МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ государственное БЮДЖЕТНОЕ

образовательное учреждение

высшего образования

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра защиты информации



**ОТЧЁТ**

**по лабораторной работе № 7**

«Итераторы и алгоритмы в языке программирования С++**»**

**по дисциплине: «***Программирование***»**

Выполнил:Проверил:

Студент гр. «АБ-121», «АВТФ» *доцент кафедры ЗИ*

*Втюрин Александр Романович Архипова А. Б.*

«24» мая 2022г«\_\_\_» \_\_\_\_\_\_ 2022 г.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (подпись)

Новосибирск 2022

**Задание к работе**: Составить 10 задач на анализ данных (STL-контейнер (допустимо использовать, например, vector и другие простейшие контейнеры)). Для каждой задачи контейнер/набор контейнеров может быть изменен. Применить алгоритмы библиотеки algorithm (использование лямбда функций обязательно, <https://en.cppreference.com/w/cpp/language/lambda>).

**Методика выполнения работы**:

1. Разработать алгоритм решения задачи по индивидуальному заданию.

2. Написать и отладить программу решения задачи.

3. Протестировать работу программы на различных исходных данных.

**Структура отчета**:

Для каждой задачи (всего 10).

1. Постановка задачи.

2. Теоретическое описание алгоритма/алгоритмов.

3. Листинг программы.

4. Результат работы программы.

**Теоретическая часть:**

В программе были использованы алгоритмы: any\_of, sort, find\_if, none\_of, copy\_if, count\_if, for\_each, generate, remove\_if, max\_element.

**Any\_of:**

Возвращает значение true, если хотя бы один элемент в указанном диапазоне соответствует условию, и значение false, если ни один элемент не соответствует условию.

**Sort:**

Упорядочивает элементы в указанном диапазоне в не нисходящем порядке или согласно критерию упорядочивания, заданному бинарным предикатом.

**Find\_if:**

Находит позицию первого вхождения элемента, удовлетворяющего определенному условию, в диапазон.

Входной итератор, ссылающийся на первый элемент в диапазоне, удовлетворяющий условию, заданному предикатом (результат предиката true). Если не найдено ни одного элемента, удовлетворяющего условию предиката, возвращается last.

**None\_of:**

Возвращает значение true, если не обнаружен ни один элемент в указанном диапазоне, соответствующий условию, и значение false, если обнаружен хотя бы один такой элемент.

**Copy\_if:**

Копирует из диапазона элементов те элементы, проверка которых на соответствие заданному условию дает значение true.

Выходной итератор, который увеличивает dest на единицу после каждого элемента, удовлетворяющего условию. Другими словами, возвращаемое значение минус dest равно количеству скопированных элементов.

**Count\_if:**

Возвращает количество элементов в диапазоне, значения которых соответствуют заданному условию.

**For\_each:**

Применяет заданный объект функции к каждому элементу в прямом порядке в пределах диапазона и возвращает объект функции.

**Generate:**

Присваивает значения, создаваемые объектом функции, каждому элементу в диапазоне.

**Remove\_if:**

Удаляет элементы, соответствующие предикату, из заданного диапазона без нарушения порядка остальных элементов и возвращает конец нового диапазона после удаления указанного значения.

**Max\_element:**

Находит первое вхождение наибольшего элемента в указанном диапазоне, где критерий упорядочивания может быть указан бинарным предикатом.

**Практическая часть:**

**Main.cpp:**

#include "Algorithms(m).h"

#include "Algorithms(v).h"

int main()

{

Algv();

Algm();

return 0;

}

**Algorithms(m).cpp:**

#include "Algorithms(m).h"

void Algm() {

cout << endl;

map<int, int> m = { {1,2}, {3,4}, {5,6} };

cout << "none\_of: " << none\_of(m.begin(), m.end(), [](pair<int, int> x) {

return (x.first > x.second);

}) << endl;

cout << endl;

map<int, string> m1 = { {1991, "Python"}, {1991, "Visual Basic"}, {2014, "Swift"}, {1983, "Ada"}, {1980, "C++"}, {1995, "PHP"}, {1995, "Java"} };

map<int, string> B;

copy\_if(m1.begin(), m1.end(), inserter(B, B.end()), [](pair<int, string> a) {return (a.first < 1995); });

cout << "copy\_if: \n" << B;

cout << endl;

map<int, bool> c = { {22, true}, {31, false}, {11, true}, {75, false}, {57, true} };

cout << "count\_if: " << count\_if(c.begin(), c.end(), [](pair<int, bool> r) {return r.second; }) << endl;

}

**Algorithms(v).cpp:**

#include "Algorithms(v).h"

void Algv() {

vector<vector<char>> any = { {'a','b','c','d'}, {'a','b','x'}, {'s','w','x'}, {'s','t','q','r'} };

cout << "any\_of: " << any\_of(any.begin(), any.end(), [](vector<char> x) {

return (x[1] == 'w');

}) << endl;

cout << endl;

vector<int> s;

mt19937 gen(43);//Генератор случайных чисел;

uniform\_int\_distribution <int> range(0, 100);//Задаёт диапозон генерируемых чисел

for (size\_t i = 0; i < 20; i++) {

s.push\_back(range(gen));//Заполнение массива

}

cout << "sort: " << s << endl;

sort(s.begin(), s.end(), [](int x, int y) {

return (x < y);

});

cout << "sort: " << s << endl;

cout << endl;

vector<string> f = { "a","ba","c","aa","bb","cc" };

auto b = find\_if(f.begin(), f.end(), [](const string& s) {

return (s[1] == 'a'); });

cout << "find\_if: " << \*b << endl;

cout << endl;

vector<int> each = { 12,534,567,123,-12,123,-31,57,8,1,0,-456 };

cout << "for\_each: ";

for\_each(each.begin(), each.end(), [](int y) {

cout << abs(y) << " ";

});

cout << endl;

cout << endl;

vector <int> m (5);

int g = 7;

generate(m.begin(), m.end(), [&g]() {

return g++; });

cout << "generate: " << m << endl;

cout << endl;

vector<int> part = { 29,23,20,22,17,15,26,51,19,12,35,40 };

partition(part.begin(), part.end(), [](int elem) {

if (elem % 2) return false;

else return true;

});

cout << "remove\_if: " << part << endl;

cout << endl;

vector <int> v = { 2, 1, 3, 6, 7, 9, 8 };

int max = \*max\_element(v.begin(), v.end());

cout << "max\_element: " << max << endl;

}

**Algorithms(m).h:**

#pragma once

#include <iostream>

#include <map>

#include <algorithm>

using namespace std;

template <typename T, typename T1>

ostream& operator<< (ostream& os, const map<T, T1>& v) {

for (auto i : v) {

os << i.first << " " << i.second << endl;

}

return os;

}

void Algm();

**Algorithms(v).h:**

#pragma once

#include <iostream>

#include <vector>

#include <algorithm>

#include <random>

#include <ctime>

using namespace std;

template <typename T>

ostream& operator<< (ostream& os, const vector<T>& v) {

for (auto i : v) {

os << i << " ";

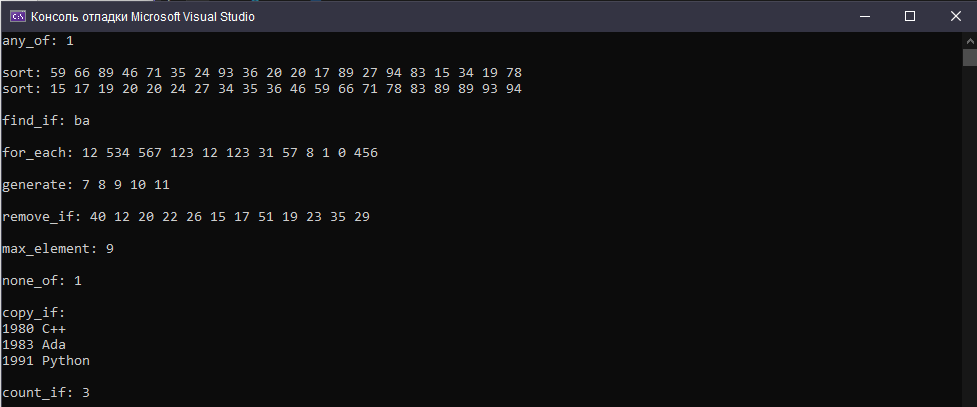
}

return os;

}

void Algv();

**Результат работы:**

****

**Вывод:**

В ходе выполнения работы были реализованы 10 задач на анализ данных в соответствии с индивидуальным заданием.