Responsable pédagogique	AF	AM	PB	**PM**
Période	Sem1	Sem2	**Sem 3**	Sem4
Volume horaire	Cours/TD		TP	
			6-8	

PSYST

TP 2 myshell

Indicateur temporel (hors rédaction du compte-rendu) :

questions	1h	2h	3h	4h	5h	6h-8h
1						
2						
3-4						

Documents à rendre : Compte-rendu contenant à minima des explications concises des commandes développées ; les sources (avec entête standard) et les résultats obtenus.

Le but est de créer un programme qui se comporte un peu comme le shell, c'est à dire que l'on puisse lui passer une liste d'arguments et de commande système.

Ex : ps -ax ; ls -l ;

1. Gestion de la ligne

A partir du source myshell2_suj.c fourni ; codez une fonction de prototype int *read_arg(int fd, char*line, int size) ;* qui lit les arguments au clavier (une fois le programme lancé) et les stocke dans *line*.

Cette fonction renverra 0 si un EOF est généré ; le nombre d'octets lus autrement.

Vérifiez la ligne saisie par un affichage type printf(" >%s< ",line) ;

2. Isoler les arguments

Principe algorithmique :

POUR chaque nouvelle ligne

    list_arg[i] ← ième chaine de la ligne courante ; { séparation des arguments par les espaces,remplacés par NUL }

    SI EOF ALORS

        SORTIR ;

    FINSI

FIN POUR

Ainsi :list_arg[0] sera le nom de la commande (1ere chaine de la ligne courante)

list_arg[1] sera la 1ère option (2 eme chaine)

etc...

list_arg[n] ← NULL ; (n+1 ieme chaine se termine par le pointeur NULL)

(Cf Annexe 1 Fig.1)

Pour coder : Vous déclarerez le tableau char*list_arg[N_ARGS] ; dans le main(), et vous allez faire une boucle qui parcoure toute la chaine line en remplaçant les espaces par des 0.

Cf Annexe 1 fig2

Astuce : Faites pointer list_arg[i] sur line + i, i.e l'adresse du début de la ième " sous-chaine "

3. Lancement de la commande

Modifier le programme actuel pour lancer la commande réelle, avec ses options.

Vous allez utiliser pour cela les fonctions execv(char *<chemin>, char **<argv>) ; execl(char *<chemin>, char *<argv0>, …, 0) ;

Pensez qu'il faut préciser le chemin de la commande /bin en général et donc argv0 sera la commande, et l'option viendra en 3eme argument de execl !!

exemple : pour faire un ls -l il faut que votre programme appelle execl(" /bin/ls ", "ls ", "-l ",0) ; où *ls* sera substitué bien sur par *list_arg[0]* et *-l* par *list_arg[1]* .

Vous pouvez par exemple pré-formater une chaine de caractères : char cmd[50]; sprintf(cmd,"/bin/%s",list_arg[0]);

4. Lancement par substitution de processus

Nous allons utiliser un mécanisme courant sous unix, la duplication de processus lourds (fork()) puis substitution par execv(), excl().

Coder et tester le programme en Annexe 2.

Intégrer le à votre shell actuel. Notez les évolutions de comportement en cas de bonnes et mauvaises commandes. (Cf figure 4)

Annexe 1 : exemples de résultats attendus

```
maylaenderpatrick@ubuntu:~/Documents/TS2/psyst/correc$ ./myshell
ps -aux | more
>ps -aux | more<
arg[0] --> ps
arg[1] --> -aux
arg[2] --> |
arg[3] --> more
maylaenderpatrick@ubuntu:~/Documents/TS2/psyst/correc$
```

Figure 1 : affichage des arguments isolés depuis la ligne de commande

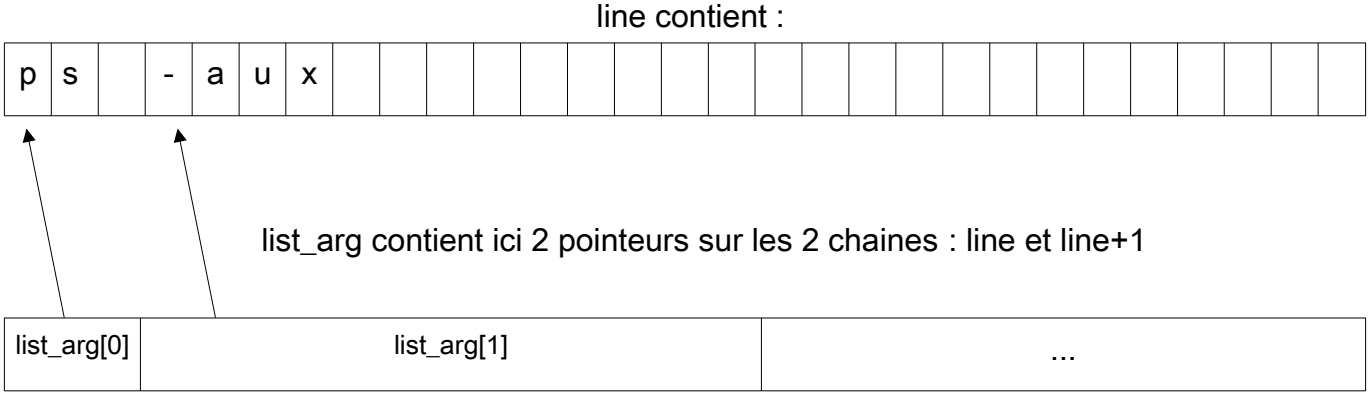

Figure 2 : schéma de principe

```
maylaenderpatrick@ubuntu:~/Documents/TS2/psyst/correc$ ./myshell
ls -l
>ls -l<
arg[0] --> ls
arg[1] --> -l
total 84
-rwxrwxr-x 1 maylaenderpatrick maylaenderpatrick 8766 juil.  4 19:19 exec
-rw-rw-r-- 1 maylaenderpatrick maylaenderpatrick  307 juil.  4 19:18 exec.c
-rw-rw-r-- 1 maylaenderpatrick maylaenderpatrick  302 juil.  4 19:18 exec.c~
-rwxrwxr-x 1 maylaenderpatrick maylaenderpatrick 8622 juil.  3 14:52 mycat
-rw-r--r-- 1 maylaenderpatrick maylaenderpatrick  288 juil.  2 16:57 mycat.c
-rwxrwxr-x 1 maylaenderpatrick maylaenderpatrick 9303 juil.  3 16:59 myrm
-rw-r--r-- 1 maylaenderpatrick maylaenderpatrick 2339 juil.  3 17:05 myrm.c
-rw-r--r-- 1 maylaenderpatrick maylaenderpatrick 2315 juil.  3 17:02 myrm.c~
-rwxrwxr-x 1 maylaenderpatrick maylaenderpatrick 8762 juil.  4 19:20 myshell
-rw-rw-r-- 1 maylaenderpatrick maylaenderpatrick 1562 juil.  4 19:20 myshell.c
-rw-rw-r-- 1 maylaenderpatrick maylaenderpatrick 1550 juil.  4 19:11 myshell.c~
-rw-r--r-- 1 maylaenderpatrick maylaenderpatrick   96 juil.  2 15:42 rep.sh
drwxrwxr-x 4 maylaenderpatrick maylaenderpatrick 4096 juil.  3 16:24 tr1
```

Figure 3 : exécution du " shell " pour un ls -l

Annexe 2 : Programme multi-tâches

```
main()
{

int pid =fork();
if(pid == 0) //fils
{
        execl("/bin/ps","ps","-aux",0);

        printf("je suis le fils de pid ---> %d\n",getpid());
        sleep(2);
}
else
{
        printf("je suis le pere de pid ---> %d\n",getpid());
        wait(0);
        }
```

```
je suis le pere de pid ---> 7414
je suis le fils de pid ---> 7415
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.2  33772  2736 ?        Ss   10:18   0:05 /sbin/init
root         2  0.0  0.0      0     0 ?        S    10:18   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    10:18   0:02 [ksoftirqd/0]

jaylaen+  7414  0.0  0.0   4196   352 pts/8    S+   19:40   0:00 ./exec
jaylaen+  7415  0.5  0.1  18404  1320 pts/8    R+   19:40   0:00 ps -aux
```

Figure 4 : shell finalisé (duplication de processus)
appliqué au ps

```
naylaenderpatrick@ubuntu:~/Documents/TS2/psyst/correc$ ./myshellfork
ls -l
>ls -l<
arg[0] --> ls
arg[1] --> -l
je suis le pere de pid ---> 7441
je suis le fils de pid ---> 7442
total 104
-rwxrwxr-x 1 maylaenderpatrick maylaenderpatrick 8766 juil.  4 19:40 exec
-rw-rw-r-- 1 maylaenderpatrick maylaenderpatrick  309 juil.  4 19:40 exec.c
-rw-rw-r-- 1 maylaenderpatrick maylaenderpatrick  307 juil.  4 19:18 exec.c~
-rwxrwxr-x 1 maylaenderpatrick maylaenderpatrick 8622 juil.  3 14:52 mycat
-rw-r--r-- 1 maylaenderpatrick maylaenderpatrick  288 juil.  2 16:57 mycat.c
-rwxrwxr-x 1 maylaenderpatrick maylaenderpatrick 9303 juil.  3 16:59 myrm
-rw-r--r-- 1 maylaenderpatrick maylaenderpatrick 2339 juil.  3 17:05 myrm.c
-rw-r--r-- 1 maylaenderpatrick maylaenderpatrick 2315 juil.  3 17:02 myrm.c~
-rwxrwxr-x 1 maylaenderpatrick maylaenderpatrick 8762 juil.  4 19:20 myshell
-rw-rw-r-- 1 maylaenderpatrick maylaenderpatrick 1562 juil.  4 19:20 myshell.c
-rw-rw-r-- 1 maylaenderpatrick maylaenderpatrick 1550 juil.  4 19:11 myshell.c~
-rwxrwxr-x 1 maylaenderpatrick maylaenderpatrick 8969 juil.  4 19:47 myshellfork
-rw-rw-r-- 1 maylaenderpatrick maylaenderpatrick 1437 juil.  4 19:47 myshellfork.c
-rw-rw-r-- 1 maylaenderpatrick maylaenderpatrick 1756 juil.  4 19:46 myshellfork.c~
-rw-r--r-- 1 maylaenderpatrick maylaenderpatrick   96 juil.  2 15:42 rep.sh
drwxrwxr-x 4 maylaenderpatrick maylaenderpatrick 4096 juil.  3 16:24 tr1
```

Figure 5 : shell finalisé (duplication de processus)
appliqué au ls