TP 6

SYSTEMES D'EXPLOITATION

INFORMATIONS GÉNÉRALES

- Date de reddition : 20 décembre 2021 à 20:00
- Travail en groupe est possible (maximum 2 par groupe).
- Délivrables:
  - Manuel de votre programme shell en PDF
  - Code source commenté et fonctionnel (avec Makefile!)
  - Dossier à rendre : <Prénom1>.<Nom1>.<Prénom2>.<Nom2>.TP6.zip(ou tar.gz)

INFORMATIONS GÉNÉRALES

- Références au cours :
  - 7. Entrées / Sorties (I/O)
  - 9. Processus
  - 11. Signaux
- Objectifs :

L'objectif général du TP est de créer son propre shell avec lequel on pourra exécuter les programmes du système.

- Créer des processus avec la fonction fork;
- Gérer ces processus, notamment éviter les zombies;
- Gérer les signaux envoyés au shell.

LE SHELL

- Utilisateur peut:
  - Intéragir avec le systéme
  - Exécuter des commandes
- Le shell doit:
  - Créer plusieurs processus (fork)
  - Gérer ces processus (evitez les zombies!)
  - Commandes builtin

1ÈRE PARTIE

- Exécute une commande d'utilisateur. Il y a deux types:
  - Job: programme du système (ex. ls, pwd, ps)
  - Builtin: commandes implémentées directement dans le shell (e.g. cd)

PARSING

- Comment lire l'entée d'utilisateur ?
  - Transformer la chaine de caractère en un tableau de chaine comme *argv*, *argc* (c.f. fonction main)
  - Dans notre cas on considérera que l'espace et la tabulation sont les deux seuls caractères qui séparent les arguments.
    - Consultez la fonction *strtok* dans le manuel. La commande *realloc* peut etre utile aussi.

```
char *token = strtok(userInput, " ");
while (token != NULL){
    print("%s\n",token);
    token = strtok(NULL, " ");
}
```

COMMANDES BUILTIN

- Deux commandes seront implémentées:
  - exit: le shell ce termine proprement
  - cd: le shell change le répertoire courant

JOBS

- Lorsque la commande tapée par l'utilisateur n'est pas builtin:
  - Le shell exécutera le programme avec la commande *execve:*

```
#include <unistd.h>
int execve(const char *filename, char *const argv[], char *const envp[]);
```

  - Le shell attendra que le programme se termine. Puis, il devra afficher le code sortie du programme si il est disponible (e.g. "Foreground job exited with code 0"), sinon un simple message (e.g. "Foreground job exited")
- Evitez les zombies!

2ÈME PARTIE ET BONUS

- Jobs en tâche de fond et signaux

- Bonus:
  - Exécution de plusieurs jobs en tâche de fond
  - Redirection des entrées/sorties
  - Implémentation de pipes

Plus de détails pour le 2éme partie seront présentés le 6 décembre