Лабораторная работа №6

**Потоки в С++**

**Цель работы:** изучение основных возможностей потоков управления и потоков ввода-вывода. Получение навыков работы со стандартными средствами управления потоками в С++11. Знакомство с классом Thread и стандартными средствами синхронизации потоков.

**Задание:**

1. Изучить основные классы и их возможности работы с потоками в С++11.
2. Разработать программу в соответствии с вариантом задания. Программа должна содержать 2 потока Thread для реализации основного задания лабораторной работы. Вывод организовать в отдельном потоке.
3. Реализовать классы и выполнить перегрузку оператора функтора для реализации поставленной основной задачи.
4. Разработать программу в соответствии с вариантом задания (номер варианта + 3), используя API CreateThread.
5. Сделать выводы о проделанной работе.

**Содержание отчета:**

1. Тема, цель работы, вариант задания.

2. Описание иерархии классов (рисунок диаграммы классов, текстовое описание свойств и методов каждого класса).

3. Исходный текст модулей программы.

4. Вывод.

**Контрольные вопросы:**

1. Дополнительные принципы ООП.
2. Способы реализация параллелизма.
3. Типизация.
4. Сохраняемость и области видимости переменных.
5. Синхронизация.
6. Потоки управления.
7. Class std::thread.
8. Мьютексы.
9. Способы запуска потоков.

**Пример**

#include <mutex>

#include <thread>

#include <iostream>

#include <windows.h>

void add(int a[10], std::mutex &m){

while (1)

{

m.lock();

for (int i = 0; i < 10; i++)

a[i] = a[i] + 1;

//Sleep(10);

m.unlock();

}

}

void minus(int a[10], std::mutex &m){

while (1){

m.lock();

for (int i = 0; i < 10; i++)

a[i] = a[i] - 1;

//Sleep(10);

m.unlock();

}

}

void output(int a[10], std::mutex &m){

int p = 0;

while (1)

{

//m.lock();

for (int i = 0; i < 10; i++)

{

//std::cout << a[i] << " ";

//std::cout << std::endl;

std::cout << char(8);

if (p == 0) { std::cout << "-"; p = 1; }

else

if (p == 1) { std::cout << "+"; p =0; }

Sleep(500);

}

std::cout << std::endl;

//m.unlock();

}

}

int main(){

int a[10];

std::mutex m;

setlocale(0, "RUS");

for (int i = 0; i < 10; i++)

a[i] = 0;

std::thread th2(minus, a, ref(m)); //1 поток

std::thread th1(add, a, ref(m)); //2 поток

std::thread th3(output, a, ref(m)); //поток вывода

th1.detach();

th2.detach();

th3.detach();

std::cout << "finish";

system("pause");

return 0;

}

**Варианты заданий:**

1. Один поток заменяет строчные символы на прописные, а другой поток - наоборот. Символы выбираются случайным образом. Произвести синхронный вывод при каждой итерации. Показать выполнение работы программы в синхронном и асинхронном режимах.

2. Один поток добавляет случайные символы в строку, а другой поток удаляет случайный символ из строки. Произвести синхронный вывод при каждой итерации. Показать выполнение работы программы в синхронном и асинхронном режимах.

3. Один поток удаляет пробелы в строке и вставляет их в случайное место, а другой поток выполняет циклический сдвиг текста. Произвести синхронный вывод при каждой итерации. Показать выполнение работы программы в синхронном и асинхронном режимах.

4. Один поток удаляет случайное число из текста, а другой поток заносит в текст символьное представление случайных чисел. Произвести синхронный вывод при каждой итерации. Показать выполнение работы программы в синхронном и асинхронном режимах.

5. Один поток генерирует входные данные для функции, F=(sin(x)+exp(2x))/tg(x) а другой поток вычисляет значение этой функции. Произвести синхронный вывод при каждой итерации. Показать выполнение работы программы в синхронном и асинхронном режимах.

6. Один поток выполняет подсчет количества гласных букв в тексте, а другой вставляет или удаляет случайным образом гласную букву. Произвести синхронный вывод при каждой итерации. Показать выполнение работы программы в синхронном и асинхронном режимах.

7. Один поток создает бегущую строку (из случайных символов), а другой вставляет или удаляет между ними случайные знаки препинания. Произвести синхронный вывод при каждой итерации. Показать выполнение работы программы в синхронном и асинхронном режимах.

8. Один поток удаляет лишние пробелы в строке, а другой подсчитывает количество слов в тексте. Произвести синхронный вывод при каждой итерации. Показать выполнение работы программы в синхронном и асинхронном режимах.

9. Один поток выводит “визуальный progress bar” в консоль, другой поток выполняет цифровой подсчет текущего значения progress bar. Произвести синхронный вывод при каждой итерации. Показать выполнение работы программы в синхронном и асинхронном режимах.

10. Реализовать функцию вывода времени в консоль. Каждую компоненту времени изменяет отдельный поток. Произвести синхронный вывод при каждой итерации. Показать выполнение работы программы в синхронном и асинхронном режимах.