Козунов Алексей, 12 группа

Отчет

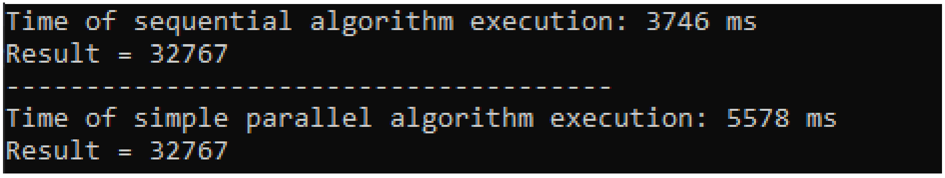
Код программы:

```
#include <iostream>
#include <algorithm>
#include <future>
#include <vector>
#include <chrono>

using namespace std;

int max_(int* start, int* end)
{
	return *max_element(start, end);
}

int parallel_max(std::vector<int>& v)
{
	using Task_type = int(int*, int*);
	packaged_task<Task_type> pt0{ max_ };
	packaged_task<Task_type> pt1{ max_ };
	packaged_task<Task_type> pt2{ max_ };
	packaged_task<Task_type> pt3{ max_ };
	packaged_task<Task_type> pt4{ max_ };
	packaged_task<Task_type> pt5{ max_ };

	future<int> f0{ pt0.get_future() };
	future<int> f1{ pt1.get_future() };
	future<int> f2{ pt2.get_future() };
	future<int> f3{ pt3.get_future() };
	future<int> f4{ pt4.get_future() };
	future<int> f5{ pt5.get_future() };

	int* first = &v[0];
	int delta = v.size() / 6;
	thread t1{ move(pt0), first, first + delta };
	thread t2{ move(pt1), first + delta, first + 2 * delta };
	thread t3{ move(pt2), first + 2 * delta, first + 3 * delta };
	thread t4{ move(pt3), first + 3 * delta, first + 4 * delta };
	thread t5{ move(pt4), first + 4 * delta, first + 5 * delta };
	thread t6{ move(pt5), first + 5 * delta, first + 6 * delta };
	t1.join();
	t2.join();
	t3.join();
	t4.join();
	t5.join();
	t6.join();
	std::vector<int> results{ f0.get(), f1.get(), f2.get(), f3.get(), f4.get(),
f5.get() };
	return *max_element(results.begin(), results.end());
}


int main()
{
	srand(0);
	unsigned int length = 1000000;
	std::vector<int> mas;
	mas.reserve(length);
	for (int i = 0; i < length; ++i)
	{
```

```
                mas.push_back(rand());
        }


        auto start = std::chrono::high_resolution_clock::now();
        int result = *max_element(mas.begin(), mas.end());
        auto end = std::chrono::high_resolution_clock::now();
        std::cout << "Time of sequential algorithm execution: " <<
std::chrono::duration_cast<std::chrono::microseconds>(end - start).count() << " ms" <<
std::endl;
        std::cout << "Result = " << result << std::endl;
        std::cout << "--------------------------------------" << std::endl;

        start = std::chrono::high_resolution_clock::now();
        result = parallel_max(mas);
        end = std::chrono::high_resolution_clock::now();
        std::cout << "Time of simple parallel algorithm execution: " <<
std::chrono::duration_cast<std::chrono::microseconds>(end - start).count() << " ms" <<
std::endl;
        std::cout << "Result = " << result << std::endl;
        std::cout << "--------------------------------------" << std::endl;

        return 0;
}
```

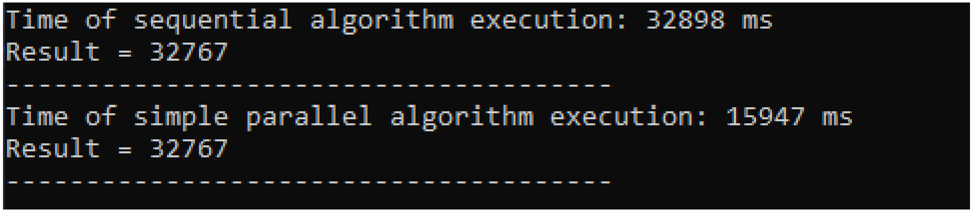
Результат:

- **Размер 100000:**



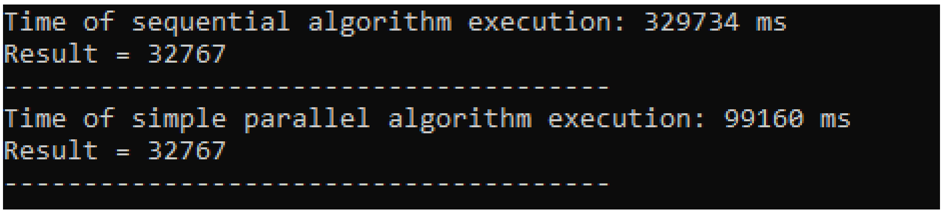
```
Time of sequential algorithm execution: 3746 ms
Result = 32767
--------------------------------------
Time of simple parallel algorithm execution: 5578 ms
Result = 32767
```

- **Размер 1000000:**



```
Time of sequential algorithm execution: 32898 ms
Result = 32767
--------------------------------------
Time of simple parallel algorithm execution: 15947 ms
Result = 32767
--------------------------------------
```

- **Размер 10000000:**



```
Time of sequential algorithm execution: 329734 ms
Result = 32767
--------------------------------------
Time of simple parallel algorithm execution: 99160 ms
Result = 32767
--------------------------------------
```