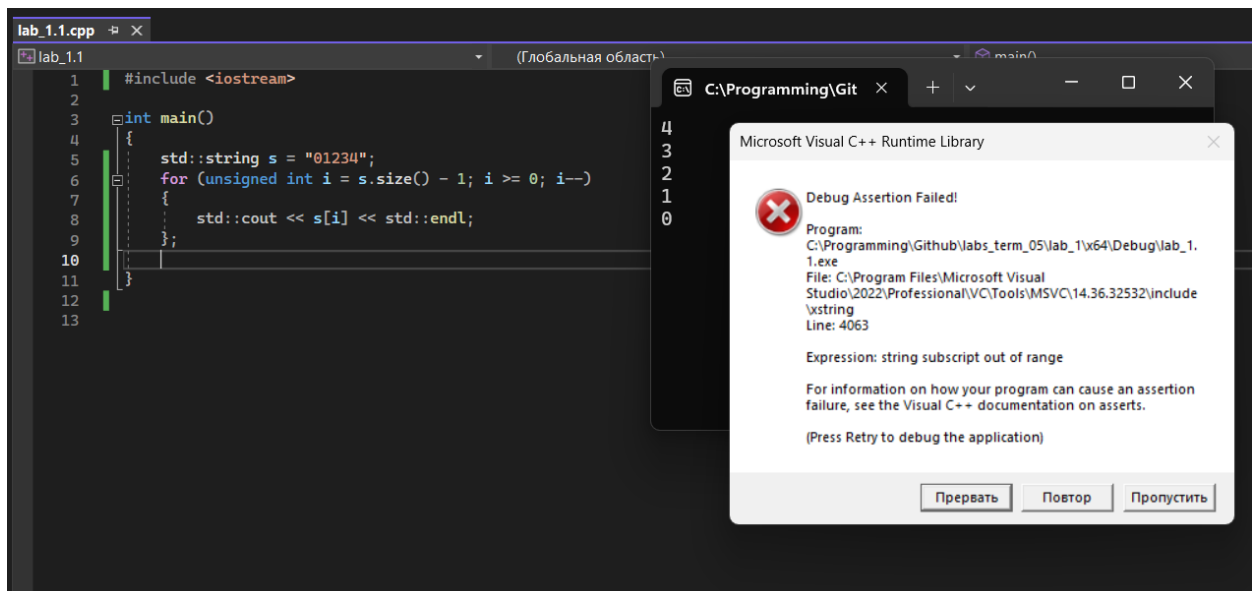


Лабораторная работа №1

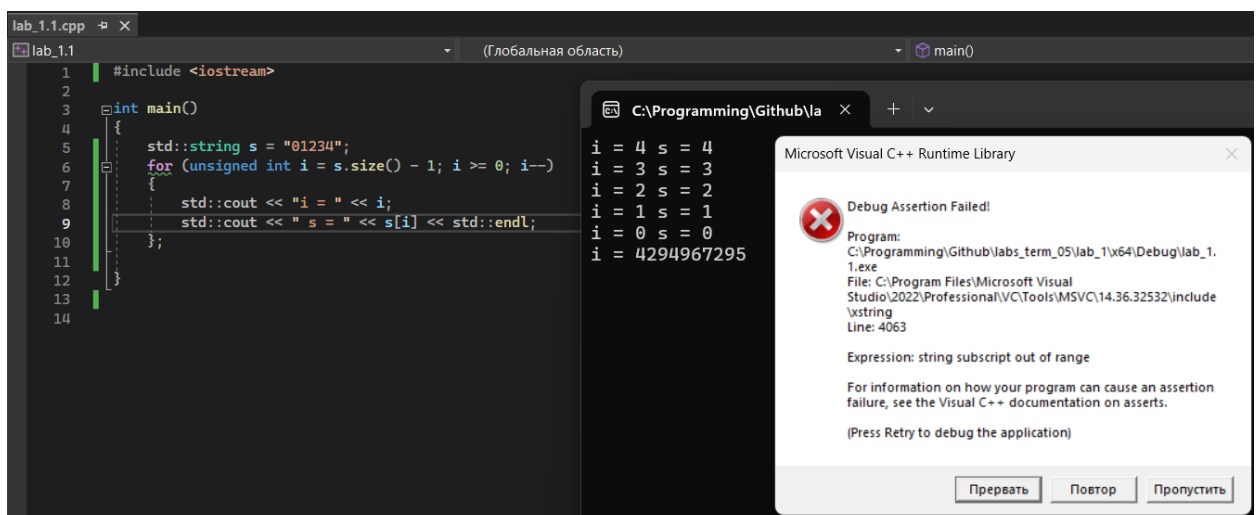
Соколов Арсений

Пункт 1.1 Разминка

Исправив недочеты в предоставленном коде, запускаем и получаем ошибку out of range:



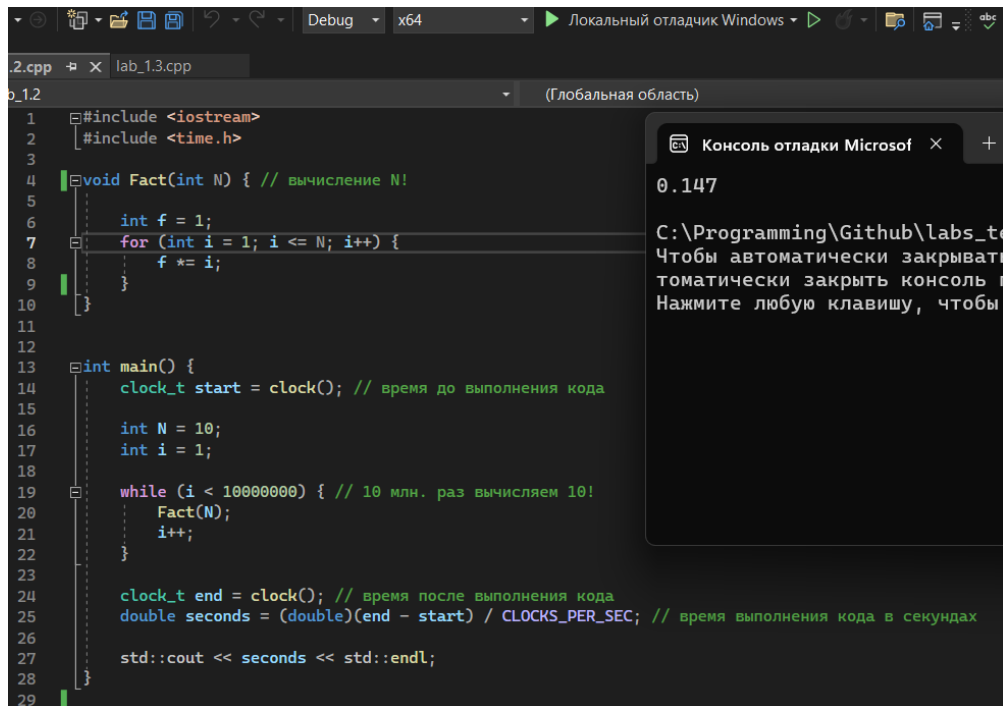
Доработаем код:



И получаем, что в последней итерации цикла мы пытаемся положить в заведомо положительную переменную *unsigned int i* значение равное -1 . Тогда компилятор кладет в переменную *i* значение равное $INT_MAX - i = 4294967296 - 1 = 4294967295$, а такого индекса в *string s* нет, поэтому получается выход из диапазона.

Пункт 1.2 Упражнение 1

Время выполнения программы в режиме Debug составляет 0,147 секунды:



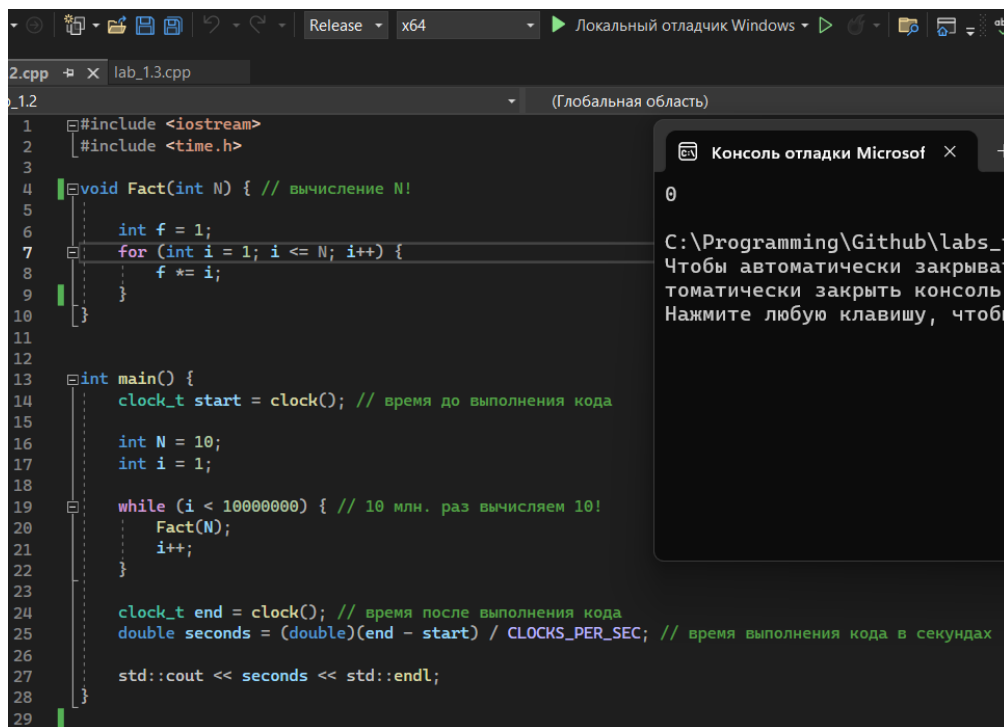
```
1  #include <iostream>
2  #include <time.h>
3
4  void Fact(int N) { // вычисление N!
5
6      int f = 1;
7      for (int i = 1; i <= N; i++) {
8          f *= i;
9      }
10 }
11
12
13 int main() {
14     clock_t start = clock(); // время до выполнения кода
15
16     int N = 10;
17     int i = 1;
18
19     while (i < 10000000) { // 10 млн. раз вычисляем 10!
20         Fact(N);
21         i++;
22     }
23
24     clock_t end = clock(); // время после выполнения кода
25     double seconds = (double)(end - start) / CLOCKS_PER_SEC; // время выполнения кода в секундах
26
27     std::cout << seconds << std::endl;
28 }
29
```

Консоль отладки Microsoft

0.147

C:\Programming\Github\labs_t...
Чтобы автоматически закрывать...
томатически закрыть консоль п...
Нажмите любую клавишу, чтобы...

Время выполнения в режиме Release составляет 0 секунд:



```
1  #include <iostream>
2  #include <time.h>
3
4  void Fact(int N) { // вычисление N!
5
6      int f = 1;
7      for (int i = 1; i <= N; i++) {
8          f *= i;
9      }
10 }
11
12
13 int main() {
14     clock_t start = clock(); // время до выполнения кода
15
16     int N = 10;
17     int i = 1;
18
19     while (i < 10000000) { // 10 млн. раз вычисляем 10!
20         Fact(N);
21         i++;
22     }
23
24     clock_t end = clock(); // время после выполнения кода
25     double seconds = (double)(end - start) / CLOCKS_PER_SEC; // время выполнения кода в секундах
26
27     std::cout << seconds << std::endl;
28 }
29
```

Консоль отладки Microsoft

0

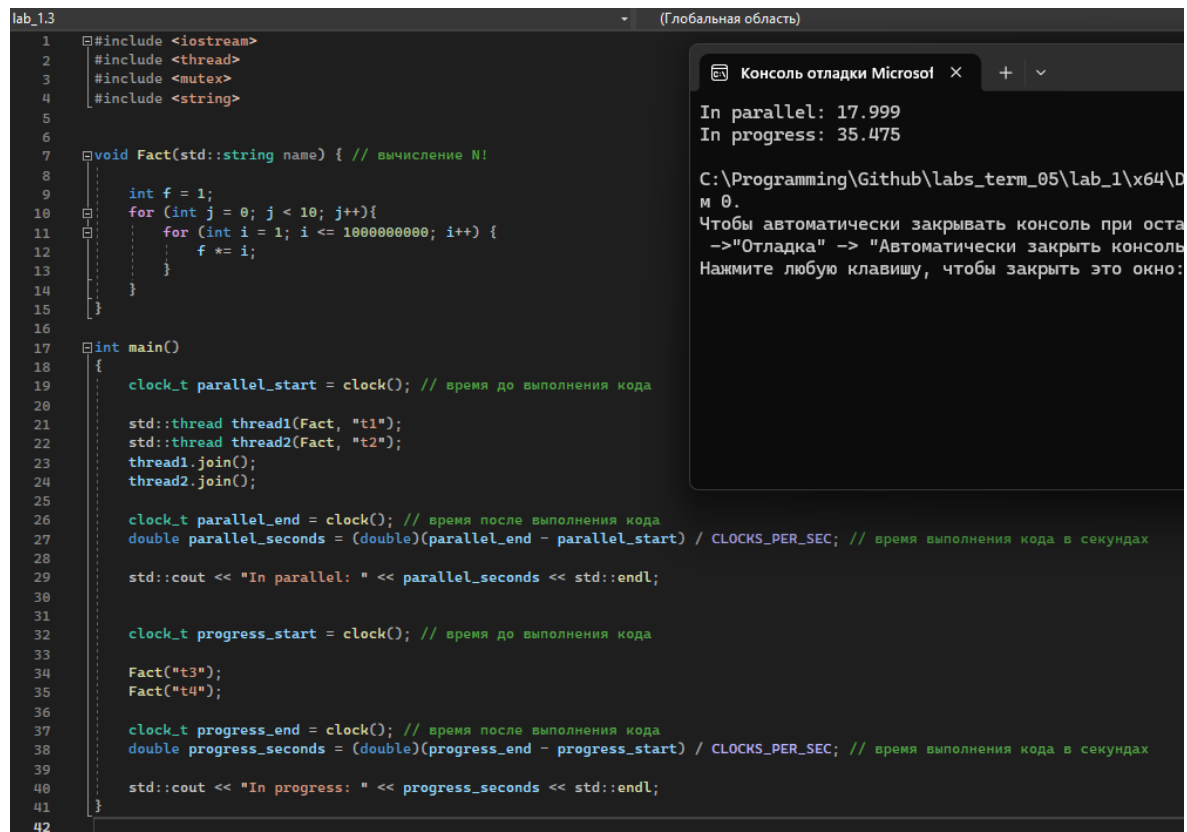
C:\Programming\Github\labs_t...
Чтобы автоматически закрывать...
томатически закрыть консоль...
Нажмите любую клавишу, чтобы...

Это обусловлено тем, что компилятор использует некоторые оптимизации в режиме Release и он настолько умный, что понимает, что цикл с факториалами абсолютно бесполезный и не прогоняет его. Таким образом, время выполнения кода, состоящего только из этого, равно 0 секунд.

Пункт 1.3 Упражнение 2

Время выполнения программы, когда потоки параллельны – 18 секунд

Время выполнения программы, когда потоки последовательны – 35,5 секунд



The screenshot shows a C++ program in a code editor and its output in a debug console. The program calculates the factorial of 1000000000 using two parallel threads. The console output shows the parallel execution time as 17.999 seconds and the sequential execution time as 35.475 seconds.

```
lab_1.3 (Глобальная область)
1  #include <iostream>
2  #include <thread>
3  #include <mutex>
4  #include <string>
5
6
7  void Fact(std::string name) { // вычисление N!
8
9      int f = 1;
10     for (int j = 0; j < 10; j++){
11         for (int i = 1; i <= 1000000000; i++) {
12             f *= i;
13         }
14     }
15 }
16
17 int main()
18 {
19     clock_t parallel_start = clock(); // время до выполнения кода
20
21     std::thread thread1(Fact, "t1");
22     std::thread thread2(Fact, "t2");
23     thread1.join();
24     thread2.join();
25
26     clock_t parallel_end = clock(); // время после выполнения кода
27     double parallel_seconds = (double)(parallel_end - parallel_start) / CLOCKS_PER_SEC; // время выполнения кода в секундах
28
29     std::cout << "In parallel: " << parallel_seconds << std::endl;
30
31     clock_t progress_start = clock(); // время до выполнения кода
32
33     Fact("t3");
34     Fact("t4");
35
36     clock_t progress_end = clock(); // время после выполнения кода
37     double progress_seconds = (double)(progress_end - progress_start) / CLOCKS_PER_SEC; // время выполнения кода в секундах
38
39     std::cout << "In progress: " << progress_seconds << std::endl;
40 }
41
42
```

Консоль отладки Microsoft

In parallel: 17.999
In progress: 35.475

C:\Programming\Github\labs_term_05\lab_1\x64\Debug\lab_1.exe
м 0.
Чтобы автоматически закрывать консоль при оставлении отладки -> "Отладка" -> "Автоматически закрыть консоль"
Нажмите любую клавишу, чтобы закрыть это окно:

Все дело в том, что ветви кода разделяются на потоки и выполняются «параллельно», то есть без предписанного порядка во времени.

Но на самом деле на устройствах с одним процессором это скорее *псевдопараллельное исполнение*, при котором создается видимость параллельной работы нескольких процессов. На самом деле они выполняются последовательно, но занимая малые кванты процессорного времени (ядро процессора в каждый такт работы выполняет строго одну операцию одного потока, затем "замораживает" его, переходя к другому и так далее происходит по циклу).