

РГПУ имени А.И. Герцена

Институт Компьютерных Наук и Технологического Образования

Информатика и вычислительная техника

Работу выполнил Цирулик И.А.

Лабораторная работа №4.

Практическое знакомство с процессами, передачей данных между процессами и их синхронизацией

Цель работы: Практическое знакомство с объектом процесс, основными механизмами передачи данных между процессами, а также синхронизацией взаимодействующих процессов в ОС Unix.

Задание: Изучить базовые возможности оболочки bash ОС Unix по управлению процессами (заданиями). Разработать приложения, реализующие схему «клиент-сервер» с использованием средств межпроцессорного взаимодействия: семафоров, разделяемой памяти, программных каналов и одной очереди сообщений.

Задание 1.

Программа-клиент выводит на экран текст, который посылает программа-сервер. По слову ex обе программы прекращают работать. Программа основана на очереди сообщений FIFO.

```
task1-1.c - KWrite
Файл Правка Вид Закладки Сервис Настройка Справка
#include <unistd.h>
#include <stdio.h>
#include <error.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <iostream>
#include <cstring>
#include <fstream>
#include <sys/stat.h>
#include <errno.h>
#include <fcntl.h>
#include <stdlib.h>
using namespace std;
#define MAXLINE 128
#define FILE_MODE (S_IRUSR|S_IWUSR|S_IRGRP|S_IROTH)
#define FIF01 "fifo.1"
#define FIF02 "fifo.2"
int main( int argc, char **argv)
{
    int readfd = -1, writefd = -1;
    size_t n = 0;
```

```
task1-1.c - KWrite
Файл Правка Вид Закладки Сервис Настройка Справка
#define FIF02 "fifo.2"
int main( int argc, char **argv)
{
    int readfd = -1, writefd = -1;
    size_t n = 0;
    char str[MAXLINE];
    char str2[MAXLINE];
    cout<<"Server is on"<<endl;
    unlink(FIF01);
    unlink(FIF02);

    if (mkfifo(FIF01, FILE_MODE) == EEXIST)cout<<"\n Pipes is exist"<<endl;
    if (mkfifo(FIF02, FILE_MODE) == EEXIST)cout<<"\n Pipes is exist"<<endl;
    cout<<"Text:"<<endl;
    writefd = open(FIF02, O_WRONLY);
    if ((writefd!=-1)){
        while (1){
            cin>>str;
            write(writefd, str, strlen(str));
            readfd = open(FIF01, O_RDONLY);
            while ((n = read(readfd, str2, MAXLINE))>0) {
```

```
task1-1.c - KWrite
Файл Правка Вид Закладки Сервис Настройка Справка
writefd = open(FIFO2, O_WRONLY);
if ((writefd != -1)) {
    while (1) {
        cin >> str;
        write(writefd, str, strlen(str));
        readfd = open(FIFO1, O_RDONLY);
        while ((n = read(readfd, str2, MAXLINE)) > 0) {
            str2[n] = 0;
            cout << str2 << endl;
            break;
        }
        if (strcmp(str, "ex") == 0) break;
    }
    close(readfd);
    close(writefd);
    unlink(FIFO1);
    unlink(FIFO2);
    cout << "Server is off" << endl;
} else cout << "error" << endl;
return 0; }
```

```
task1-2.c - KWrite
Файл Правка Вид Закладки Сервис Настройка Справка
#include <unistd.h>
#include <stdio.h>
#include <error.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <iostream>
#include <cstring>
#include <fstream>
#include <sys/stat.h>
#include <errno.h>
#include <fcntl.h>
#include <stdlib.h>
using namespace std;
#define MAXLINE 128
#define FILE_MODE (S_IRUSR|S_IWUSR|S_IRGRP|S_IROTH)
#define FIFO1 "fifo.1"
#define FIFO2 "fifo.2"
int main( int argc, char **argv)
{
    int readfd = -1, writefd = -1;
    size_t n = 0;
```

```
task1-2.c - KWrite
Файл Правка Вид Закладки Сервис Настройка Справка
#define FIF02 "fifo.2"
int main( int argc, char **argv)
{
    int readfd = -1, writefd = -1;
    size_t n = 0;
    char str[MAXLINE];
    char str2[MAXLINE];
    cout<<"Client is on"<<endl;
    while (1)
    {
        readfd = open(FIF02, O_RDONLY, 0);
        if (readfd!=-1)
        {
            while ((n=read(readfd, str, MAXLINE))>0)
            {
                str[n] = 0;
                cout<<str<<endl;
                break;
            }
            strcpy(str2, "Client's confirmation");
            writefd = open(FIF01, O_WRONLY, 0);

```

```
task1-2.c - KWrite
Файл Правка Вид Закладки Сервис Настройка Справка
        readfd = open(FIF02, O_RDONLY, 0);
        if (readfd!=-1)
        {
            while ((n=read(readfd, str, MAXLINE))>0)
            {
                str[n] = 0;
                cout<<str<<endl;
                break;
            }
            strcpy(str2, "Client's confirmation");
            writefd = open(FIF01, O_WRONLY, 0);
            write(writefd, str2, strlen(str2));
            close(readfd);
            close(writefd);
        }
        sleep(1);
        if (strcmp(str, "ex") ==0) break;
    }
    cout<<"Client is off";
    return 0;
}
```

Результат сервера:

Server is on

Text:

test1

Client's confirmation

test2

Client's confirmation

test3

Client's confirmation

ex

Client's confirmation

Server is off

Результат клиента:

Client is on

test1

test2

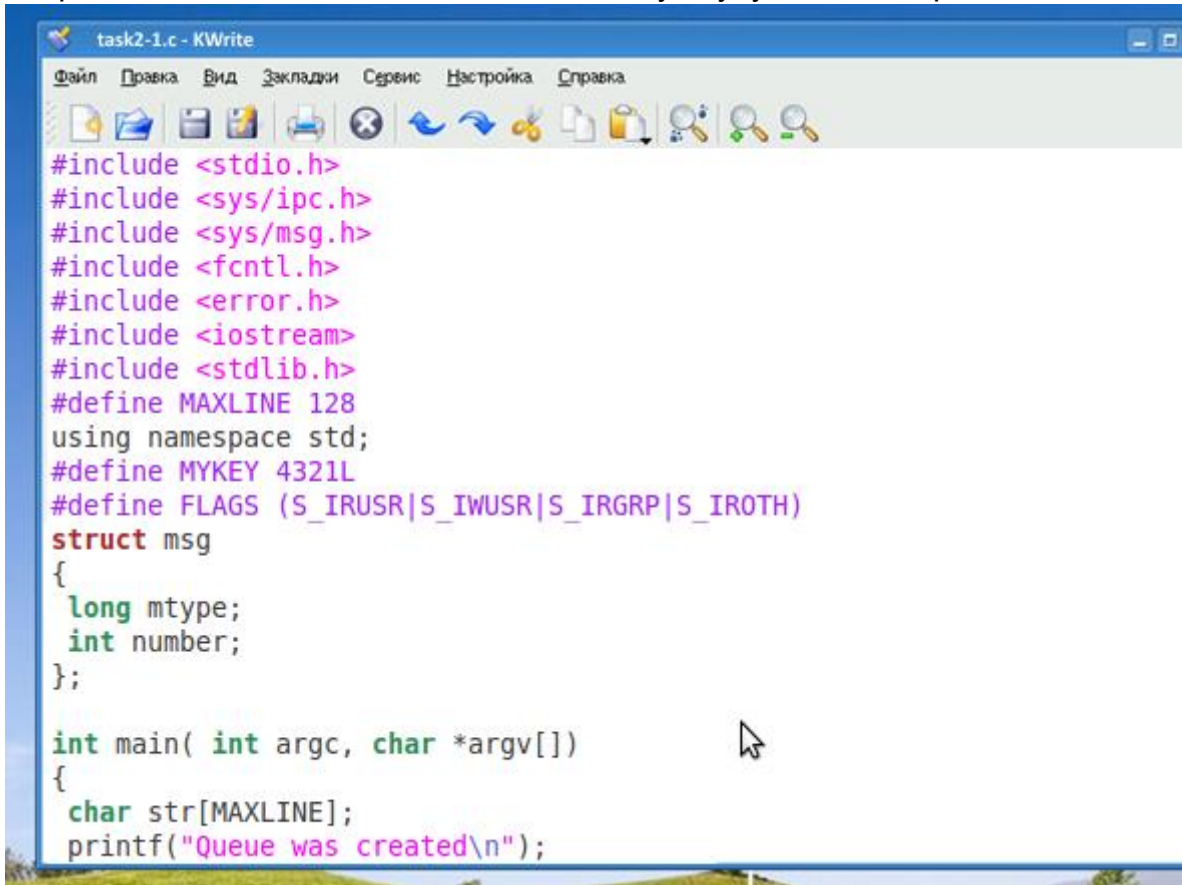
test3

ex

Client is off

Задание 2.

Программа-сервер записывает в очередь числа, программа-клиент считывает из очереди эти числа, выводит на дисплей и сумму, удаляет очередь



```
task2-1.c - KWrite
Файл  Правка  Вид  Закладки  Сервис  Настройка  Справка

#include <stdio.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <fcntl.h>
#include <error.h>
#include <iostream>
#include <stdlib.h>
#define MAXLINE 128
using namespace std;
#define MYKEY 4321L
#define FLAGS (S_IRUSR|S_IWUSR|S_IRGRP|S_IROTH)
struct msg
{
    long mtype;
    int number;
};

int main( int argc, char *argv[])
{
    char str[MAXLINE];
    printf("Queue was created\n");
```



```
task2-1.c - KWrite
Файл Правка Вид Закладки Сервис Настройка Справка
[Icons]

int main( int argc, char *argv[])
{
    char str[MAXLINE];
    printf("Queue was created\n");

    int msgid = msgget( ftok("maxim",0), FLAGS|IPC_CREAT);
    cout<< msgid<<endl;

    struct msg s_msg;
    s_msg.mtype = 1;
    printf("Data sending\n");
    for (int i=1; i<10; i++)
    {
        s_msg.number = i;
        msgsnd( msgid, &s_msg, sizeof(s_msg), IPC_NOWAIT);
    }
    printf("Server is off");
    return 0;
}
```

```
task2-2.c - KWrite
Файл Правка Вид Закладки Сервис Настройка Справка
[Icons]

#include <stdio.h>
#include <sys/ipc.h>
#include <sys/msg.h>
#include <fcntl.h>
#include <error.h>
#include <iostream>
#include <stdlib.h>
using namespace std;
#define MYKEY 4321L
#define FLAGS (S_IRUSR|S_IWUSR|S_IRGRP|S_IROTH)
struct msg
{
    long mtype;
    int number;
};

int main( int argc, char *argv[])
{
    printf("Queue was opened\n");
    int msgid = msgget(ftok("maxim", 0), FLAGS);
    cout<<msgid<<endl;
```

Результат сервера:

Queue was created

65536

Data sending

Server is off

Результат клиента:

Queue was opened

65536

Number: 1

Number: 2

Number: 3

Number: 4

Number: 5

Number: 6

Number: 7

Number: 8

Number: 9

Sum: 45

Queue was deleted

Выводы:

В ходе лабораторной работы мы познакомились с основными механизмами передачи данных и синхронизацией между процессами, такими как очереди сообщений, сегменты разделяемой памяти и синхронизация посредством семафоров. Изучили принципы работы потоков и команды bash по их управлению.