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: Лабораторная работа №13

"Стандартные обобщенные алгоритмы библиотеки STL"

Семестр 2

Выполнил работу

Студент группы РИС-22-1Б

Баженов Т.И.

Проверил

Доцент кафедры ИТАС

Полякова О.А.

Г. Пермь-2023

# Постановка задачи

Вариант 15

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());

4. Удалить элементы в соответствии с заданием (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. Выполнить задание варианта для полученного контейнера (использовать алгоритм for\_each()) ;

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

Условия варианта:

1. Контейнер - список;

2. Тип элементов Pair (см. лабораторную работу №3).

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());

4. Удалить элементы в соответствии с заданием (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 элемент с заданным ключевым полем (использовать алгоритмы find(), find\_if(), count(), count\_if());

7. Выполнить задание варианта для полученного контейнера (использовать алгоритм for\_each()) ;

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

Условие варианта:

Адаптер контейнера – очередь с приоритетами.

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());

4. Удалить элементы в соответствии с заданием (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 элемент с заданным ключевым полем (использовать алгоритмы find(), find\_if(), count(), count\_if());

7. Выполнить задание варианта для полученного контейнера (использовать алгоритм for\_each());

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

Условие варианта:

Ассоциативный контейнер - словарь.

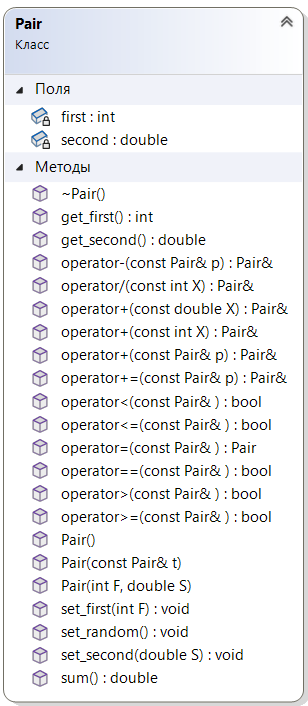
Задания варианта ко всем задачам:

1. Найти среднее арифметическое и добавить его в конец контейнера;

2. Найти элементы ключами из заданного диапазона и удалить их из контейнера;

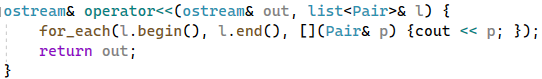
3. К каждому элементу добавить сумму минимального и максимального элементов контейнера.

# UML-диаграмма класса Pair

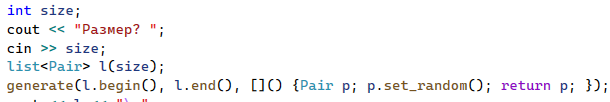


# Задача 1

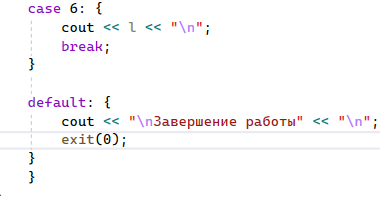
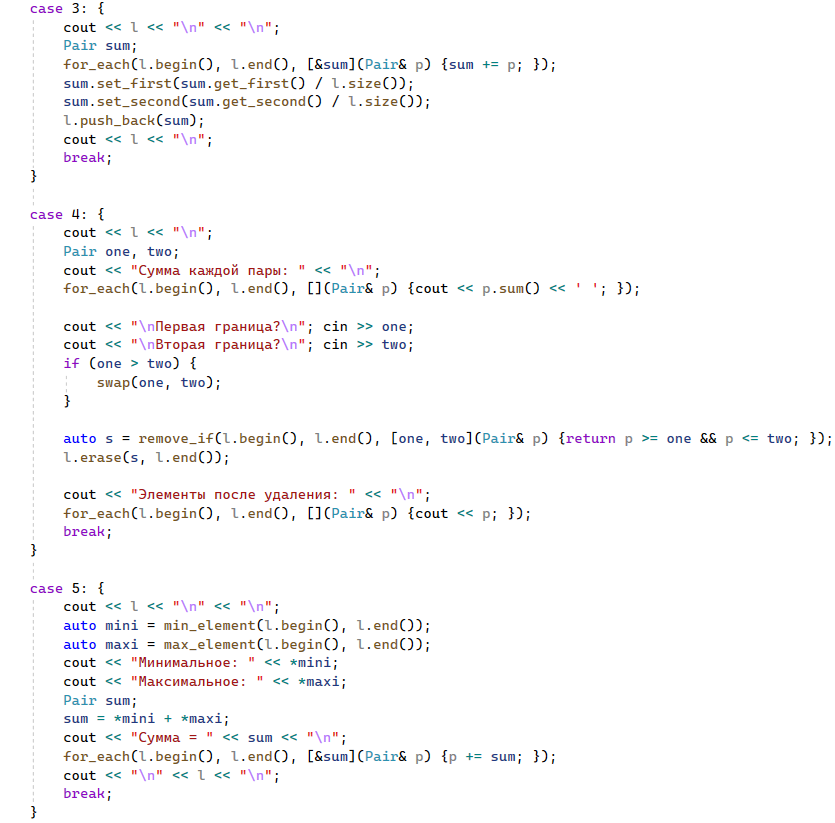
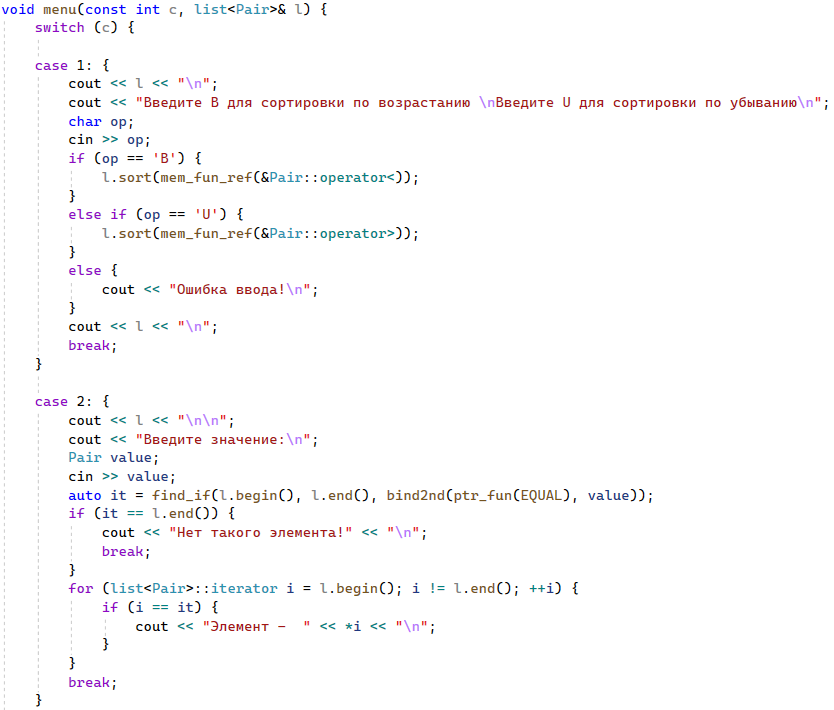
Перегрузка оператора “<<”:



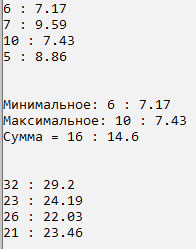
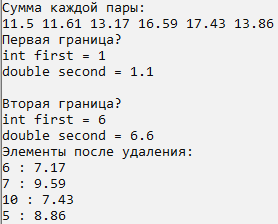
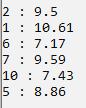
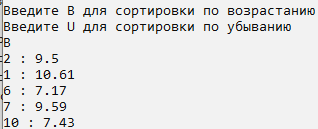
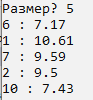
Генерация контейнера:



Методы menu():

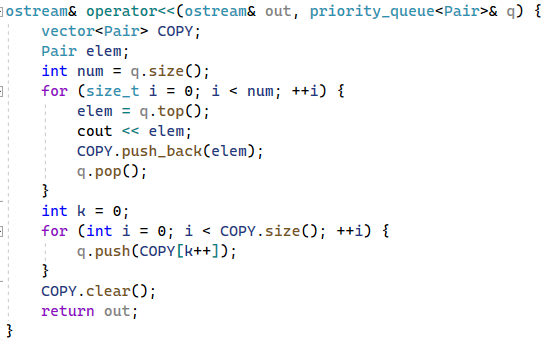


Работа программы:

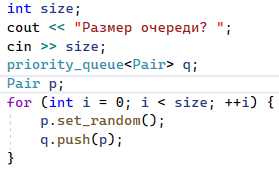


# Задача 2

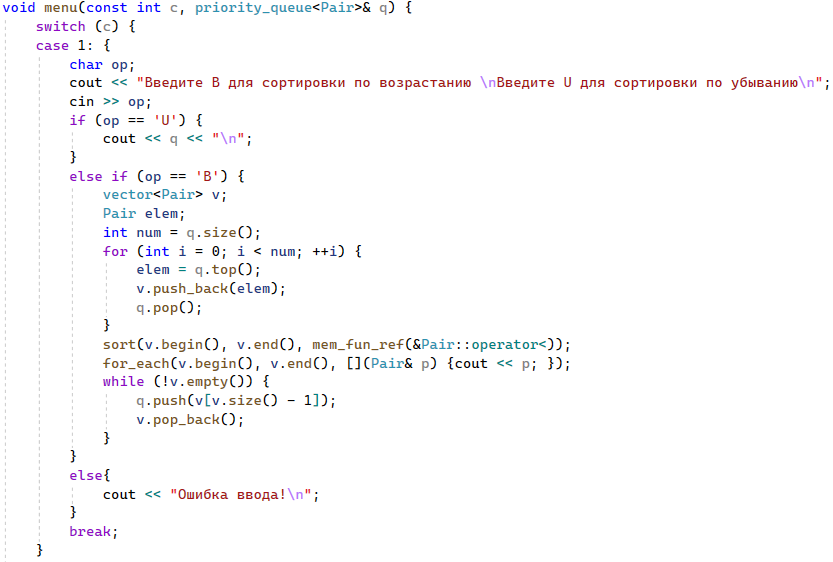
Перегрузка оператора “<<”:

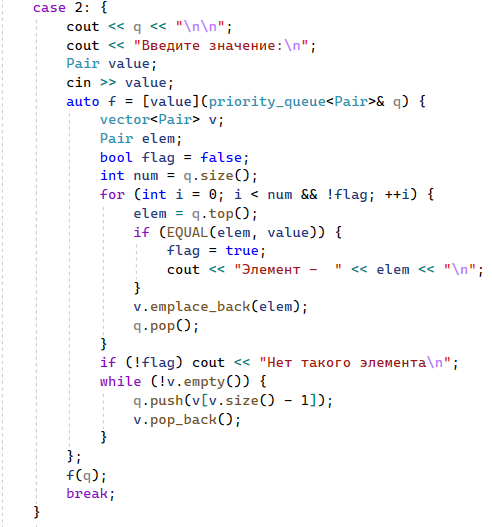


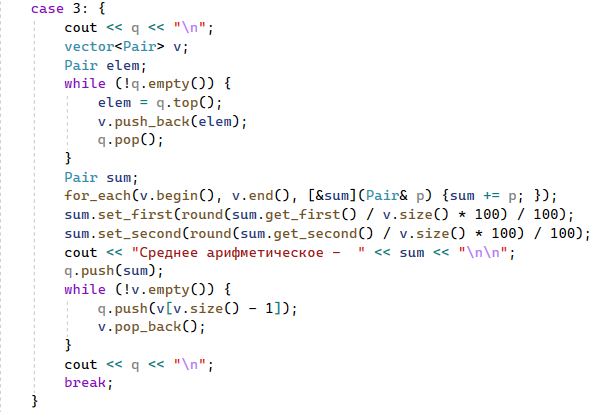
Генерация контейнера:

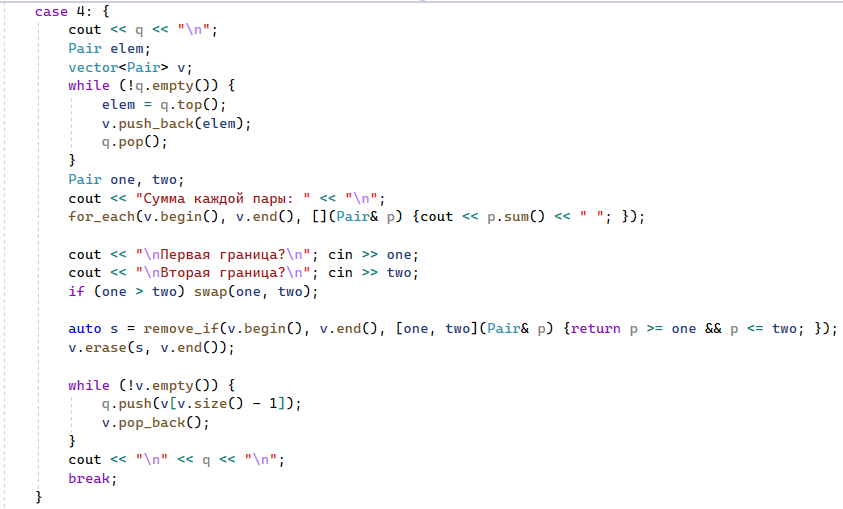


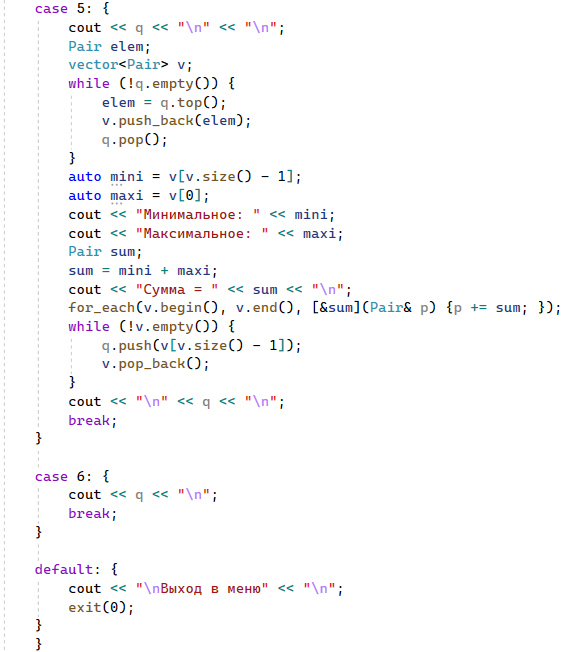
Методы menu():



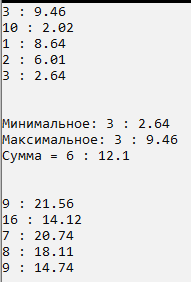
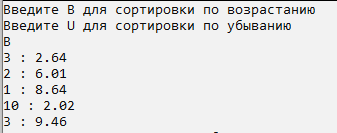
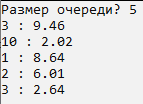






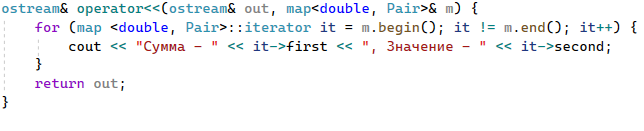


Работа программы:

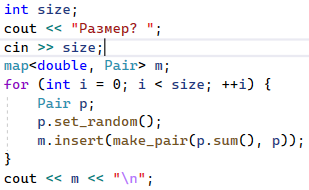


# Задача 3

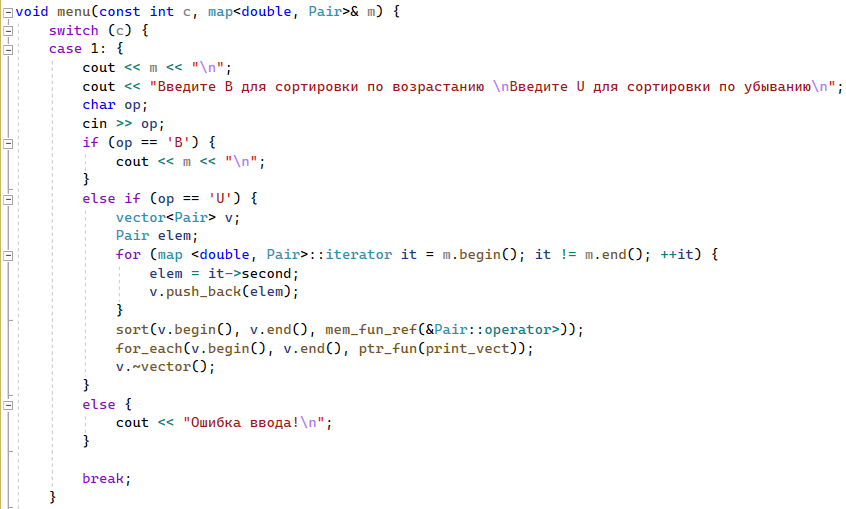
Перегрузка оператора “<<”:

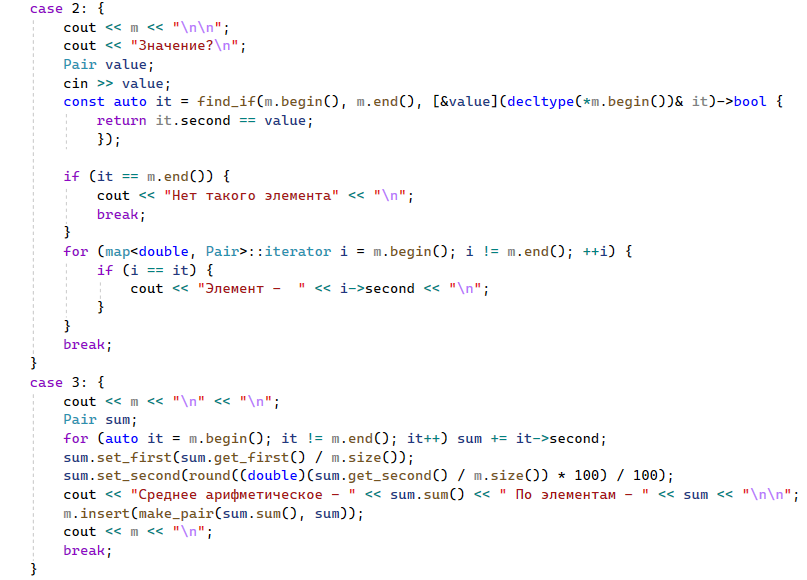


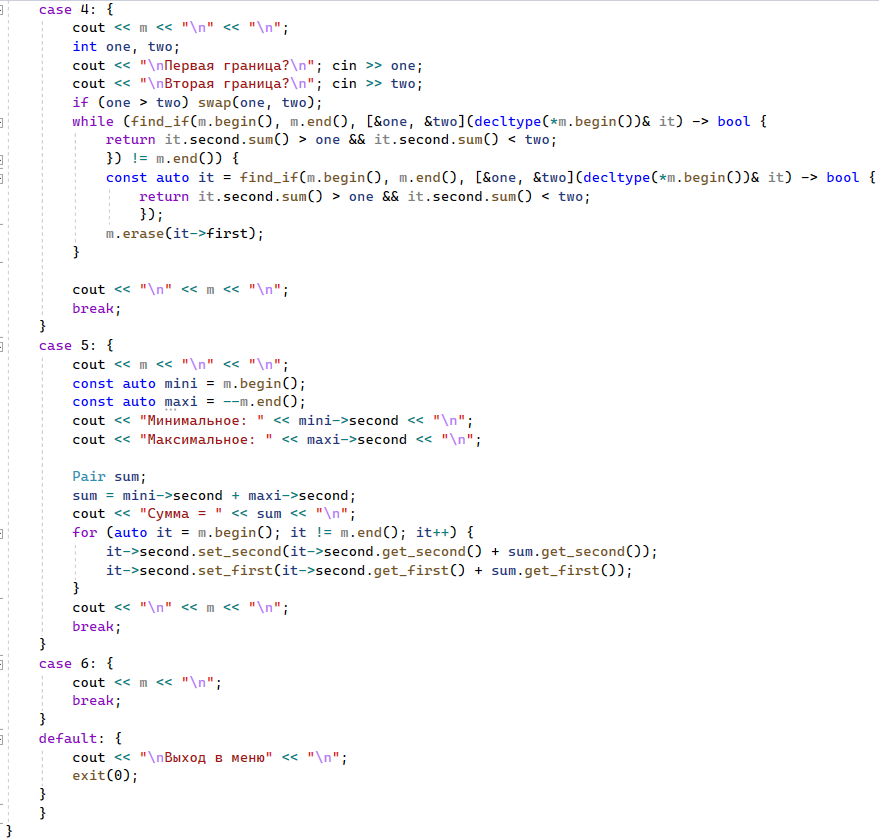
Генерация контейнера:



Методы menu():







Работа программы:

