Лабораторная 8

Задание:программно реализуйте вычисление суммы последовательности чисел на основе последовательного кода, интерфейсов Pthreads и C++11 <thread>. Сравните время вычислений.
Цель:знакомство с программными интерфейсами управления потоками в Linux.

C++11 с использованием std::thread для вычисления суммы последовательности чисел от 1 до SIZE.

```
#include <chrono>
#include <iostream>
#include <thread>
#include <vector>

#define SIZE 1000000

struct ThreadData {
  int start;
  int end;
  long long result;
};

void partial_sum(ThreadData& data) {
  data.result = 0;
  for (int i = data.start; i < data.end; ++i) {
    data.result += i + 1;
  }
}

int main() {
  std::vector<std::thread> threads(2);
  ThreadData thread_data[2];

  thread_data[0].start = 0;
  thread_data[0].end = SIZE / 2;

  thread_data[1].start = SIZE / 2;
  thread_data[1].end = SIZE;

  auto start_time = std::chrono::high_resolution_clock::now();

  threads[0] = std::thread(partial_sum, std::ref(thread_data[0]));
  threads[1] = std::thread(partial_sum, std::ref(thread_data[1]));

  for (auto& t : threads) {
    t.join();
  }

  long long total_sum = thread_data[0].result + thread_data[1].result;
```

```
  auto end_time = std::chrono::high_resolution_clock::now();
  std::chrono::duration<double> elapsed = end_time - start_time;

  std::cout << "Total Sum: " << total_sum << "\n";
  std::cout << "Elapsed Time (std::thread): " << elapsed.count()
            << " seconds\n";

  return 0;
}
```

на C++ с использованием Pthreads
Pthreads является библиотекой на языке C,но она полностью совместима с C++

```
#include <pthread.h>

#include <chrono>
#include <iostream>
#include <numeric>

#define SIZE 1000000

struct ThreadData {
  int start;
  int end;
  long long result;
};

void* partial_sum(void* arg) {
  ThreadData* data = (ThreadData*)arg;
  data->result = 0;
  for (int i = data->start; i < data->end; ++i) {
    data->result += i + 1;
  }
  return NULL;
}

int main() {
  pthread_t threads[2];
  ThreadData thread_data[2];

  thread_data[0].start = 0;
  thread_data[0].end = SIZE / 2;
```

```
  thread_data[1].start = SIZE / 2;
  thread_data[1].end = SIZE;

  auto start_time = std::chrono::high_resolution_clock::now();

  pthread_create(&threads[0], NULL, partial_sum,
(void*)&thread_data[0]);
  pthread_create(&threads[1], NULL, partial_sum,
(void*)&thread_data[1]);

  pthread_join(threads[0], NULL);
  pthread_join(threads[1], NULL);

  long long total_sum = thread_data[0].result + thread_data[1].result;

  auto end_time = std::chrono::high_resolution_clock::now();
  std::chrono::duration<double> elapsed = end_time - start_time;

  std::cout << "Total Sum: " << total_sum << "\n";
  std::cout << "Elapsed Time (Pthreads): " << elapsed.count() << "
seconds\n";

  return 0;
}
```

результат работы программ:
lab8.cpp программа на C++11 с использованием std::thread
lab8_1.c программа на C с использованием Pthreads
lab8_2.cpp - программа на C++ с использованием Pthreads

```
tania@TaniaLaptop:~/5sem/os/os_labs/lab8$ g++ lab8.cpp -o 8
tania@TaniaLaptop:~/5sem/os/os_labs/lab8$ ./8
Total Sum: 500000500000
Elapsed Time (std::thread): 0.00407347 seconds
tania@TaniaLaptop:~/5sem/os/os_labs/lab8$ gcc lab8_1.c -o 81
tania@TaniaLaptop:~/5sem/os/os_labs/lab8$ ./81
Total Sum: 500000500000
Elapsed Time (Pthreads): 0.000154 seconds
tania@TaniaLaptop:~/5sem/os/os_labs/lab8$ g++ lab8_2.cpp -o 82
tania@TaniaLaptop:~/5sem/os/os_labs/lab8$ ./82
Total Sum: 500000500000
Elapsed Time (Pthreads): 0.00598269 seconds
tania@TaniaLaptop:~/5sem/os/os_labs/lab8$
```

При сравнении времени выполнения на одной и той же системе результаты могут варьироваться в зависимости от нагрузки на процессор и других факторов.