Licence III Informatique — UFR des Sciences
Université de Picardie Jules Verne
2017-2018

Programmation des Systèmes d'Information

Sujet du projet : mini-shell

> *Le projet est à réaliser **en binôme**. Il doit être le résultat d'un travail **personnel et original**. Il sera à rendre par voie électronique et fera l'objet d'une soutenance rapide à une date qui vous sera communiquée ultérieurement. Pour rappel, le plagiat ou la reprise même partielle d'un code dont vous n'êtes pas les auteurs sont des fraudes susceptibles de sanctions disciplinaires.*

Ce projet consiste en la réalisation d'un shell Unix très élémentaire, permettant au minimum :

— De se déplacer dans les répertoires

```
$ cd dir
```

— D'exécuter des commandes simple, en "avant plan" et en "arrière plan"

```
$ commande
$ commande &
$ commande1 ; commande2 ; commande3 & commande4 ; ...
```

— De rediriger les entrées sorties des commandes

```
$ commande > file.output
$ commande 2> file.error
$ commande < file.input
$ commande >> file.appout
$ commande 2>> file.apperr
$ commande >&2
$ commande 2>&1
$ commande1 | commande2 | commande3 | ...
```

— De gérer les variables d'environnement

```
$ export VAR="ma variable"
$ echo $VAR
$ unset VAR
...
```

— De gérer les opérateurs `!`, `&&` et `||` comme les gère le bash

Vous pouvez bien sûr également implémenter les redirections `<<` et `<<<` ; la gestion des jobs (`jobs`, `fg` , ...) ; les commandes imbriquées, les structures de contrôle `if`, `while`, des "jokers" (`*`, `?`, ...) ; l'historique des commandes ; la complétion automatique ; etc.
Des points bonus vous seront attribués en fonction de la qualité de l'implémentation.

Vous organiserez votre code dans plusieurs fichiers et le projet devra être compilé et installé grâce à l'utilitaire `CMake`. Vous permettrez de choisir le compilateur, les options de compilation et l'emplacement de l'installation (via une commande `make install`).

Pour la réalisation du projet, vous respecterez les consignes suivantes :

— Chaque fichier source du projet commencera par un commentaire respectant la forme suivante :

```
/*
   Fichier foo.c : description rapide du contenu
   Auteur : Jean Dupont
   Dépendances : bar.h foo.h
*/
```

— Chaque fonction sera commentée de la manière suivante :

```
/*
   Fonction foo
   Paramètre a : entier permettant de passer la valeur de...
   Paramètre b : entier permettant de passer la valeur de...
   Retourne un entier, valeur de...
*/
int foo(int a, int b) {
   ...
}
```

— Le code sera « raisonnablement » commenté, donnant les informations essentielles sur son fonctionnement

— Votre code sera organisé comme suit :
  — Un répertoire principal (racine du projet) nommé à partir des noms des deux binômes (exactement `shell-Nom1_Nom2/`)
  — Le sous-répertoire `src` contiendra les sources du projet
  — Le sous-répertoire `doc` contiendra la documentation succinte du projet
  — Un fichier `CMakeLists.txt` à la racine du projet permettra de le compiler en respectant les dépendances de ses différentes parties
  — Un fichier `exemples` contenant une liste de commandes testées avec succès dans votre shell (une par ligne)

— Le programme compilé se nommera obligatoirement `minishell`

Les différentes étapes de réalisation du projet seront traitées en partie durant les séances de TP :

— **Étape 1 :** parsing simple et gestion des variables d'environnement.
— **Étape 2 :** exécution et redirections simples.
— **Étape 3 :** pipe et enchaînement des commandes.
— **Étape 4 :** parsing avancé.