Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №2 по курсу**

**«Операционные системы»**

**Управление процессами и каналами**

Студент: Камеш Михаил Алексеевич

Группа: М8О–207Б–18

Вариант: 28

Преподаватель: Соколов Андрей Алексеевич

Оценка: \_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_

Москва, 2020.

**Содержание**

1. Постановка задачи
2. Общие сведения о программе
3. Общий метод и алгоритм решения
4. Основные файлы программы
5. Примеры работы
6. Вывод

Цель работы

Приобретение практических навыков в:

• Управление процессами в ОС

• Обеспечение обмена данных между процессами посредством каналов

Задание

Составить и отладить программу на языке Си, осуществляющую работу с процессами и взаимодействие между ними в одной из двух операционных систем. В результате работы программа (основной процесс) должен создать для решение задачи один или несколько дочерних процессов. Взаимодействие между процессами осуществляется через системныесигналы/события и/или каналы (pipe).

Необходимо обрабатывать системные ошибки, которые могут возникнуть в результате работы.

Вариант задания.

Родительский процесс считывает две координаты передает их через канал дочернему процессу.

Дочерний процесс определяет к какой четверти относится точка, а далее передает результат родительскому процессу

Общие сведения о программе

Программа компилируется из одного файла main.c. В программе используются следующие системные вызовы:

1. read – для чтения данных из файла

2. write – для записи данных в файл

3. pipe – для создания однонаправленного канала, через который могут общаться два процесса. Системный вызов возвращает два дескриптора файлов. Один для чтения из канала, другой для записи в канал.

4. fork– для создания дочернего процесса.

5. close – для закрытия файла.

Общий метод и алгоритм решения.

Для реализации поставленной задачи необходимо:

1. Используя системный вызов pipe создать 2 канала, по которым будут обмениваться данными два процесса.

2. Используя системный вызов fork создать дочерний процесс.

3. В родительском процессе считывать данные cо стандартного потока

4. Передать данный в дочерний процесс командой write

5. Считать данные командой read, произвести необходимые вычисления, передать данные обратно

6. Считать обработанные данные и вывести на экран

Файлы программы:

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<sys/types.h>

#include<string.h>

#include<sys/wait.h>

#include <stdbool.h>

int main()

{

int fd1[2];

int fd2[2];

int fd3[2];

int firstcoordinate;

int secondcoordinate;

pid\_t p;

if (pipe(fd1)==-1)

{

fprintf(stderr, "Pipe Failed" );

return 1;

}

if (pipe(fd2)==-1)

{

fprintf(stderr, "Pipe Failed" );

return 1;

}

p = fork();

if (p < 0)

{

fprintf(stderr, "fork Failed" );

return 1;

}

else if (p > 0)

{

if (scanf("%d %d", &firstcoordinate, &secondcoordinate)!=2)

{

printf("Failed to read, exiting\n");

return 1;

}

int result;

close(fd1[0]);

close(fd3[0]);

write(fd1[1], &firstcoordinate, 100);

close(fd1[1]);

write(fd3[1], &secondcoordinate, 100);

close(fd3[1]);

wait(NULL);

close(fd2[1]);

close(fd3[1]);

read(fd2[0], &result, 100);

printf("Point lies in quarter %d\n", result);

close(fd2[0]);

}

else

{

close(fd1[1]);

close(fd3[1]);

bool i = true;

int calcresult;

int result = 0;

read(fd1[0], &calcresult, 10);

if (calcresult < 0)

{i = false;}

read(fd3[0], &result, 10);

if (result < 0 && i == false)

{result = 3;}

else if (result < 0 && i == true)

{result = 4;}

else if (result >= 0 && i == true)

{result = 1;}

else if (result >= 0 && i == false)

{result = 2;}

close(fd1[0]);

close(fd3[0]);

close(fd2[0]);

write(fd2[1], &result, 100);

close(fd2[1]);

exit(0);

}

}

Демонстрация работы программы.

1 4

Point lies in quarter 1

1 -4

Point lies in quarter 4

-1 4

Point lies in quarter 2

-1 -4

Point lies in quarter 3

Выводы:

В процессе проведения работы были изучены процессы в операционных систем и то, как они работают. Были изучены каналы и способы изпользования их для передачи информации

между двумя процессами, связанными между собой форком. Для передачи информации между процессами использовались одновременно два канала - для одновременного хранения двух

переменных (по одному в канале). Возможно, можно было передать переменные с помощью только одного канала, но такой выход показался мне более простым и наглядным.