Corrigé de Système d'exploitation (Yves de Verdière Ing1 2K16)

- Quelles sont les infos indispensables afin de gérer correctement un processus pour un S E ?

Le PID (identifiant du processus) et sa priorité

- A quoi sert la mémoire virtuelle ? Quelles en sont les applications ?

Wikipedia :

En informatique, le mécanisme de **mémoire virtuelle** a été mis au point dans les années 1960. Il repose sur l'utilisation de traduction à la volée des adresses (virtuelles) vues du logiciel, en adresses physiques de mémoire vive. La mémoire virtuelle permet :

- d'utiliser de la mémoire de masse comme extension de la mémoire vive ;
- d'augmenter le taux de multiprogrammation ;
- de mettre en place des mécanismes de protection de la mémoire ;
- de partager la mémoire entre processus.

- A quoi sert la ligne # !/bin/bash ?

Cette ligne est un commentaire spécial qui permet de préciser à l'environnement que les commandes qui suivent sont du script Bash.

On pourrait remplacer cette ligne par # !/bin/python pour executer du python, # !/bin/ls si on veut « redéfinir » la commande ls.

!/bin/sh , # !/bin/ksh pour d'autres versions de shell

Pour rappel : (bash = Bourne Again Shell)

Exo 2 : ecrire un programme en C qui provoque la création de 2 processus en plus du principal :

```
#include <stdio.h>  // printf()

#include <stdlib.h>  // NULL

#include <sys/wait> //waitpid()
```

```
#include <unistd.h>   //fork()



int main(int argc, char** argv){

pid_t   pid ;

pid = fork() ;


if ( pid == 0 ) {

        printf(« fils ») ;

}

Else if ( pid != 0) {

        pid =fork() ;

        waitpid(pid , &status , NULL) ; //permet d'attendre les fils avant de fermer (plus joli)

}

Return 0 ;

}  //le père cree donc deux fils.
```

Exo 3 : (Exo relou que Rachid a dû expliquer a tout le monde)

```
# !/bin/bash

#ensuite j'ai utilisé les feuilles magiques ou il y avait deux trois commandes :
```

Exo 4 : (super exo de rachid que personne n'avait fait) Ils ont été indulgent sur la notation. Donc j'avais fait un truc « normal » et j'ai remplacé les fonctions haut niveau par les bas niveau.

```
#include <stdio.h>  // printf()

#include <stdlib.h>  // NULL

#include <sys/wait> //waitpid() wait()

#include <unistd.h>   //fork() sleep()


int main(int argc, char** argv){

pid_t   pid ;

FILE* fichier = fopen(« SHELL-OUA »,  « rb+ ») ; //ouverture du fichier de Rachid en lecture binaire et ecriture a la suite du fichier. //haut niveau

char chaine[3] ;  //deux octets et le caractere de fin de chaine '\0'

pid = fork() ;


if ( pid == 0 ) {  //pid fils

chaine = 'DY' ;

        fputs ( chaine , fichier) ;  //insère dans le fichier 2 octets puisqu' un caractère est codé sur 1 octet

        sleep(1) ;

        fgets ( chaine, 2, fichier ) ;  //récupère deux 2 octets

}

Else if ( pid != 0) {

        fgets ( chaine, 2, fichier ) ;  //récupère deux 2 octets

        sleep (3) ;

        fputs ( chaine , fichier) ;  //insère dans le fichier 2 octets puisqu' un caractère est codé sur 1 //octet

        wait( WIFEXITED) ;   //renvoie si le fils se termine normalement, marche aussi sans //arguments.

//alternative :

        waitpid(pid , &status , NULL) ; //permet d'attendre les fils avant de fermer (plus joli)

        fclose ( fichier) ;
```

```
}

return 0 ;

}
```