#include <stdio.h>

#include <stdlib.h>

#include <wait.h>

#include <sys/types.h>

#include <signal.h>

#include <unistd.h>

#include <ctype.h>

#include <fcntl.h>

#define SIZE 8000

int main() {

int F, fd1, fd2, i,n;

char buff[SIZE];

F = fork();

if (F < 0 ) {

printf("Errore creazione figlio \n");

exit(1); }

if (F==0){ //processo figlio

signal(SIGINT, SIG\_IGN);

if ((fd1 = open("PIPPO.txt", O\_RDONLY , NULL)) == -1) {

printf("Errore apertura file\n");

exit(1); }

if ((fd2 = open("TOPOLINO.txt", O\_WRONLY | O\_EXCL | O\_CREAT , 0600)) == -1) {

printf("Errore apertura file\n");

exit(1); }

int cont = 0;

while ((n = read(fd1, buff , SIZE)) > 0 ) {

for(i = 0; i<n; i++) {

if(buff [i] == 'a' || buff[i] == 'e' || buff [i] == 'i'|| buff [i] == 'o'|| buff [i] == 'u' ||

buff [i] == 'A' || buff[i] == 'E' || buff [i] == 'I'|| buff [i] == 'O'|| buff [i] == 'U') {

buff[i] = 'X';

cont++; }

}

if(write(fd2, buff, n) != n ) {

printf("Errore scrittura file \n");

exit(1);

}

}

printf("Numero cambiamenti = %d \n", cont);

close(fd1);

close(fd2);

}

else {

signal(SIGINT, SIG\_IGN);

wait(NULL);

printf("processo padre %d : mio figlio è terminato. Pid = %d \n",getpid(), F);

exit(0); }

return 0; }

Signal necessaria per ignorare il segnale ctlr+c.

Wait necessaria nel processo padre per attendere che il figlio termini(nell’elaborato l’ho inserita nel processo figlio).