Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное учреждение высшего образования

«Новгородский государственный университет имени Ярослава Мудрого»

Институт «Электронных и информационных систем»

Кафедра «Информационных систем и технологий»

Лабораторная работа №7 по учебной дисциплине «Операционные системы»

По направлению 09.03.01 «Информатика и вычислительная техника»

Отчёт

Принял преподаватель:

\_\_\_\_\_\_\_\_\_ Ананьев В. В.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2021 г.

Выполнил студент группы 9091: Баринов Д. А.

\_\_\_\_\_\_\_

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2021 г.

Великий Новгород

2021

**Цель работы:** научиться работать с очередям сообщений в UNIX-like системах

**Задание:**

В данной работе требуется использовать параллельные потоки (threads), а не процессы.

Основной поток должен сгенерировать и вывести на экран 4 случайных числа.

Далее эти числа должны быть переданы во второй процесс через очередь сообщений.

Второй поток должен сгенерировать все возможные перестановки из этих чисел без повторений («1 2 3 4», «2 1 3 4», «4 3 1 2» и т.д.) и передать их обратно.

Первый поток должен получить все эти перестановки и вывести их на экран. В конце требуется вывести количество полученных перестановок.

Первый поток должен самостоятельно выполнить освобождение всех выделенных ресурсов в конце своей работы.

В отчете привести исходный код программы, а также результаты, выведенные на экран.

**Код**

#include <pthread.h>

#include <unistd.h>

#include <stdio.h>

#include <sys/types.h>

#include <sys/msg.h>

#include <sys/ipc.h>

#include <iostream>

#include <time.h>

using namespace std;

int comp(const void \*a, const void \*b)

{

return (\*((int \*)a) - \*((int \*)b));

}

void swap(int \*a, int i, int j)

{

int s = a[i];

a[i] = a[j];

a[j] = s;

}

bool NextSet(int \*a)

{

int j = 2;

while (j != -1 && a[j] >= a[j + 1]) j--;

if (j == -1)

return false;

int k = 3;

while (a[j] >= a[k]) k--;

swap(a, j, k);

int l = j + 1, r = 3;

while (l<r)

swap(a, l++, r--);

return true;

}

struct message

{

long mtype;

int mas[4];

}mess;

void\* comb(void\* idm)

{

int id = \*((int\*)idm);

struct message thr;

msgrcv(id, &thr, 16, 5, 0);

qsort(thr.mas,4, sizeof(int),comp);

thr.mtype = 6;

msgsnd(id, &thr, 16, 0);

while (NextSet(thr.mas))

{

thr.mtype = 6;

msgsnd(id, &thr, 16, 0);

}

thr.mtype = 6;

thr.mas[0] = -1;

msgsnd(id, &thr, 16, 0);

}

int main()

{

int iter = 0;

int msgid = msgget(IPC\_PRIVATE, 0666|IPC\_CREAT);

if (msgid < 0)

{

cout << "error with msgget." << endl;

exit(-1);

}

srand(time(NULL));

cout << "Source: ";

for (int i = 0; i < 4; i++)

{

mess.mas[i] = rand()%100;

cout << mess.mas[i] << "; ";

} cout << endl;

mess.mtype = 5;

msgsnd(msgid, &mess, 16, 0);

pthread\_t thread0;

pthread\_create(&thread0, NULL, comb, (void\*)&msgid);

pthread\_join(thread0, NULL);

while(1)

{

msgrcv(msgid, &mess, 16, 6, 0);

if(mess.mas[0] == -1)

{ break;}

else

{

for(int i = 0; i < 4; i++)

{

cout << mess.mas[i] << " ";

}

iter++;

cout << endl;

}

}

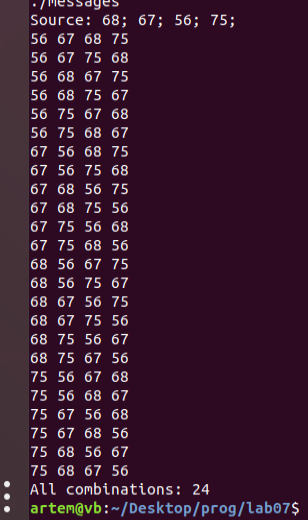
cout << "All combinations: " << iter << endl;

msgctl(msgid, IPC\_RMID, NULL);

return 0;

}

**Результат выполнения программы:**



**Вывод:** В процессе выполнения лабораторной работы я научился работать очередями сообщений.