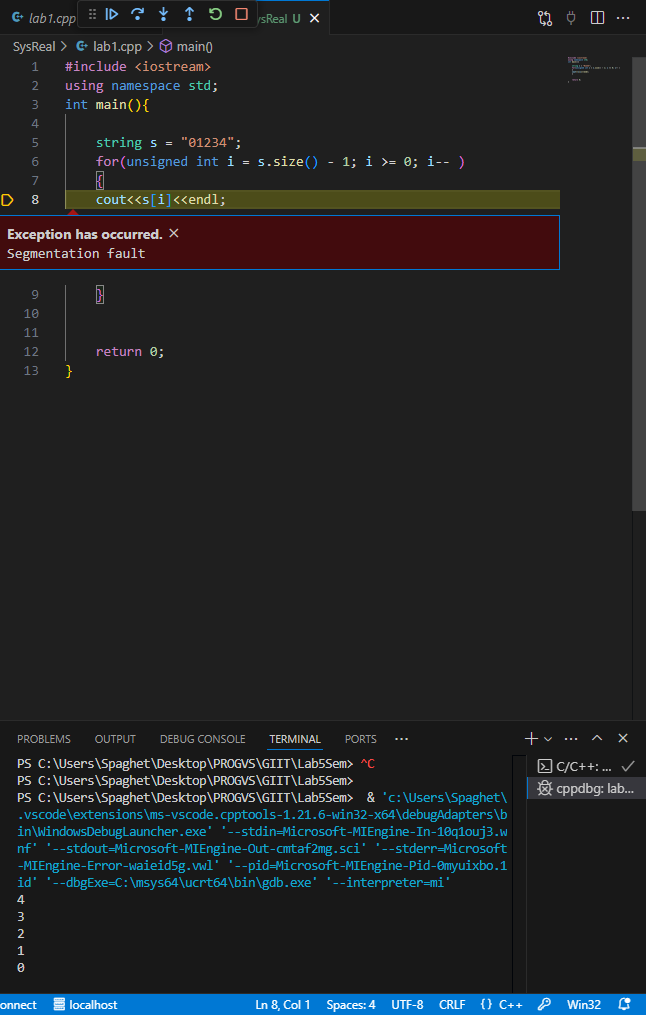
**Лабораторная №1 Утренняя зарядка**

***Пункт 1.1***

программа выдает segmentation fault, поскольку i задается как unsigned integer, который принимает значения от 0 до 4,294,967,295, поэтому когда i = 0, и выполнятся i--, i принимает своё максимальное значение, следовательно выходя за границы размера строки s

***Пункт 1.2 Упражнение 1***

Используемый код:

#include <iostream>

#include <time.h>

using namespace std;

int factorial(int n) {

    int faktr = 1;

    for (int i = 1; i <= n; ++i) {

        faktr \*= i;

    }

    return faktr;

}

int main() {

    clock\_t start = clock();

    for (int i = 0; i < 10000000; i++) {

        int result = factorial(10);

    }

    clock\_t end = clock();

    double seconds = (double)(end - start) / CLOCKS\_PER\_SEC;

    cout << seconds << endl;

}

В режиме Release время выполнения всегда отображается 0.05, а редим Debug ~150. Выполнение программы в режиме Release всегда будет быстрее, поскольку Release выполняться с оптимизациями (например устраняет повторы или устранение неиспользуемых переменных), тем временем debug этого не делает и при этом на debug влияют фоновые задачи на компьютере, поэтому у debug время постоянно меняется.

***Пункт 1.3 Упражнение 2***

Используемый код:

#include <iostream>

#include <thread>

#include <mutex>

#include <time.h>

#include <string>

using namespace std;

void Func(string name) {

    int value = 1;

    for (int i = 0; i < 10000000; i++) {

        for (int j = 0; j < 10; j++) {

            value \*= j;

        }

    }

}

int main() {

    thread thread\_1(Func, "t1");

    thread thread\_2(Func, "t2");

clock\_t start\_1 = clock();

    thread\_1.join();

    thread\_2.join();

    clock\_t end\_1 = clock();

    double time1 = (double)(end\_1 - start\_1) / CLOCKS\_PER\_SEC;

    cout << "Время параллельно: " << time1 <<"\n";

    clock\_t start\_2 = clock();

    Func("first");

    Func("second");

    clock\_t end\_2 = clock();

    double time2 = (double)(end\_2 - start\_2) / CLOCKS\_PER\_SEC;

    cout << "Время последовательно: " << time2 << "\n";

}



В параллельном запуске, два потока выполняются одновременно и выполнение программы может быть разделено между ядрами процессора. При последовательном запуске программа выполняется только на одном ядре, выполняя задачи одна за другой, поэтому время выполнения программы при параллельном выполнении почти в два раза быстрее чем при последовательном