Laboratory 2 - Algorithms

Reference: https://cplusplus.com/reference/algorithm/
STL containers: https://cplusplus.com/reference/stl/

1. Global function

example:

```
int abs(int x)
{
	if (x < 0)
		return -x;
	else
		return x;
}
```

2. Lambda expression

syntax:

```
[captures] (params) -> return_type { body }
```

example:

```
auto abs = [ ] (int x) -> int {
	if (x < 0)
		return -x;
	else
		return x;
};
```

reference: https://en.cppreference.com/w/cpp/language/lambda

3. Functional objects

example:

```
class Abs
{
	public:
		int operator()(int x) { return (x < 0 ? -x : x); }
};
```

4. Ready-made STL templates

reference: https://cplusplus.com/reference/functional/

An example of modifying algorithm that uses lambda expression:

```
#include <algorithm>
#include <vector>

//creates vector of 5 string elements
std::vector<std::string> v(5);

char char_to_fill = '_';

//lambda expression that returns several '_' characters
auto blank = [char_to_fill] (int x) -> std::string {
   std::string result = "";
   for (int i = 0; i < x; i++)
       result += char_to_fill;
   return result;
};

//function from <algorithm> that fills container
std::fill(v.begin(), v.end(), blank(2));
//__ __ __ __ __
```

Exercises

1. Modifying algorithms
   Create a vector of 10 integer values. Use std::generate or std::generate_n function and:
   a. global function random_value that returns random values from <0;100>
   b. lambda function that returns random values from <0;100>
   c. functional object with operator() that returns random values from <0;100>

2. Non-modifying algorithm
   Use std::count_if to find in your vector each even value. Use lambda expression.

3. Sorting
   Use std::sort and ready-made STL template std::greater to sort your vector in descending order.

4. Removing elements using container method
   Use std::vector::erase to remove 5th element of your vector.

5. Multiply each element of your vector by n. Use std::transform and lambda expression with capture (n should be passed via capture) and store the result in a new vector.