Signals

LAB # 11

Fall 2023

CSE-302L Systems Programming Lab

Submitted by: **Ali Asghar**

Registration No.: **21PWCSE2059**

Class Section: **C**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Submitted to:

Engr. Abdullah Hamid

Date:

1st February 2024

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

Task 1:

Implement wait ( ) function

a. By changing the default behavior of SIGCHLD (without using pause or sigsuspend or sigwait)

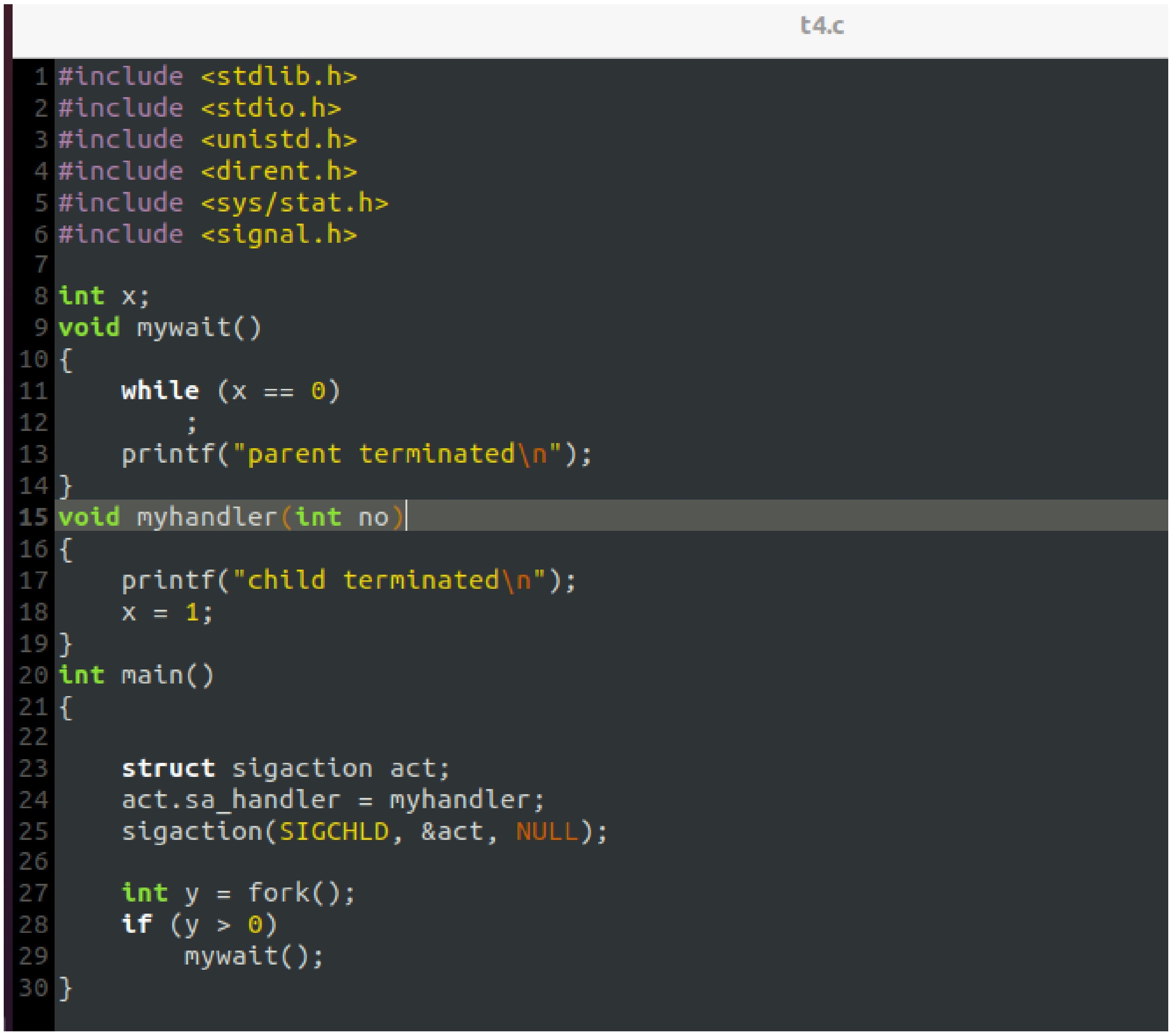
b. Using pause () function

c. Using signal suspend option

d. Using sigwait

Part a:

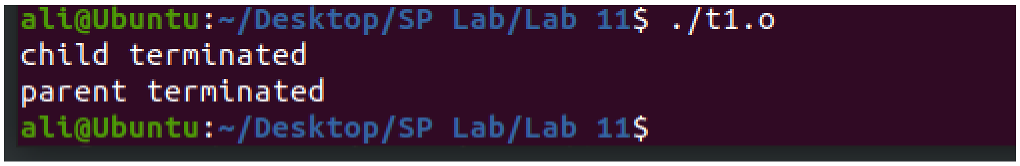
Code:



t4.c

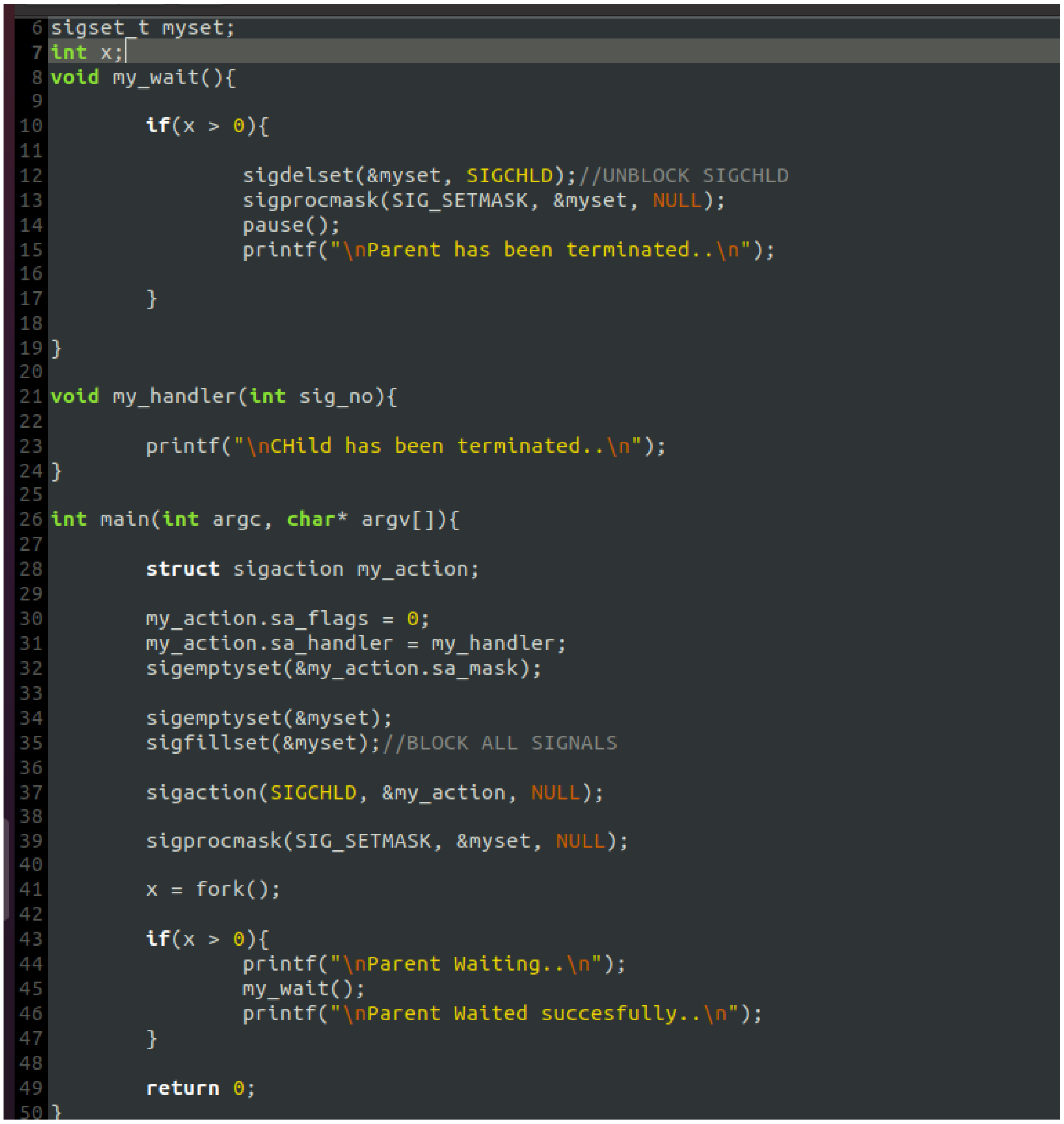
```
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
#include <dirent.h>
#include <sys/stat.h>
#include <signal.h>

int x;
void mywait()
{
    while (x == 0)
        ;
    printf("parent terminated\n");
}
void myhandler(int no)
{
    printf("child terminated\n");
    x = 1;
}
int main()
{

    struct sigaction act;
    act.sa_handler = myhandler;
    sigaction(SIGCHLD, &act, NULL);

    int y = fork();
    if (y > 0)
        mywait();
}
```

Output:

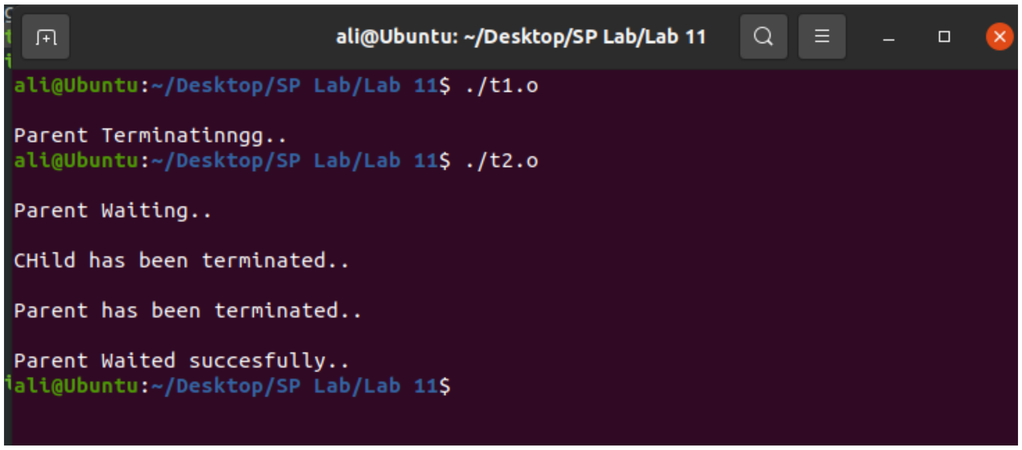

```
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$ ./t1.o
child terminated
parent terminated
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$
```

Part b:

Code:


```c
sigset_t myset;
int x;
void my_wait(){

        if(x > 0){

                sigdelset(&myset, SIGCHLD);//UNBLOCK SIGCHLD
                sigprocmask(SIG_SETMASK, &myset, NULL);
                pause();
                printf("\nParent has been terminated..\n");

        }

}

void my_handler(int sig_no){

        printf("\nCHild has been terminated..\n");
}

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

        struct sigaction my_action;

        my_action.sa_flags = 0;
        my_action.sa_handler = my_handler;
        sigemptyset(&my_action.sa_mask);

        sigemptyset(&myset);
        sigfillset(&myset);//BLOCK ALL SIGNALS

        sigaction(SIGCHLD, &my_action, NULL);

        sigprocmask(SIG_SETMASK, &myset, NULL);

        x = fork();

        if(x > 0){
                printf("\nParent Waiting..\n");
                my_wait();
                printf("\nParent Waited succesfully..\n");
        }

        return 0;
}
```

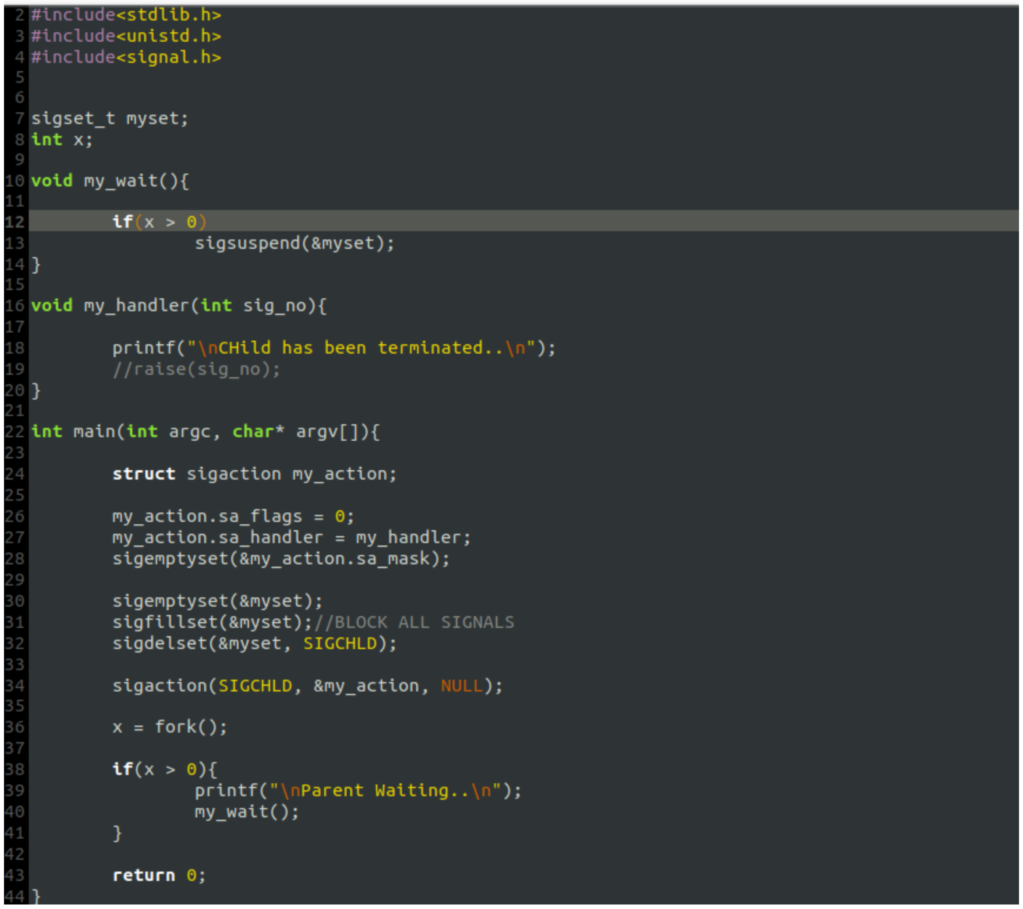
Output:



```
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$ ./t1.o

Parent Terminatinngg..
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$ ./t2.o

Parent Waiting..

CHild has been terminated..

Parent has been terminated..

Parent Waited succesfully..
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$
```

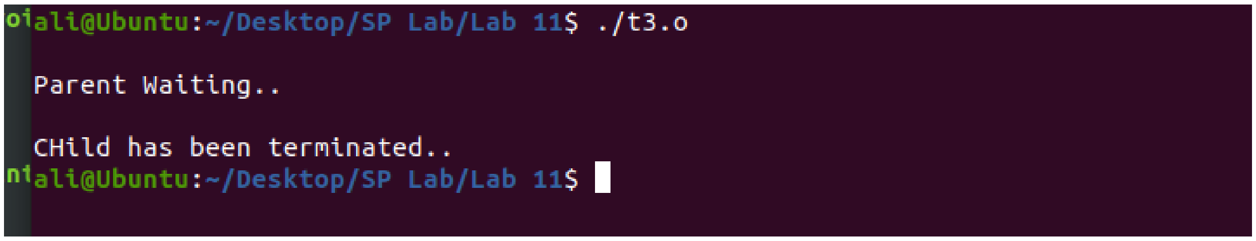
Part c:

Code:



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


sigset_t myset;
int x;

void my_wait(){

	if(x > 0)
		sigsuspend(&myset);
}

void my_handler(int sig_no){

	printf("\nCHild has been terminated..\n");
	//raise(sig_no);
}

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

	struct sigaction my_action;

	my_action.sa_flags = 0;
	my_action.sa_handler = my_handler;
	sigemptyset(&my_action.sa_mask);

	sigemptyset(&myset);
	sigfillset(&myset);//BLOCK ALL SIGNALS
	sigdelset(&myset, SIGCHLD);

	sigaction(SIGCHLD, &my_action, NULL);

	x = fork();

	if(x > 0){
		printf("\nParent Waiting..\n");
		my_wait();
	}

	return 0;
}
```

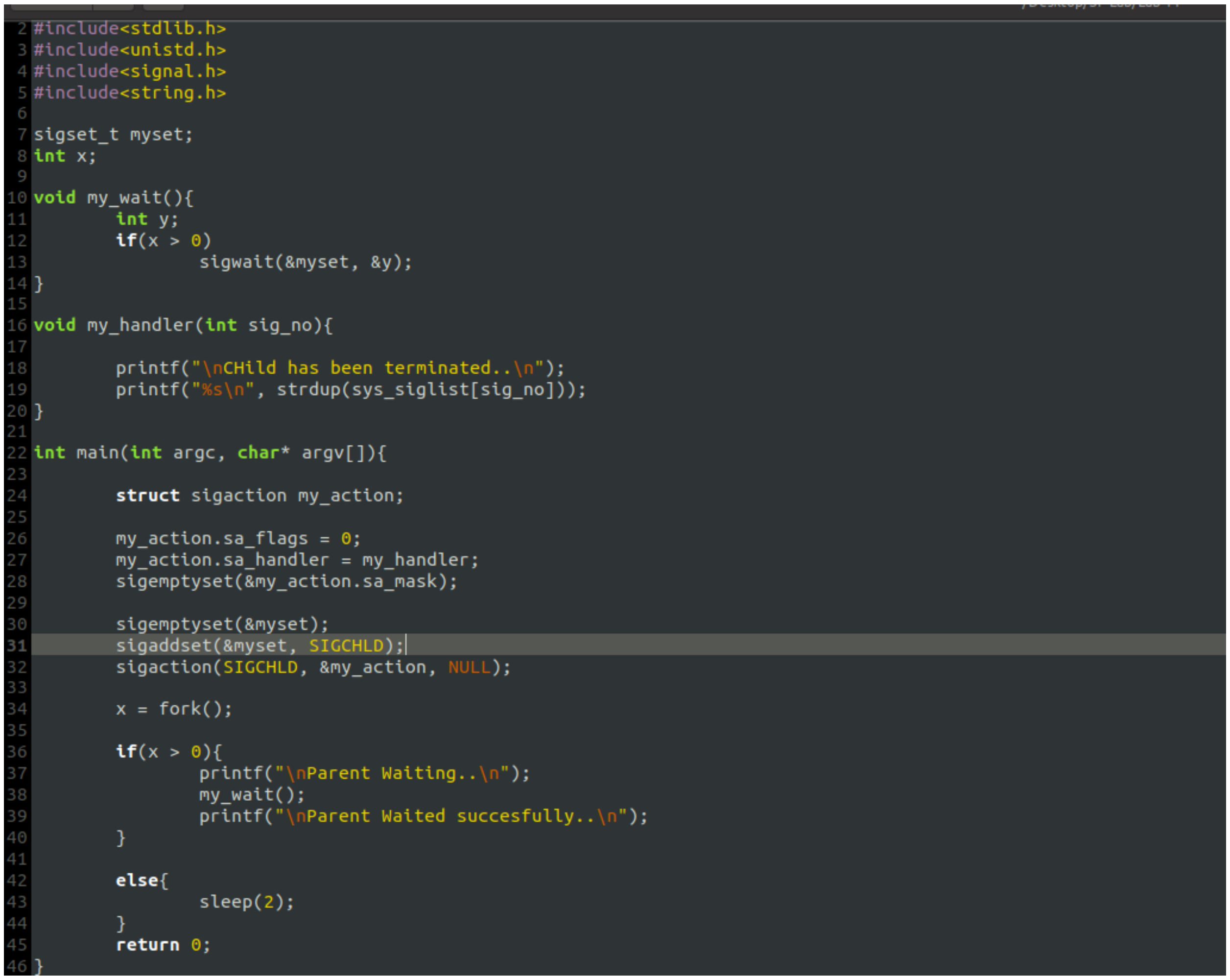
Output:



```
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$ ./t3.o

Parent Waiting..

CHild has been terminated..
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$
```

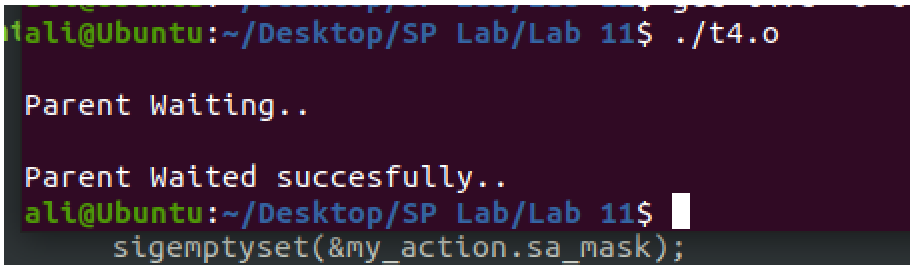
Part d:

Code:



```c
#include<stdlib.h>
#include<unistd.h>
#include<signal.h>
#include<string.h>

sigset_t myset;
int x;

void my_wait(){
        int y;
        if(x > 0)
                sigwait(&myset, &y);
}

void my_handler(int sig_no){

        printf("\nCHild has been terminated..\n");
        printf("%s\n", strdup(sys_siglist[sig_no]));
}

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

        struct sigaction my_action;

        my_action.sa_flags = 0;
        my_action.sa_handler = my_handler;
        sigemptyset(&my_action.sa_mask);

        sigemptyset(&myset);
        sigaddset(&myset, SIGCHLD);
        sigaction(SIGCHLD, &my_action, NULL);

        x = fork();

        if(x > 0){
                printf("\nParent Waiting..\n");
                my_wait();
                printf("\nParent Waited succesfully..\n");
        }

        else{
                sleep(2);
        }
        return 0;
}
```

Output:



```
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$ ./t4.o

Parent Waiting..

Parent Waited succesfully..
ali@Ubuntu:~/Desktop/SP Lab/Lab 11$
    sigemptyset(&my_action.sa_mask);
```