Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«**Пермский национальный исследовательский политехнический университет»**

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ**

Дисциплина: «Информатика»

Тема: Стандартные обобщенные алгоритмы библиотеки STL.

Семестр 2

Выполнил работу

Студент группы РИС-22-1Б

Аксёнов Д.О.

Проверил

Доцент кафедры ИТАС

Полякова О.А.

Г. Пермь-2023

**Постановка задачи**

### Задача 1.

* + - 1. Создать последовательный контейнер.
      2. Заполнить его элементами пользовательского типа (тип указан в варианте). Для пользовательского типа перегрузить необходимые операции.
      3. Заменить элементы в соответствии с заданием (использовать алгоритмы

replace\_if(), replace\_copy(), replace\_copy\_if(), fill()).

* + - 1. Удалить элементы в соответствии с заданием (использовать алгоритмы

remove(),remove\_if(), remove\_copy\_if(),remove\_copy())

* + - 1. Отсортировать контейнер по убыванию и по возрастанию ключевого поля (использовать алгоритм sort()).
      2. Найти в контейнере заданный элемент (использовать алгоритмы find(), find\_if(), count(), count\_if()).
      3. Выполнить задание варианта для полученного контейнера (использовать алгоритм

for\_each()) .

* + - 1. Для выполнения всех заданий использовать стандартные алгоритмы библиотеки STL.

### Задача 2.

1. Создать адаптер контейнера.
2. Заполнить его элементами пользовательского типа (тип указан в варианте). Для пользовательского типа перегрузить необходимые операции.
3. Заменить элементы в соответствии с заданием (использовать алгоритмы

replace\_if(), replace\_copy(), replace\_copy\_if(), fill()).

1. Удалить элементы в соответствии с заданием (использовать алгоритмы

remove(),remove\_if(), remove\_copy\_if(),remove\_copy())

1. Отсортировать контейнер по убыванию и по возрастанию ключевого поля (использовать алгоритм sort()).
2. Найти в контейнере элемент с заданным ключевым полем (использовать

алгоритмы find(), find\_if(), count(), count\_if()).

1. Выполнить задание варианта для полученного контейнера (использовать алгоритм

for\_each()) .

1. Для выполнения всех заданий использовать стандартные алгоритмы библиотеки STL.

### Задача 3

1. Создать ассоциативный контейнер.
2. Заполнить его элементами пользовательского типа (тип указан в варианте). Для пользовательского типа перегрузить необходимые операции.
3. Заменить элементы в соответствии с заданием (использовать алгоритмы

replace\_if(), replace\_copy(), replace\_copy\_if(), fill()).

1. Удалить элементы в соответствии с заданием (использовать алгоритмы

remove(),remove\_if(), remove\_copy\_if(),remove\_copy())

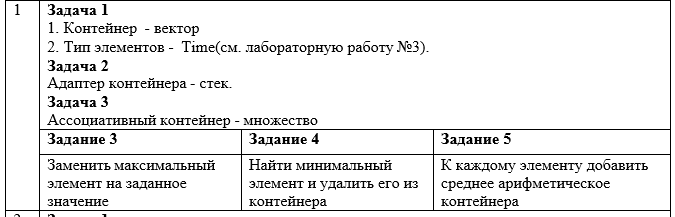
1. Отсортировать контейнер по убыванию и по возрастанию ключевого поля (использовать алгоритм sort()).
2. Найти в контейнере элемент с заданным ключевым полем (использовать

алгоритмы find(), find\_if(), count(), count\_if()).

1. Выполнить задание варианта для полученного контейнера (использовать алгоритм

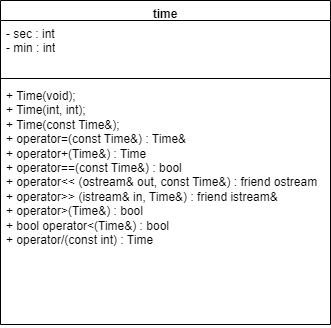
for\_each()) .

1. Для выполнения всех заданий использовать стандартные алгоритмы библиотеки STL.



**UML**

**1**

****

**Функции для решения задачи 1**

//формирование вектора

TVector make\_vector(int n)

{

Time a;

TVector v;

for (int i = 0;i < n;i++)

{

cin >> a; v.push\_back(a);

}

return v;

}

//печать вектора

void print\_vector(TVector v)

{

for (int i = 0;i < v.size();i++)

cout << v[i] << endl; cout << endl;

}

Time srednee(TVector v)

{

Time s = v[0];

//перебор вектора

for (int i = 1;i < v.size();i++) s = s + v[i];

int n = v.size();//количество элементов в векторе

return s/n;

}

void add(Time& t, Time& average)

{

t = t + average;

}

**Основная программа задачи 1**

void main()

{

int n; cout << "N? ";

cin >> n;

TVector v;

v = make\_vector(n);

print\_vector(v);

TVector::iterator i;

//поставили итератор i на максимальный элемент

cout << endl;

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

Time m = \*(i);

Time a;

cin >> a;

cout << endl;

replace(v.begin(), v.end(), m, a);

print\_vector(v);

cout << endl;

i = min\_element(v.begin(), v.end());

m = \*(i);

i = remove(v.begin(), v.end(), m);

v.erase(i, v.end());

print\_vector(v);

cout << endl;

Time s;

s = srednee(v);//нашли среднее арифметическое вектора

for\_each(v.begin(), v.end(), [&](Time& t) {

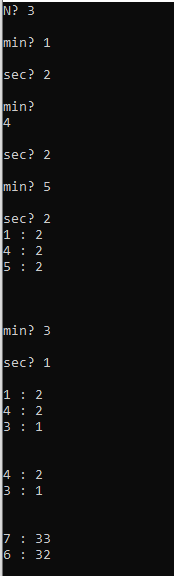
add(t, s);

});

print\_vector(v);

}

**Объяснение результатов работы программы 1**



**Функции для решения задачи 2**

//копирует стек в вектор

Vec copy\_stack\_to\_vector(St s)

{

Vec v;

while (!s.empty())//пока стек не пустой

{

//добавить в вектор элемент из вершиы стека

v.push\_back(s.top());

s.pop();

}

return v; //вернуть вектор как результат функции

}

//копирует вектор в стек

St copy\_vector\_to\_stack(Vec v)

{

St s;

for (int i = 0;i < v.size();i++)

{

s.push(v[v.size() - 1 - i]);//добавить в стек элемент вектора

}

return s; //вернуть стек как результат функции

}

void print\_stack(St s)

{

v = copy\_stack\_to\_vector(s);

for (int i = 0;i < v.size();i++)

{

Time temp = v[v.size() - 1 - i];

cout << temp << " ";

}

cout << endl;

}

St make\_stack(int n)

{

St s;

Time t;

for (int i = 0;i < n;i++)

{

cin >> t;//ввод переменной

s.push(t);//добавление ее в стек

}

return s;//вернуть стек как результат функции

}

Time sred(St s)

{

v = copy\_stack\_to\_vector(s);

int n = s.size();

Time m = s.top();

s.pop();

while (!s.empty())

{

m = m + s.top();

s.pop();

}

m = m / n;

s = copy\_vector\_to\_stack(v);

return m;

}

void add(Time& t, Time& average)

{

t = t + average;

}

void replaceMaxElement(St& s, const Time& newValue)

{

Vec v = copy\_stack\_to\_vector(s);

auto maxElementIter = max\_element(v.begin(), v.end());

if (maxElementIter != v.end()) {

replace(v.begin(), v.end(), \*maxElementIter, newValue);

s = copy\_vector\_to\_stack(v);

}

}

void eraseMinElement(St& s)

{

Vec v = copy\_stack\_to\_vector(s);

auto minElementIter = min\_element(v.begin(), v.end());

Time min = \*minElementIter;

minElementIter = remove(v.begin(), v.end(), min);

v.erase(minElementIter, v.end());

s = copy\_vector\_to\_stack(v);

}

void for\_each\_add(St& s)

{

Vec v = copy\_stack\_to\_vector(s);

Time sredn = sred(s);

for\_each(v.begin(), v.end(), [&](Time& t) {

add(t, sredn);

});

s = copy\_vector\_to\_stack(v);

}

**Основная программа задачи 2**

int main()

{

int n;

cout << "N? ";

cin >> n;

St v;

v = make\_stack(n);

print\_stack(v);

Time a;

cin >> a;

replaceMaxElement(v, a);

print\_stack(v);

cout << endl;

eraseMinElement(v);

print\_stack(v);

cout << endl;

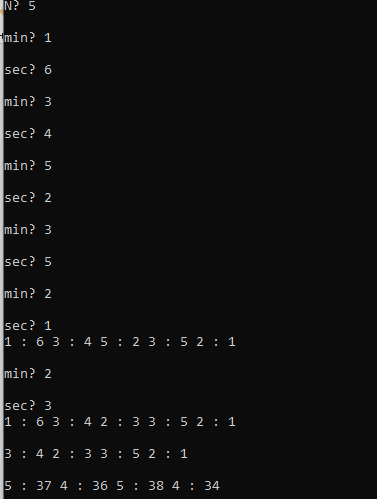
for\_each\_add(v);

print\_stack(v);

return 0;

}

**Объяснение результатов работы программы 2**



**Функции для решения задачи 3**

void replaceMax(Set& container, int newValue)

{

if (!container.empty()) {

// Находим максимальный элемент

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

// Удаляем максимальный элемент

container.erase(maxElement);

// Вставляем новое значение

container.insert(newValue);

}

}

// Функция для нахождения минимального элемента и его удаления

void removeMin(Set& container)

{

if (!container.empty()) {

// Находим минимальный элемент

int minElement = \*container.begin();

// Удаляем минимальный элемент

container.erase(minElement);

}

}

// Функция для добавления среднего арифметического ко всем элементам

void addAverage(Set& container)

{

if (!container.empty()) {

// Вычисляем среднее арифметическое

int sum = 0;

for (int num : container) {

sum += num;

}

int average = sum / container.size();

// Создаем временный контейнер для хранения измененных значений

std::set<int> tempContainer;

// Добавляем среднее арифметическое к каждому элементу

for (int num : container) {

tempContainer.insert(num + average);

}

// Заменяем исходный контейнер измененными значениями

container = std::move(tempContainer);

}

}

void print\_set(Set container)

{

for (int num : container) {

cout << num << " ";

}

cout << endl;

}

Set make\_set(int n)

{

Set temp;

int a;

for (int i = 0; i < n; i++)

{

cin >> a;

temp.insert(a);

}

return temp;

}

**Основная программа задачи 3**

int main()

{

int n;

cout << "n? " << endl;

cin >> n;

Set container = make\_set(n);

print\_set(container);

cout << "replace? " << endl;

cin >> n;

replaceMax(container, n);

print\_set(container);

// Нахождение минимального элемента и его удаление

removeMin(container);

print\_set(container);

// Добавление среднего арифметического ко всем элементам

addAverage(container);

print\_set(container);

return 0;

}

**Объяснение результатов работы программы 3**

