Mini-Shell (`mysh`)

Description du Projet

`mysh` est un mini-shell interactif implémenté en C, offrant une interface utilisateur simple pour exécuter des commandes système, gérer des tâches en arrière-plan, manipuler des variables locales et d'environnement, et gérer des redirections et pipelines. Ce projet illustre le fonctionnement d'un interpréteur de commandes Unix personnalisé.

Répartition des Contributions

La répartition des contributions au projet est la suivante :

- **Mamoudou Diaby** : 30%
- **Ali Can Cebi** : 50%
- **Ihab Toumi** : 20%

Fonctionnalités

- **Commandes système :**
    - Exécution de commandes système standard (ex. `ls`, `cat`, `grep`).
    - Gestion des pipelines (`|`) pour enchaîner plusieurs commandes.
    - Redirection d'entrée/sortie (`>`, `<`, `2>`, `>&`) avec support pour l'ajout (`>>`).
- **Commandes internes :**
    - `cd`: Changement de répertoire.
    - `exit`: Quitter le shell.
    - `status`: Afficher le statut de la dernière commande exécutée.
    - `myjobs`: Afficher les tâches en cours en arrière-plan.
    - `myfg <job_id>`: Ramener une tâche en arrière-plan au premier plan.
    - `mybg <job_id>`: Relancer une tâche suspendue en arrière-plan.
    - `set name=value`: Définir une variable locale.
    - `unset name`: Supprimer une variable locale.
    - `setenv name=value`: Définir une variable d'environnement.
    - `unsetenv name`: Supprimer une variable d'environnement.

- **Commandes personnalisées :**

- `myls`: Listage avancé des fichiers avec tri, couleur, et récursivité.
- `myps`: Affichage des processus en cours avec des informations détaillées (CPU, mémoire, utilisateur, etc.).

Structure du Projet

Répertoire principal :

- `mysh.c`: Point d'entrée principal du shell, gestion des commandes utilisateur et des fonctionnalités internes (`cd`, `exit`, etc.).
- `executor.c`: Contient la logique pour exécuter des commandes externes et gérer les tâches en arrière-plan.
- `parser.c`: Analyse les commandes utilisateur (supporte `&&`, `||`, `;`).
- `redirection.c`: Gère la redirection des entrées/sorties.
- `wildcard.c`: Implémente l'expansion des caractères génériques (`*`, `?`, `[...]`).
- `process_manager.c`: Gère les tâches en arrière-plan et leur état (`myjobs`, `myfg`, `mybg`).
- `variable.c`: Gestion des variables locales et d'environnement avec mémoire partagée.
- `myls.c`: Implémentation de la commande `myls`.
- `myps.c`: Implémentation de la commande `myps`.

Répertoire `include` :

Contient les fichiers d'en-tête pour chaque composant du projet, par exemple :

- `executor.h`
- `parser.h`
- `redirection.h`
- `wildcard.h`
- `process_manager.h`
- `variable.h`

`Tests :`

`Prise en charge d'enchaînements`

`- Conditionnels :`

```
bash
~> gcc -o mysh myshell.c && ./mysh
~> test -d .can || mkdir .can
```

- Inconditionnels :

bash

~> ls ; cat /etc/passwd

Wildcards :

Les lignes de commandes du shell remplacent les caractères « jokers » (*, ?, []) par leur(s) correspondance(s) calculée(s) à partir du répertoire courant.

bash

~> cat *.[ch]

~> ls ../[A-Z.]*[^~]

~> wc -l /etc/?????

ex:

bash

mysh ~> wc -l /etc/?????

wc: /etc/gnome: est un dossier

  0 /etc/gnome

  3 /etc/magic

wc: /etc/avahi: est un dossier

  0 /etc/avahi

    .

    .

    .

wc: /etc/mysql: est un dossier

  0 /etc/mysql

118 total

Changer de répertoire :

Afin de pouvoir se ballader dans l’arborescence de fichiers, le shell dispose d'un cd interne.

Sortie du Shell :

Pour la sortie du shell

- La commande interne exit (quitte sans tue les tâches de fonds);

- Le Ctrl c qui demande confirmation et tue les tâches de fonds avant de quitter;

bash

mysh ~> ^C

Do you want to quit mysh? (y/n) y

Code de retour d'un processus :

La commande interne status affiche pour le dernier processus xxxx exécuté en premier plan (foreground):

- xxxx terminé avec comme code de retour YYY si le processus s’est terminé normalement;

- xxxx terminé anormalement dans le cas d’une terminaison anormale (comme par exemple l’interruption via un Ctrl c);

bash

mysh ~> status

No terminated process.

mysh ~> ls

files.txt include Makefile mysh Rapport src tests.txt test.txt

mysh ~> status

ls terminé avec comme code de retour 0

mysh ~>

Lister le contenu d'un répertoire :

Pour lister le contenu d'un répertoire, le shell utilise un programme nommé myls (équivalent de ls -l).

bash

~> myls / .. foo bar duc

~> myls -aR /

~> myls -R

~> myls -R -a /

~> myls -Ra ~

Afficher l’état des processus en cours :

Pour afficher l'état des processus en cours, le shell utilise un programme myps (équivalent à ps aux avec affichage en couleur en fonction de l'état du processus).

bash

/home/trab/Bureau/mysh ~> myps

USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND

root 1 0.0 0.0 0 0 ? S 22:26 00:03 /sbin/init

root 2 0.0 0.0 0 0 ? S 22:26 00:00 [(kthreadd)]

root 3 0.0 0.0 0 0 ? S 22:26 00:00 [(pool_workqueue_release)]

root 4 0.0 0.0 0 0 ? I 22:26 00:00 [(kworker/R-rcu_g)]

root 5 0.0 0.0 0 0 ? I 22:26 00:00 [(kworker/R-rcu_p)]

```
root           6   0.0   0.0        0       0 ?      I 22:26 00:00
[(kworker/R-slub_)]

root           7   0.0   0.0        0       0 ?      I 22:26 00:00
[(kworker/R-netns)]

root          12   0.0   0.0        0       0 ?      I 22:26 00:00
[(kworker/R-mm_pe)]

root          13   0.0   0.0        0       0 ?      I 22:26 00:00
[(rcu_tasks_kthread)]

    .

    .

    .

   mamoudou   36981   0.0   0.0        0       0 0      R 21:41 00:00
./mysh

mamoudou   37125   0.0   0.0        0       0 0      S 21:47 00:00
/snap/chromium/3002/usr/lib/chromium-browser/chrome --type=renderer
--string-annotations --crashpad-handler-pid=4023 --enable-crash-
reporter=,snap --change-stack-guard-on-fork=enable --lang=fr --num-
raster-threads=2 --enable-main-frame-before-activation -

root       37154   0.0   0.0        0       0 0      I 21:47 00:00
[(kworker/u9:2-rtw_tx_wq)]

root       37200   0.0   0.0        0       0 0      I 21:52 00:00
[(kworker/2:2-events)]

root       37247   0.0   0.0        0       0 0      I 21:57 00:00
[(kworker/u8:3)]

root       37273   0.0   0.0        0       0 0      I 21:59 00:00
[(kworker/u9:0-rtw_tx_wq)]

mysh:/home/mamoudou/Documents/mini-shell ~>
```

Permettre l'usage du pipeline (|) :

Il permettra de rediriger la sortie standard du programme précédant le | vers l’entrée standard du programme suivant le |.

bash

```
~> ls | sort -r
~> ps | grep mysh | wc -l
```

Le premier exemple permet de trier dans l’ordre alphanumérique inverse les entrées du répertoires courant, alors que le second permet de compter le nombre de processus mysh en cours d’exécution.

Redirections vers ou depuis un fichier :

```
bash
~> echo "Bonjour" > files.txt
~> echo "Nouveau" >> files.txt
~> cat < files.txt
~> ls nn 2> test.txt
~> ls cc 2>> test.txt
~> ls jj Rapport >& tests.txt
~> ls kk Rapport >>& tests.txt
```

Premier et arrière plans :

```
bash
~> emacs &
~> ls -lR | gzip > ls-lR.gz &
```

Commande myjobs :

```
bash
```

Les variables :

```
bash
```

```
~>set a=foo
~>echo $a
foo
~>setenv b=duc
~>mysh
~>echo $b
duc
~>set a=bar
~>set b=tmp
~>echo $b
tmp
~>unset a
```
