Министерство образования и науки Украины

Днепропетровский национальный университет железного

транспорта имени академика В.Лазаряна

**Лабораторная работа №1**

по дисциплине: «Операционные системы»

Выполнил: ст. группы ПЗ1611

Галушка Александр  
Принял: Андрющенко В. А.

г.Днепр

2018 год

**Тема: Створення потоків і їх синхронізація в режимі користувача.**

**Варіанти індивідуальних завдань.**

Характеристики ресурсу.

1. Черга на основі масиву текстової інформації фіксованого розміру.
2. Черга на основі динамічного масиву текстової інформації.
3. Черга на основі масиву структурованих об’єктів фіксованого розміру.
4. Стек на основі динамічного списку.

Спосіб використання ресурсу.

1. Потоки першого виду додають інформацію, потоки другого виду видаляють інформацію.
2. Потоки першого виду додають інформацію, потоки другого виду виконують пошук інформації.

**Текст програми з критичною секцією**

#include "stdafx.h"

#include <windows.h>

#include <process.h>

#include <iostream>

#include <conio.h>

#include <queue>

#include <time.h>

#include <list>

using namespace std;

#define SIZE 7

CRITICAL\_SECTION critsect;

struct Point {

int x, y;

Point(int x, int y) {

this->x = x;

this->y = y;

}

Point(){}

};

//критическая секция

queue <Point\*> qq;

HANDLE SetQ, PrintQ, SortQ; //Потоки для работы с массивом

void Set(void \*) {

EnterCriticalSection(&critsect);

cout << "begin set \n";

srand(time(NULL));

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

qq.push(new Point(rand() % 10 + 1, rand() % 10 + 1));

cout << "end of set \n";

LeaveCriticalSection(&critsect);

\_endthread();

}

void Print(void \*) {

EnterCriticalSection(&critsect);

cout << "begin print \n";

while (!qq.empty()) {

cout << "[" << qq.front()->x << " , " << qq.front()->y << "]";

qq.pop();

}

cout << "end of print \n";

LeaveCriticalSection(&critsect);

\_endthread();

}

void Sort(void \*) {

EnterCriticalSection(&critsect);

cout << "begin sort \n";

list <Point\*> List;

while (!qq.empty()) {

List.push\_front(qq.front());

qq.pop();

}

List.sort();

for (auto it = List.begin(); it != List.end(); it++)

qq.push(\*it);

cout << "end of sort \n";

LeaveCriticalSection(&critsect);

\_endthread();

}

int main()

{

InitializeCriticalSection(&critsect);

SetQ = (HANDLE)\_beginthread(Set, 1024, NULL);

Sleep(1000);

SortQ = (HANDLE)\_beginthread(Sort, 1024, NULL);

Sleep(1000);

PrintQ = (HANDLE)\_beginthread(Print, 1024, NULL);

Sleep(1000);

DeleteCriticalSection(&critsect);

system("pause");

return 0;

}

**Текст програми без критичної секції**

#include "stdafx.h"

#include <windows.h>

#include <process.h>

#include <iostream>

#include <conio.h>

#include <queue>

#include <time.h>

#include <list>

using namespace std;

#define SIZE 7

struct Point {

int x, y;

Point(int x, int y) {

this->x = x;

this->y = y;

}

Point(){}

};

CRITICAL\_SECTION critsect; //критическая секция

queue <Point\*> qq;

HANDLE SetQ, PrintQ, SortQ; //Потоки для работы с массивом

void Set(void \*) {

cout << "begin set \n";

srand(time(NULL));

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

qq.push(new Point(rand() % 10 + 1, rand() % 10 + 1));

cout << "end of set \n";

}

void Print(void \*) {

cout << "begin print \n";

Sleep(1);

while (!qq.empty()) {

cout << "[" << qq.front()->x << " , " << qq.front()->y << "]";

qq.pop();

}

cout << "end of print \n";

}

void Sort(void \*) {

cout << "begin sort \n";

list <Point\*> List;

while (!qq.empty()) {

List.push\_front(qq.front());

qq.pop();

}

List.sort();

for (auto it = List.begin(); it != List.end(); it++)

qq.push(\*it);

cout << "end of sort \n";

}

int main()

{

InitializeCriticalSection(&critsect);

SetQ = (HANDLE)\_beginthread(Set, 1024, NULL);

SortQ = (HANDLE)\_beginthread(Sort, 1024, NULL);

PrintQ = (HANDLE)\_beginthread(Print, 1024, NULL);

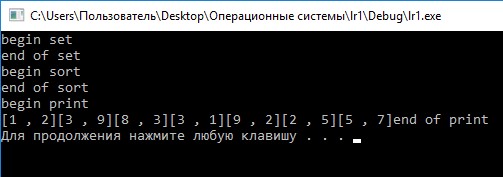
DeleteCriticalSection(&critsect);

system("pause");

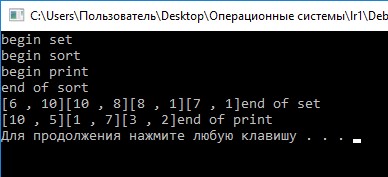
return 0;

}

**Результати виконання програми з критичною секцією**



**Результати виконання програми без критичної секції**



**Висновок**

В процесі цієї роботи я навчився користуватися критичною секцією і потоками.