Horloge

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <signal.h>
#include <sys/types.h>
#include <wait.h>
int SEC=0;
int MIN=0;
int HEU=0;

void incr(){SEC++;}
void incr_m(){MIN++;}
void incr_h(){HEU++;}

void minute(pid_t fils){
    signal(SIGUSR1,incr_m);
    printf("Minute de PID : %d et de père : %d\n",getpid(),getppid());
    while (1)
    {
        pause();
        printf("Minutes+1 et m=%d\n",MIN);
        kill(fils,SIGUSR1);
        if(MIN==5){MIN=-1;kill(getppid(),SIGUSR1);pause();}
    }
    exit(0);
}
void seconde(){
    signal(SIGALRM,incr);
    signal(SIGUSR1,incr);
    printf("Seconde de PID : %d et de père : %d\n",getpid(),getppid());
    while(1){
    //sleep(1);
    alarm(1);
    pause();
    printf("s=%d\n",SEC);
    if(SEC==5){SEC=-1;kill(getppid(),SIGUSR1);pause();}
    }
    exit(0);
}
void heure(pid_t fils){
    signal(SIGUSR1,incr_h);
    printf("Heure de PID : %d et de père : %d\n",getpid(),getppid());
    while (1)
    {
        pause();
        printf("Heures+1 et h=%d\n",HEU);
        kill(fils,SIGUSR1);
    }
```

```
    exit(0);
}


int main(int argc, char const *argv[])
{
    pid_t minutes=-1;
    pid_t secondes=-1;
    minutes=fork();
    if(minutes==0){//dans minutes
        secondes=fork();
        if(secondes!=0)minute(secondes);//on exec minutes que si l'on est pas dans
secondes
    }
    else if(minutes==-1){printf("ERROR FORK\n");}
    else{//dans heure
        heure(minutes);
    }
    seconde();
}
```

Tube anonyme

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/wait.h>

// Écrire un programme qui prend en argument deux commandes (commande1 et
commande2) et réalise une exécution équivalente à l'exécution de « commande1 |
commande2 » dans un shell.

int main(int argc, char** argv){
    if (argc != 3) {
        printf("Usage : %s commande1 commande2\n", argv[0]);
        exit(1);
    }

    // Création du tube
    int tube[2];
    pipe(tube);

    if (fork() == 0) {
        // Fermeture du tube en lecture
        close(tube[0]);
        // Redirection de la sortie standard vers le tube
        dup2(tube[1], 1);
        // Fermeture du tube en écriture
        close(tube[1]);
```

```
        // Exécution de la commande 1
        execlp(argv[1], argv[1], NULL);
        exit(0);
    }

    if (fork() == 0) {
        // Fermeture du tube en écriture
        close(tube[1]);
        // Redirection de l'entrée standard vers le tube
        dup2(tube[0], 0);
        // Fermeture du tube en lecture
        close(tube[0]);
        // Exécution de la commande 2
        execlp(argv[2], argv[2], NULL);
        exit(0);
    }

    // Fermeture du tube en lecture
    close(tube[0]);
    // Fermeture du tube en écriture
    close(tube[1]);
    // Attente de la fin des fils
    wait(NULL);
    wait(NULL);
    exit(0);
}
```