TD Programmation Système

Série 2 Signaux : Correction

Exercice 1

```
#include<stdio.h>
#include<stdlib.h>
#include<signal.h>

main(int argc,char *argv[])
{
      int pid,sig;

      pid=atoi(argv[1]);
      sig=atoi(argv[2]);
      kill (pid,sig);

}
```

Exercice 2

On vous demande d'écrire deux programmes

-"Boucle.c" qui redéfini le signal passé en paramètre et qui boucle après avoir afficher son pid

-"Envoi.c" qui récupère un pid et un signal passés en paramètre et qui envoi le signal au processus identifié par pid

```
#include<stdio.h>
#include<stdlib.h>
#include<signal.h>

void afficher (int sig)
      {
printf("numero du signal : %d\n", sig);
      }


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

        int sig, pid;

        sig=atoi(argv[1]);
        signal(sig,afficher);
         printf("pid=%d\n", getpid());
         while(1);

`
```

Boucle.c

```
#include<stdlib.h>
#include <sys/types.h>
#include <signal.h>



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

       int sig, pid;

          sig=atoi(argv[1]);
          pid=atoi(argv[2]);
          kill(pid,sig);


}
```

Envoi.c

Exercice 3

```
#include<stdio.h>
#include<stdlib.h>
#include<signal.h>
#include <unistd.h>


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

        int NB;

       NB=atoi(argv[1]);
       /*on ignore le signal SIGINT(ctr^c)
        signal(SIGINT,SIG_IGN);
       sleep(NB);

      /* On remet le comportement par Défaut de SIGINT */
       signal(SIGINT,SIG_DFL);
}
```

Exercice 4 :

```
#include<stdio.h>
#include<stdlib.h>
#include<signal.h>

void afficher (int sig){
printf("numero du signal : %d\n", sig);
}
main(int argc,char *argv[]){
      int i;

      /* on redéfinie le comportement des 65 signaux définis sous UNIX*/
      for(i=1;i<65;i++){
           signal(i,afficher);

         /* on exclut les signaux 9 et 19 plus le 32 et le 33*/
          if(i==9||i==19||i==32||i==33){
            switch (i){
            case 9 :printf("on peut pas definir le signlal 9 \n");break;
            case 19:printf("on peut pas definir le signlal 19 \n");break;
            case 32:printf("on peut pas definir le signlal 32 \n");break;
            case 33:printf("on peut pas definir le signlal 33 \n");break;
                      }
            }
        else{
            kill(getpid(),i);
            }     }
```

Exercice 5 :

```
#include<stdio.h>
#include<stdlib.h>
#include <unistd.h>
#include<signal.h>

void Handler(int sig)
{
/* handler sans comportement */
}

main(){
int pid,i=0;

signal(10,Handler);/* Redéfinition du Signal 10 : SIGUSR1*/
pid=fork();

if(pid==-1){
            perror("fork");
            exit(0);
            }
while(i<99){
if (pid==0){
                  printf("fils:");
                   do { i=i+2;
                        printf(" %d",i);

                      }while (i%5!=0);
                    printf("\n");
                    kill(getppid(),10);
                    pause();
      }
else{

                    printf("pere:");
                   do { i=i+3;
                        printf(" %d",i);

                      }while (i%5!=0 && i<99);
                      printf("\n");
                      kill(pid,10);

  }

}

}
```