Partiel de programmation système.

Informatique 2ème année 2009/2010.

—*Denis Barthou* - dbarthou@enseirb.fr —

La durée du partiel est de 2h. Les documents sont autorisés (polys et énoncés de TDs notamment). Lisez bien les énoncés de chaque exercice. Les exercices sont indépendants.

▸Exercice 1. *Descripteurs de fichier*

Ecrire un programme qui écrit sur la sortie standard tout le contenu d'un fichier dont le nom est passé en ligne de commande (accessible par les arguments du `main`*). Le programme écrira les caractères lus sous forme hexadécimale (utiliser le format "%x" de* `printf`*) et affichera le résultat par lignes d'au plus 80 caractères. On écrira le programme en utilisant les fonctions* `open`*,* `read`*,* `printf` *et* `close`*. On ne fera aucune gestion des erreurs et on omettra les includes. Attention, votre code ne doit pas excéder 15 lignes !*

▸Exercice 2. *Signaux*
Considérez le programme suivant:

```
#include <unistd.h>
#include <signal.h>
#include <stdio.h>
#include <stdlib.h>
int x=0;
void decr(int s) {
  x=x-1;
}
int main() {
  pid_t child,father;
  int i;
  int status;
  father=getpid();
  child=fork();
  if (child!=0) {
    signal(SIGTERM,decr);
    for (i=0; i<5; i++) x=x+1;
    waitpid(child,&status,0);
    printf("x = %d\n",x);
  } else {
    kill(father,SIGTERM);
    kill(father,SIGTERM);
  }
  return EXIT_SUCCESS;
}
```

Que fait le programme ? Quels peuvent être les valeurs possibles de `x` *dans le processus père ? Vous justifierez votre réponse.*

▸Exercice 3. *Création de processus*
Considérez le programme suivant:

```
#include <unistd.h>
#include <stdlib.h>
int n=0;
int main() {
  int status;
  pid_t child1,child2;
  child1=0;
  do {
    n++;
    child1=fork();
```