Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Лабораторная работа №4 по курсу**

**«Операционные системы»**

**Управление процессами с помощью mmap**

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 файл мапинга

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

Использовать mmap.

Общие сведения о программе

Программа компилируется из одного файла main.c. В данном файле используются заголовочные файлы stdio.h, unistd.h, string.h, stdlib.h. В программе используются следующие системные вызовы:

1. mmap - создает новое отображение в виртуальном адресном пространстве вызывающего процесса.

2. close – для закрытия файла.

3. fork– для создания дочернего процесса.

4. sleep - переводит в неактивное состояние на определенный период времени.

Общий метод и алгоритм решения.

Для реализации поставленной задачи необходимо:

2. Используя системный вызов fork создать дочерний процесс.

3. В родительском процессе считывать данные cо стандартного потока

4. Передать данный в дочерний процесс командой mmap

5. Произвести необходимые вычисления, передать данные обратно командой mmap

6. Считать обработанные данные и вывести на экран

Файлы

#include <stdio.h>

#include <stdlib.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <fcntl.h>

#include <unistd.h>

#include <sys/mman.h>

#include <string.h>

#include <sys/wait.h>

#include <string.h>

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

{

int fd;

struct stat mystat;

char \*pmap;

pid\_t p;

fd = open("test.txt", O\_RDWR);

if (fd == -1)

{

perror("open");

exit(1);

}

if(fstat(fd, &mystat) < 0)

{

perror("fstat");

close(fd);

exit(1);

}

pmap = mmap(0, mystat.st\_size, PROT\_READ | PROT\_WRITE, MAP\_SHARED, fd, 0);

if(pmap == MAP\_FAILED)

{

perror("mmap");

close(fd);

exit(1);

}

p = fork();

if (p < 0)

{

fprintf(stderr, "fork Failed" );

return 1;

}

else if (p > 0)

{

char a[256] = "1";

char b[256] = "1";

scanf("%s %s", a, b);

char\* con = (char\*)malloc(strlen(a)+strlen(b)+1);

for (int i = 0; i < strlen(a); i++)

{

con[i]=a[i];

}

con[strlen(a)]='#';

for (int i = 0; i < strlen(b); i++)

{

con[strlen(a)+i+1]=b[i];

}

con[strlen(a)+strlen(b)+1]='#';

for (int i = 0; i < strlen(con); i++)

{

pmap[i] = con[i];

}

wait(NULL);

printf ("Point lies in %c coordinate square\n", pmap[0]);

}

else

{

sleep(4);

char a[256] = "1";

char b[256] = "1";

char coord = 'a';

int c = 0;

for (int i = 0; i < strlen(pmap); i++)

{

if (pmap[i] != '#') {

a[i] = pmap [i];

c++;

}

else {

break;

}

}

for (int i = 0; i < strlen(pmap)-c-1; i++)

{

if (pmap[i+c+1] != '#') {

b[i] = pmap[i+c+1];

}

else {

break;

}

}

int firstcoordinate = atoi(a);

int secondcoordinate = atoi(b);

if (firstcoordinate >= 0 && secondcoordinate >= 0) {

coord = '1';

}

if (firstcoordinate < 0 && secondcoordinate < 0) {

coord = '3';

}

if (firstcoordinate >= 0 && secondcoordinate < 0) {

coord = '4';

}

if (firstcoordinate < 0 && secondcoordinate >= 0) {

coord = '2';

}

pmap[0] = coord;

}

close(fd);

return 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 использовалась команда mmap. Задание было аналогично задаче 2, однако

вместо каналов использовалась команда mmap. Проблемы возникли при считке информации из буфера после маппинга, потому что переменные были разделены специальным символом,

и возникали большие проблемы и индексами.