Министерство высшего образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет» (ПНИПУ)**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

ОТЧЁТ

по лабораторной работе №13

Тема: «Стандартные обобщенные алгоритмы

Библиотеки STL»

Выполнил

Студент группы РИС-22-2б

Хохряков А.С.

Проверил доц. Кафедры ИТАС

Полякова О. А.

Пермь 2023

# Постановка задачи

**Задача 1:**

1. Создать последовательный контейнер. Контейнер - двунаправленная очередь

2. Заполнить его элементами пользовательского типа Time (класс Time для работы с временными интервалами. Интервал должен быть

представлен в виде двух полей: минуты типа int и секунды типа int. при выводе минуты

отделяются от секунд двоеточием. Реализовать:

добавление секунд (учесть, что в минуте не может быть более 60 секунд)

вычитание секунд). Для пользовательского типа перегрузить необходимые операции.

3. Найти элемент с заданным ключом и добавить его на заданную позицию контейнера(использовать алгоритмы

replace if(), replace copy(), replace copy\_if(), fill()).

4. Найти элемент с заданным ключом и удалить его из контейнера(использовать алгоритмы

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

5. Отсортировать контейнер по убыванию и по возрастанию ключевого поля

(использовать алгоритм sort () ).

6. Найти в контейнере заданный элемент (использовать алгоритмы find () ,

find if(), count(), count if()).

7. Найти разницу между максимальным и минимальным элементами контейнера и

вычесть ее из каждого элемента

контейнера(использовать алгоритм

for each()).

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

**Задача 2:**

1. Создать адаптер контейнера. Адаптер контейнера - очередь с приоритетами

2. Заполнить его элементами пользовательского типа Time (класс Time для работы с временными интервалами. Интервал должен быть

представлен в виде двух полей: минуты типа int и секунды типа int. при выводе минуты

отделяются от секунд двоеточием. Реализовать:

добавление секунд (учесть, что в минуте не может быть более 60 секунд)

вычитание секунд). Для пользовательского типа перегрузить необходимые операции.

3. Найти элемент с заданным ключом и добавить его на заданную позицию контейнера(использовать алгоритмы

replace if(), replace copy(), replace copy\_if(), fill()).

4. Найти элемент с заданным ключом и удалить его из контейнера(использовать алгоритмы

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

5. Отсортировать контейнер по убыванию и по возрастанию ключевого поля

(использовать алгоритм sort () ).

6. Найти в контейнере заданный элемент (использовать алгоритмы find () ,

find if(), count(), count if()).

7. Найти разницу между максимальным и минимальным элементами контейнера и

вычесть ее из каждого элемента

контейнера(использовать алгоритм

for each()).

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

**Задача 3:**

1. Создать ассоциативный контейнер. Ассоциативный контейнер - словарь.

2. Заполнить его элементами пользовательского типа Time (класс Time для работы с временными интервалами. Интервал должен быть

представлен в виде двух полей: минуты типа int и секунды типа int. при выводе минуты

отделяются от секунд двоеточием. Реализовать:

добавление секунд (учесть, что в минуте не может быть более 60 секунд)

вычитание секунд). Для пользовательского типа перегрузить необходимые операции.

3. Найти элемент с заданным ключом и добавить его на заданную позицию контейнера(использовать алгоритмы

replace if(), replace copy(), replace copy\_if(), fill()).

4. Найти элемент с заданным ключом и удалить его из контейнера(использовать алгоритмы

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

5. Отсортировать контейнер по убыванию и по возрастанию ключевого поля

(использовать алгоритм sort () ).

6. Найти в контейнере заданный элемент (использовать алгоритмы find () ,

find if(), count(), count if()).

7. Найти разницу между максимальным и минимальным элементами контейнера и

вычесть ее из каждого элемента

контейнера(использовать алгоритм

for each()).

8. Для выполнения всех заданий использовать стандартные алгоритмы библиотеки STL

**UML-Диаграмма**





