Министерство образования Республики Беларусь
Учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа №6
по дисциплине: **ОСИСП**
Тема: Средства межпроцессного взаимодействия

Выполнил
студент 2 курса
Корнасевич И. Д.

Проверил
Давидюк Ю. И.

Брест 2021

Задание

- Программные каналы
- Родитель передает потомку три стороны треугольника, потомок возвращает его периметр

Написать программу, которая порождает дочерний процесс, и общается с ним через средства взаимодействия согласно варианту, передавая и получая информацию согласно варианту. Передачу и получение информации каждым из процессов сопровождать выводом на экран информации типа "процесс такой-то передал/получил такую-то информацию". Сообщение вводит пользователь через терминал. Дочерние процессы начинают операции после получения сигнала SIGUSR1 от родительского процесса. После отработки дочерний процесс должен возвращать результат родительскому процессу!

main.c

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

void sig_handler_parent(int signum) {
    printf("Parent : Received a response signal from child \n");
}

void sig_handler_child(int signum) {
    printf("Child : Received a signal from parent \n");
}

int main() {
    int fd[2];

    if(pipe(fd) < 0){
        printf("Can\'t create pipe\n");
        return -1;
    }

    pid_t pid;
    if ((pid = fork()) < 0) {
        printf("Fork Failed\n");
        exit(1);
    }
        /* Child Process */
    else if (pid == 0) {
        signal(SIGUSR1, sig_handler_child);
        pause();
        double sides[3];
        read(fd[0], sides, 3*sizeof(double));
        close(fd[0]);
        sides[0] += sides[1] + sides[2];
        write(fd[1], sides, sizeof(double));
        close(fd[1]);
        printf("Child: sending signal to Parent\n");
```

```
38          kill(getppid(), SIGUSR1);
39      }
40          /* Parent Process */
41      else {
42          signal(SIGUSR1, sig_handler_parent);
43          double sides[3] = {100, 200, 150};
44          write(fd[1], sides, 3*sizeof(double));
45          close(fd[1]);
46          printf("Parent: sending signal to Child\n");
47          kill(pid, SIGUSR1);
48          printf("Parent: waiting for response\n");
49          pause();
50          double side[1];
51          read(fd[0], side, sizeof(double));
52          close(fd[0]);
53          printf("Parent: perimeter is: %f", side[0]);
54      }
55      return 0;
56  }
```

results

```
1  Parent: sending signal to Child
2  Parent: waiting for response
3  Child : Received a signal from parent
4  Child: sending signal to Parent
5  Parent : Received a response signal from child
6  Parent: perimeter is: 450.000000
7  Process finished with exit code 0
```

Оказалось очень интересно, что получение сигнала прерывает pause().

Вывод: Я использовал pipe для передачи информации между родственными процессами, также давал команду процессам для начала работы используя сигналы и создал собственный обработчик сигналов.