Nom :
Prénom :

Contrôle de connaissances 2017/2018
des étudiants de 1ère année (EI1)

CSC3102 : Contrôle Final Session 2 — Sujet 2

Date : lundi 29 janvier 2018 — Durée : 1 heure 30

Coordonnateurs : É. Brunet et G. Thomas

Documents autorisés : polycopié papier et notes manuscrites

Consignes :

- Toute entorse à ces consignes est pénalisée.
- Répondez aux questions sur la copie.
- En plus des règles usuelles des contrôles, il est strictement interdit de :
  1. commencer le contrôle avant le signal de démarrage de l'enseignant,
  2. continuer le contrôle après le signal d'arrêt de l'enseignant.
- Le barême est donné à titre indicatif.

Nom Prénom :

1 QCM - 5 points

Consignes : Entourez les numéros des propositions que vous estimez correctes. Pour chaque question, il se peut qu'aucune, qu'une, que plusieurs ou que toutes les réponses proposées soient correctes.
Principe de notation (à titre indicatif seulement) : 1 bonne réponse = 1 point, 1 réponse non fournie = 0 point, 1 mauvaise réponse = -1 point

1. Vous êtes positionné à la racine de votre compte. Pour effacer le fichier `prog.c` se trouvant dans le sous-répertoire `Programmation` de votre répertoire de connexion, vous utilisez la commande :
   (a) `del prog.c`
   (b) `ls  /Programmation/prog.c | rm`
   (c) `rm Programmation/prog.c`
   (d) `rm ../Programmation/prog.c`
   (e) `rm $HOME/Programmation/prog.c`

2. La commande `grep tomate carotte | wc -l`
   (a) n'affiche rien
   (b) affiche le nombre de lignes contenant le mot `tomate` dans le fichier `carotte`
   (c) affiche les lignes du fichier `tomate` contenant le mot `carotte`, puis affiche le nombre de lignes du fichier `tomate`
   (d) affiche les lignes du fichier `carotte` contenant le mot `tomate`, puis affiche le nombre de lignes du fichier `carotte`
   (e) affiche le nombre de lignes du fichier `tomate` contenant le mot `carotte`

3. Pour rendre un fichier exécutable, il faut :
   (a) le placer dans un répertoire nommé `bin`
   (b) le compiler avec la commande `javac`
   (c) placer l'indication `#!/bin/sh` sur la première ligne
   (d) changer son statut avec la commande `chown`
   (e) changer son statut avec la commande `chmod`

4. Un processus s'exécutant en arrière-plan :
   (a) peut être détruit par la commande `kill`
   (b) peut être détruit par `CTRL-C`
   (c) peut être détruit par `CTRL-Z`
   (d) ne peut pas être détruit

5. La commande `echo *`
   (a) affiche une étoile (*)
   (b) affiche les noms de tous les fichiers du répertoire courant
   (c) permet de ramener le curseur en début de ligne
   (d) affiche un saut de ligne (passage à la ligne)
   (e) affiche les noms des fichiers cachés du répertoire courant

6. La commande `chmod 364 .` ayant été exécutée,
   (a) le groupe ne peut pas supprimer un fichier dans le répertoire de travail
   (b) le groupe peut lire le contenu des entrées du répertoire de travail
   (c) l'utilisateur a le droit de créer un fichier dans le répertoire de travail
   (d) l'utilisateur a le droit de lire le contenu du répertoire de travail
   (e) les autres peuvent lister les entrées du répertoire de travail

2 Tournoi de Ping-pong ! - 15 points

Consignes : Répondez à chaque question dans l'encadré associé. Si vous deviez ne pas avoir assez de place, signalez dans l'encadré que vous continuez à la fin de la copie.
Sauf mention contraire, les cas où les paramètres des scripts sont incorrects ne sont pas à traiter.
Les questions sont indépendantes les unes des autres. Cependant, afin de répondre à une question donnée, vous devrez avoir lu les précedentes.
Vous avez à disposition les scripts `P.sh` et `V.sh` utilisés pendant le module.
Le barème est donné à titre indicatif.

Après une année d'entraînement intensif, il est temps pour l'association sportive de ping-pong locale d'organiser un tournoi de fin d'année !

1. **Mise en place de l'infrastructure (3 points) :** Actuellement, les données internes de l'association sont organisées suivant l'arborescence suivante :

```
~ - Assos ---- Scripts - pingpongParty.sh
           \\\          \ letsPlay.sh
            \\\
             \\\ EnCours - partiesOuvertes.txt
              \\ Membres - 2016-2017 - membres.txt
               \          \ 2017-2018 - membres.txt
                \ Tournoi - inscrits.txt
                           \ tableau.txt
```

   De manière à ne perdre aucune donnée antérieure et à collecter celle relative au tournoi annuel, l'arborescence actuelle doit être réorganisée comme suit :

```
~ - Assos --- Scripts - pingpongParty.sh
          \\\         \ letsPlay.sh
           \\\
            \\\ EnCours - partiesOuvertes.txt
             \\ Membres - 2016-2017 - membres.txt
              \          \ 2017-2018 - membres.txt
               \ Tournois - 2016-2017 - inscrits.txt
                          \           \ tableau.txt
                           \ 2017-2018
```

Dans le cadre suivant, écrivez la suite de commandes modifiant l'arborescence courante de manière à ce que le répertoire `Tournoi` soit renommé en `Tournois`, que les fichiers le composant soient dans un répertoire `2016-2017` et qu'un répertoire `2017-2018` soit prêt à accueillir les données de l'année :

2. **Inscription des joueurs (6points) :** Les joueurs sont invités à s'inscrire au tournoi grâce au script `register.sh` du répertoire `Scripts`. Ce script prend un seul paramètre correspondant au nom du joueur voulant s'inscrire. Après avoir vérifié que le joueur est membre de l'association (en le recherchant dans le fichier `membres.txt` de l'année en cours), le script doit ajouter le nom du joueur à la fin du fichier `inscrits.txt` de l'année courante. Attention, il faut s'imaginer que les inscriptions se font en ligne et que plusieurs joueurs peuvent donc s'inscrire en même temps.
   Écrivez le script `register.sh` dans le cadre suivant :

3. **Tournoi (6 points) :** Les joueurs à présent inscrits, le tournoi est lancé !
   Le script `letsPlay.sh` (que vous avez écrit au CF1) permet d'apparier des joueurs pour qu'ils disputent une partie de ping-pong. Ce script a deux paramètres d'entrée. Le premier correspond au nom du joueur voulant jouer. Le script initialise une partie pour ce joueur ou le fait rejoindre une partie initialisée pour un autre joueur. Le second paramètre correspond au chemin vers un fichier auquel sera ajouté le nom du joueur en cas de victoire.
   Un tournoi se déroule en plusieurs tours. Au premier tour, tous les joueurs inscrits participent ; aux tours suivants, seuls les joueurs ayant gagné leur rencontre au tour précédent. Petite précision importante pour votre implémentation : nous partons du principe que le nombre de joueurs disputant un tour est toujours pair.
   Écrivez un script `tournoi.sh` qui permette de lancer les différents tours jusqu'à obtenir le vainqueur final. À chaque tour, vous devez faire jouer l'ensemble des joueurs dont le nom figure dans le fichier `inscrits.txt` (grâce au script `letsPlay.sh`) et faire inscrire les gagnants dans un fichier que vous nommez `gagnants.txt`. Attention à bien attendre la fin de toutes les parties du tour en cours avant de passer au suivant. De plus, d'un tour à l'autre, les gagnants du tour courant deviennent les inscrits au tour suivant.
   Écrivez le script `tournoi.sh` dans le cadre suivant :