МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА»
(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и
автоматизированных систем

Лабораторная работа № 6

по дисциплине: «Объектно-ориентированное
программирование»

Выполнил: ст. группы ПВ-211
Медведев Дмитрий Сергеевич

Проверил:
Буханов Дмитрий Геннадьевич
Харитонов Сергей Дмитриевич

Белгород 2023 г.

Потоки в C++

Вариант 8

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

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.

Задание 1

```
#include <iostream>
#include "mutex"
#include "thread"
#include "string"

std::mutex _lock;

class ExtraSpacesCleaner {
public:
    std::string &target;

    explicit ExtraSpacesCleaner(std::string &Target) : target(Target) {};

    void operator()() {
        bool previousSymbolIsSpace = false;
        for (int i = 0; i < target.size(); i++) {
            _lock.lock();
            if (target[i] == ' ') {
                if (previousSymbolIsSpace) {
                    target.erase(target.begin() + i);
                    i--;
                }
                previousSymbolIsSpace = true;
            } else {
                previousSymbolIsSpace = false;
            }
            std::cout << "Thread 2: " << target << "\n";
            _lock.unlock();

            std::this_thread::sleep_for(std::chrono::milliseconds
(rand()%20));
        }
    }
```

```
};


class WordCounter {
public:
    std::string &target;

    explicit WordCounter(std::string &Target) : target(Target) {};

    int operator()(int &count) {
        bool previousSymbolIsSpace = false;
        for (int i = 0; i < target.size(); i++) {
            _lock.lock();
            if (target[i] == ' ')
                previousSymbolIsSpace = true;
            else {
                if (previousSymbolIsSpace)
                    count++;
                previousSymbolIsSpace = false;
            }

            std::cout << "Thread 1: " << target << "\n";
            _lock.unlock();

            std::this_thread::sleep_for(std::chrono::milliseconds
(rand()%10));
        }

        count += target[target.size()] != ' ';

        return count;
    }
};
```

Асинхронный вариант:

```
int main() {
    std::string test = "Первое     правило        Бойцовского      клуба:
никому не       рассказывать    о    Бойцовском       клубе.\n";
    WordCounter wordCounter(test);
    ExtraSpacesCleaner extraSpacesCleaner(test);

    int countWords;
    std::thread t1(wordCounter, std::ref(countWords));
    std::thread t2(extraSpacesCleaner);

    t1.join();
    t2.join();

    std::cout << countWords;
}
```

Однопоточный вариант:

```
int main() {
    std::string test = "Первое     правило        Бойцовского      клуба:
никому не       рассказывать    о    Бойцовском       клубе.\n";
    WordCounter wordCounter(test);
    ExtraSpacesCleaner extraSpacesCleaner(test);

    int countWords;
    wordCounter(countWords);
    extraSpacesCleaner();

    std::cout << countWords << "\n";
}
```

Задание 2

```
#include <iostream>
#include <Windows.h>
#include <mutex>
#include <string>
#include "cctype"

#define ITER 50

using namespace std;
mutex mtx;

// Функция для добавления случайных символов в строку
DWORD WINAPI make_lowercase(LPVOID lpParam)
{
 string& str = *(string*)lpParam;
 for (size_t i = 0; i < ITER; i++) {
  // Захватываем мьютекс для синхронизации вывода
  mtx.lock();

  // Делаем случайный символ в нижнем регистре
  int pos = rand() % str.size();
  str[pos] = tolower(str[pos]);

  // Выводим текущее состояние строки
  cout << "string: " << str << endl;

  // Освобождаем мьютекс
  mtx.unlock();

  // Задержка для эффекта анимации
  Sleep(100);
 }
 return 0;
}

// Функция для удаления случайного символа из строки
DWORD WINAPI make_uppercase(LPVOID lpParam)
{
 string& str = *(string*)lpParam;
 for (size_t i = 0; i < ITER; i++) {
```

```
  // Захватываем мьютекс для синхронизации вывода
  mtx.lock();

  // Делаем случайный символ в верхнем регистре

   int pos = rand() % str.size();
   str[pos] = toupper(str[pos]);

   // Выводим текущее состояние строки
   cout << "string: " << str << endl;


  // Освобождаем мьютекс
  mtx.unlock();

  // Задержка для эффекта анимации
  Sleep(100);
 }
 return 0;
}

int main()
{
 string str = "qwerty";

 // Создаем два потока
 HANDLE hThread1 = CreateThread(NULL, 0, make_lowercase, &str, 0, NULL);
 HANDLE hThread2 = CreateThread(NULL, 0, make_uppercase, &str, 0, NULL);

 // Ждем, пока оба потока завершат свою работу
 WaitForSingleObject(hThread1, INFINITE);
 WaitForSingleObject(hThread2, INFINITE);

 // Закрываем дескрипторы потоков
 CloseHandle(hThread1);
 CloseHandle(hThread2);

 return 0;
}
```

Вывод: в ходе лабораторной работы мы познакомились с многопоточным программированием в C++, написали программу, работающую в многопоточном режиме.