Vincenzo Loris Proto Matricola 0512103715

#include<stdio.h>

#include<stdlib.h>

#include<sys/types.h>

#include<unistd.h>

#include<sys/wait.h>

#include<signal.h>

void mull(int sd){}

int fatt(int n){

if(n==0) return 1;

return n\*fatt(n-1);

}

void handler\_1(int sig){

printf("Ricevuto segnale: %d\n",sig);

printf("File nella directory sono : \n");

execl("/bin/ls","ls","-l",NULL);

}

void handler\_2(int sig){

char answer[512];

printf("\n\n %s%d \n\n %s","Ricevuto segnale di interruzione, segnale = ",sig,

"vuoi continuare (c) o uscire (q) =");

scanf("%s",answer);

if(\*answer=='c')

signal(SIGINT,handler\_2);

else{

printf("Processo terminato dall'utente \n\n");

exit(1);

}

}

int fib(int n){

if(n<=1) return n;

return (fib(n-1) + fib(n-2));

}

int main(void){

int pid,pid2,status;

pid=fork();

if(pid<0)

{

printf("Errore");

exit(1);

}

if(pid==0){ //figlio uno

if(signal(SIGINT,handler\_1)==SIG\_ERR){

printf("Errore assegnazione sigusr1\n");

exit(1);

}

for(int i=0;i<40;i++)

printf("fib(%d) : %d\n", i,fib(i));

sleep(3);

}

else{ //padre

pid2=fork();

if (pid2<0){

printf("Errore");

exit(1);

}

if(pid2==0){//figlio due

if(signal(SIGINT,handler\_2)==SIG\_ERR){

printf("Errore assegnazione sigusr1\n");

exit(1);

}

for(int i=0; i<40;i++){

printf("fatt(%d) : %d\n", i,fatt(i));

sleep(1);

}

}

else{//padre

if(signal(SIGINT,mull)==SIG\_ERR){

printf("Errore assegnazione sigusr1\n");

exit(1);

}

printf("I miei figli %d e %d sono terminati \n",pid,pid2);

//printf("Processo %d terminato con status %d\n",wait(&status),status);

printf("Processo %d terminato con status %d\n",waitpid(pid,&status,0),status);

//printf("Processo %d terminato con status %d\n",wait(&status),status);

printf("Processo %d terminato con status %d\n",waitpid(pid2,&status,0),status);

}

}

}

Ho aggiunto le due waitpid() necessarie a sospendere il processo padre al fine di far terminare prima i processi figli.