Programmation Système - TD 3
Programmation des processus : exec

20/10/20

Remarque : *Pour compiler les exemples du cours, on devra ajouter les fichiers* include *qui sont omis sur les diapositives pour des questions de place. Il suffit d'utiliser la commande* man *pour trouver ceux qui sont nécessaires à chacun des appels systèmes ou routines du programme considéré.*

1 Exercice 1 : exec()

- Programmer l'exemple du cours *exemple_execvp.c* (diapositive 4) et le tester. Remplacer la première ligne de la fonction *main*() par :

  char * myargv[] = { "-l", "-i", (char *) NULL};

  Quelle différence observez-vous à l'exécution? Donnez votre explication sur cette différence.

- Programmer l'exemple du cours *exemple_execve.c* (diapositive 6) et le tester.

- Programmer l'exemple du cours *exemple_execlp.c* (diapositive 7) et le tester.

2 Exercice 2 : system()

- Programmer l'exemple du cours *exemple_system.c* (diapositive 9) et le tester.

- Écrire le programme *mysystem_arg.c* qui implémente une version simplifiée de la fonction *system*() et exécute avec *execv*() la commande passée en paramètre à la fonction *mysystem*(). La ligne correspondante à cet appel sera :
  execv("/bin/sh", argv);

  Consultez `man 3 system` et `man 3 execv` pour voir les paramètres et comprendre le fonctionnement de ces routines.
  Le corps de la fonction main() ressemblera à :

```
int main(int argc, char * argv[]) {
  int code = 0;
  char LIGNE[MAX_LIGNE];


  code = system("ls"); // le vrai
  printf("code retour : %d\n", code);


  code = mysystem("ls"); // pareil avec le notre
  printf("code retour : %d\n", code);

  code = mysystem("who"); // on peut faire autre chose que ls
  printf("code retour : %d\n", code);

  assert(argc > 1);
  // coller tous les arguments de la ligne de commande en un seul char *
  dans LIGNE ... à compléter

  code = mysystem(LIGNE);
  printf("code retour : %d\n", code);
  exit(EXIT_SUCCESS);
  }
```

- Tester votre programme depuis la ligne de commande avec :
  $ ./mysystem_arg ls exemple*

  puis faites le même test depuis la fonction *main*() en insérant la ligne :
  code = mysystem("ls exemple*");

  Est-ce que les deux exécutions donnent le même résultat ? Font-elles la même chose ? Donnez votre explication.

3 Exercice 3 : mini bash

Écrire le programme minibash.c qui imite le shell bash :

- Une boucle infinie propose une invite de commande ;
- La commande entrée par l'utilisateur est exécutée dans un processus fils ;
- Les commandes sont toutes exécutées au premier plan, sans redirection ou autres combinaisons.

Dans un premier temps on traitera des commandes simples sans arguments. La saisie de la ligne de commande sera réalisée avec $fgets()$ et l'exécution de cette commande dans un processus fils sera réalisée avec *execvp*().