡  
📡echo "Soir"  
Soir
```

```
% heure_acceptable 8633220200 15 8002580606 9 17 $DONNEES/codes_fuseau.csv && 📡  
📡echo "Toujours ok"  
Toujours ok
```

```
% heure_acceptable 8002580606 15 4935220400 9 17 $DONNEES/codes_fuseau.csv && 📡  
📡echo "Toujours ok"  
Toujours ok
```