

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
**«Сибирский государственный университет науки и технологий  
имени академика М.Ф. Решетнева»**

Институт информатики и телекоммуникаций

Кафедра информатики и вычислительной техники

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ**

Языки программирования

**Лабораторная работа №4 Контейнеры и алгоритмы STL**

---

Руководитель

\_\_\_\_\_

подпись, дата

**А.В. Проскурин**

\_\_\_\_\_  
инициалы, фамилия

Обучающийся **БПИ23-02, 23151451**

\_\_\_\_\_  
номер группы, зачетной книжки

\_\_\_\_\_

подпись, дата

**С.А. Черкашин**

\_\_\_\_\_  
инициалы, фамилия

Красноярск 2024 г.

## **ЦЕЛЬ РАБОТЫ**

Получение практических навыков разработки и отладки программ по обработке строк с использованием стандартной библиотеки шаблонов (STL)

## ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Ознакомиться с общей постановкой задачи.
2. Ознакомиться с вариантом задания – соответствует вашему номеру в списке группы (при нехватке заданий вариант задания вычисляется как номер\_в\_списке\_группы - количество\_заданий).
3. Разработать классы согласно варианту задания.
4. Написать и отладить программу на подготовленных наборах тестовых данных.
5. Подготовить отчет по лабораторной работе. Отчет должен включать в себя:
  - a. титульный лист;
  - b. цель лабораторной работы;
  - c. постановку задачи;
  - d. схему наследования классов (UML диаграмма классов);
  - e. текст программы с комментариями;
  - f. демонстрацию работы программы (Снимки экрана при выполнении действий программы с описанием).
  - g. краткие ответы на контрольные вопросы;
  - h. выводы по лабораторной работе.
6. Защитить лабораторную работу перед преподавателем.

## ПОСТАНОВКА ЗАДАЧИ

Необходимо реализовать класс (а также при необходимости шаблонные функции), который будет выполнять обработку текстового файла согласно варианту задания.

При реализации необходимо использовать контейнеры и алгоритмы стандартной библиотеки шаблонов (STL) подходящие для решения задачи (например, контейнеры: `vector`, `map`, `set`, `list`, и алгоритмы: `find`, `sort`, и т.п.), а также методы класса `string`.

В зависимости от оценки, на которую вы претендуете, необходимо выполнить следующие задания (Для каждой следующей оценки нужно выполнить ВСЕ предыдущие задания, если обратное не указано явно):

Оценка	Общее задание
Удовлетворительно	1. Написать класс, выполняющий обработку текста согласно варианту.
Хорошо	2. Реализовать работу с текстовым файлом: исходные данные берутся из одного файла, результат записывается в другой
Отлично	3. С помощью функций реализовать меню для организации взаимодействия с пользователем. С помощью меню необходимо выполнить демонстрацию функционирования всей программы (т.е. после выполнения действия происходит вывод меню до тех пор, пока в меню не будет выбран пункт «Закончить работу с программой»).

### Вариант №24.

Формулировка задания: на основании таблицы результатов некоторого шахматного турнира (квадратная матрица  $A[N][N]$ ), в котором участвовало  $N$  шахматистов ( $N > 4$ ) необходимо вывести ФИО участников и набранные очки, в порядке убывания набранных ими очков. Подсчёт очков проводится следующим образом: за выигрыш дается 1 очко, за ничью - 0,5 очка, за проигрыш – 0 очков.

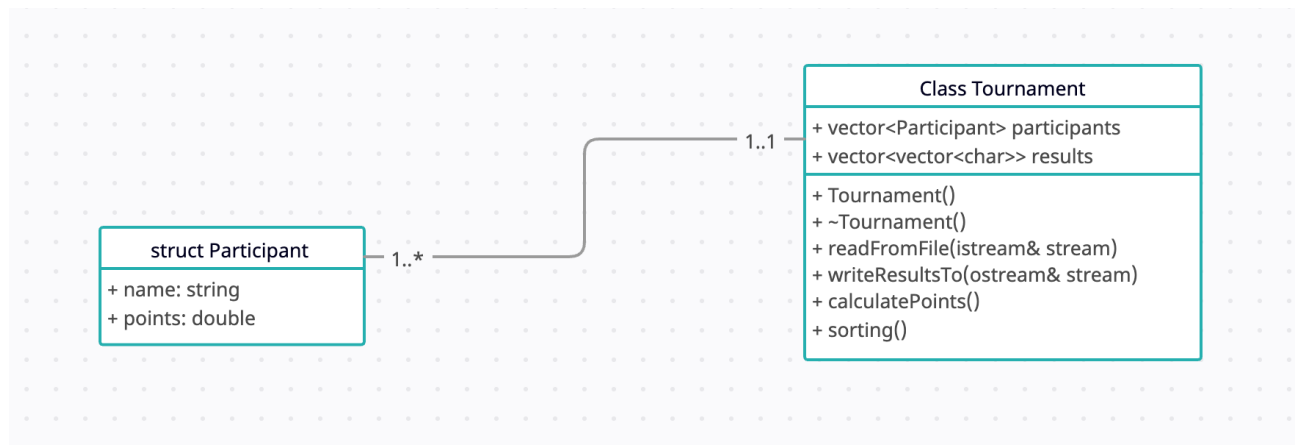
Формат входных данных: текстовый файл, содержащий таблицу с результатами турнира, где в первой строке указано сколько было участников, во второй перечислены ФИО участников, дальше содержание в виде обозначений:

$A[i][j] = В$ , если  $i$ -ый участник выиграл у  $j$ -того (при этом  $A[j,i] = П$ );

$A[i][j] = Н$ , если  $i$ -ый и  $j$ -ый участник сыграли вничью;

$A[i][i] = Х$ .

## UML ДИАГРАММА



# ХОД РАБОТЫ

## Tournament.h

```
Lab4 > src > lib > h Tournament.h > ...
1  #pragma once
2  #include <iostream>
3  #include <string>
4  using namespace std;
5  Codeium: Refactor | Explain
6  class Tournament
7  {
8  private:
9      Codeium: Refactor | Explain
10     struct Participant
11     {
12         string name;
13         double points;
14     };
15     vector<Participant> participants;
16     vector< vector<char> > results;
17 public:
18     Tournament();
19     ~Tournament();
20     void readFromFile(istream& stream);
21     void writeResultsTo(ostream& stream) const;
22     void calculatePoints();
23     void sorting();
24 };
```

## Tournament.cpp

```
Lab4 > src > lib > C++ Tournament.cpp > ...
1  #include </Users/kalmar4ic/Documents/VSCodeC++/Lab4/src/lib/Tournament.h>
2  #include <algorithm>
3  using namespace std;
4
5  Codeium: Refactor | Explain | Generate Function Comment | X
6  Tournament::Tournament()
7  {
8  }
9
10 Codeium: Refactor | Explain | Generate Function Comment | X
11 Tournament::~~Tournament()
12 {
13     participants.clear();
14     results.clear();
15 }
16
17 Codeium: Refactor | Explain | Generate Function Comment | X
18 void Tournament::readFromFile(istream &stream)
19 {
20     int size;
21     stream >> size;
22     stream.ignore();
23     participants.resize(size);
24     for (int i = 0; i < participants.size(); i++)
25     {
26         if (i == participants.size() - 1)
27         {
28             getline(stream, participants[i].name, '\n');
```

```
C++ main.cpp  h Tournament.h  C++ Tournament.cpp ×  1.txt

Lab4 > src > lib > C++ Tournament.cpp > ...
15 void Tournament::readFromFile(istream &stream)
21     for (int i = 0; i < participants.size(); i++)
23         if (i == participants.size() - 1)
25             getline(stream, participants[i].name, '\n');
26         }
27         else
28         {
29             getline(stream, participants[i].name, '|');
30         }
31         participants[i].points = 0;
32     }
33
34     results.resize(size, vector<char>(size));
35     for (int i = 0; i < size; i++)
36     {
37         for (int j = 0; j < size; j++)
38         {
39             stream >> results[i][j];
40         }
41     }
42 }
43
Codeium: Refactor | Explain | Generate Function Comment | X
44 void Tournament::writeResultsTo(ostream &stream) const
45 {
46     for (int i = 0; i < participants.size(); i++)
47     {
48         stream << participants[i].name << " " << participants[i].points << endl;
49     }
50 }
51
Codeium: Refactor | Explain | Generate Function Comment | X
52 void Tournament::calculatePoints()
53 {
54     for (int i = 0; i < results.size(); i++)
55     {
56         for (int j = 0; j < results[i].size(); j++)
57         {
58             if (results[i][j] == 'W')
59             {
60                 participants[i].points += 1;
61             }
62             else if (results[i][j] == 'D')
63             {
64
Строка 76, столбец 1  Пробелов: 4  UTF-8  LF  {} C++  Codeium: {...}  Mac

C++ main.cpp  h Tournament.h  C++ Tournament.cpp ×  1.txt

Lab4 > src > lib > C++ Tournament.cpp > sorting()
52 void Tournament::calculatePoints()
54     for (int i = 0; i < results.size(); i++)
56         for (int j = 0; j < results[i].size(); j++)
58             if (results[i][j] == 'W')
59             {
60                 participants[i].points += 1;
61             }
62             else if (results[i][j] == 'D')
63             {
64                 participants[i].points += 0.5;
65             }
66         }
67     }
68 }
69
Codeium: Refactor | Explain | Generate Function Comment | X
70 void Tournament::sorting()
71 {
72     sort(participants.begin(), participants.end(),
73         [](const Participant& a, const Participant &b) { return a.points > b.points; });
74 }
```

## Main.cpp

```
Lab4 > src > C++ main.cpp > main()
1  #include <iostream>
2  #include <vector>
3  #include <fstream>
4  #include <string>
5  using namespace std;
6  #include "/Users/kalmar4ic/Documents/VSCodeC++/Lab4/src/lib/Tournament.h"
7
8  Codeium: Refactor | Explain | Generate Function Comment | X
9  int main()
10
11      Tournament tournament;
12      ifstream file;
13      ofstream file2;
14      file.open("1.txt");
15      file2.open("2.txt");
16      int choice;
17      while (choice != 5)
18      {
19          cout << "Выберите 1, чтобы создать турнир из файла" << endl;
20          cout << "Выберите 2, чтобы вывести результаты" << endl;
21          cout << "Выберите 3, чтобы отсортировать результаты по порядку убывания" << endl;
22          cout << "Выберите 4, чтобы записать результаты в файл" << endl;
23          cout << "Выберите 5, чтобы выйти из программы" << endl;
24          cin >> choice;
25          cin.ignore();
26          if (choice == 1)
27          {
28              tournament.readFromFile(file);
29              tournament.calculatePoints();
30          }
31          else if (choice == 2)
32          {
33              tournament.writeResultsTo(cout);
34          }
35          else if (choice == 3)
36          {
37              tournament.sorting();
38          }
39          else if (choice == 4)
40          {
41              tournament.writeResultsTo(file2);
42          }
43      }
44      file.close();
45
46  Строка 46, столбец 2  Пробелов: 4  UTF-8  LF  {} C++  Codeium: {...}  Mac
```

```
C++ main.cpp x  h Tournament.h  C++ Tournament.cpp  1.txt
Lab4 > src > C++ main.cpp > main()
8  int main()
16      while (choice != 5)
38          else if (choice == 4)
41      {
42      }
43      file.close();
44      file2.close();
45      return 0;
46
```



## Работоспособность программы

Ввод из файла:

1.txt

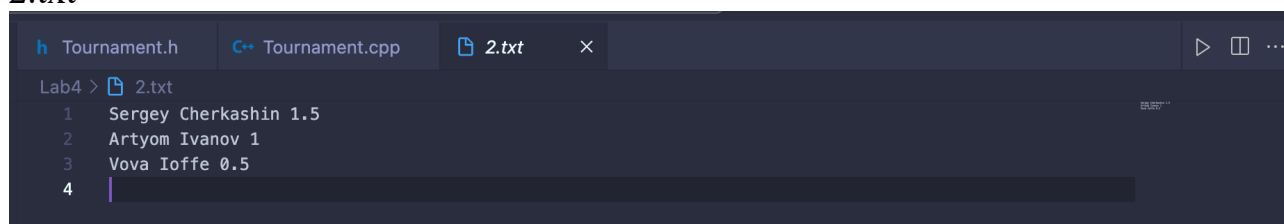
```
Tournament.h Tournament.cpp 1.txt x
Lab4 > 1.txt
1 3
2 Artyom Ivanov|Sergey Cherkashin|Vova Ioffe
3 X L W
4 W X D
5 L D X
```

```
ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ КОММЕНТАРИИ КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ ПОРТЫ + - make - Lab4
g++ -c src/main.cpp -o out/main.o
g++ out/lib/Tournament.o out/main.o -o out/main
o kalmar4ic@MacBook-Air-Sergej Lab4 % make run
./out/main
Выберите 1, чтобы создать турнир из файла
Выберите 2, чтобы вывести результаты
Выберите 3, чтобы отсортировать результаты по порядку убывания
Выберите 4, чтобы записать результаты в файл
Выберите 5, чтобы выйти из программы
1
Выберите 1, чтобы создать турнир из файла
Выберите 2, чтобы вывести результаты
Выберите 3, чтобы отсортировать результаты по порядку убывания
Выберите 4, чтобы записать результаты в файл
Выберите 5, чтобы выйти из программы
2
Artyom Ivanov 1
Sergey Cherkashin 1.5
Vova Ioffe 0.5
Выберите 1, чтобы создать турнир из файла
Выберите 2, чтобы вывести результаты
Выберите 3, чтобы отсортировать результаты по порядку убывания
Выберите 4, чтобы записать результаты в файл
Выберите 5, чтобы выйти из программы
█
Строка 5, столбец 6 Пробелов: 4 UTF-8 LF Простой текст Codeium: {...}
```

Сортировка по убыванию:

```
ПРОБЛЕМЫ ВЫХОДНЫЕ ДАННЫЕ КОММЕНТАРИИ КОНСОЛЬ ОТЛАДКИ ТЕРМИНАЛ ПОРТЫ + - make - Lab4
Выберите 3, чтобы отсортировать результаты по порядку убывания
Выберите 4, чтобы записать результаты в файл
Выберите 5, чтобы выйти из программы
2
Artyom Ivanov 1
Sergey Cherkashin 1.5
Vova Ioffe 0.5
Выберите 1, чтобы создать турнир из файла
Выберите 2, чтобы вывести результаты
Выберите 3, чтобы отсортировать результаты по порядку убывания
Выберите 4, чтобы записать результаты в файл
Выберите 5, чтобы выйти из программы
3
Выберите 1, чтобы создать турнир из файла
Выберите 2, чтобы вывести результаты
Выберите 3, чтобы отсортировать результаты по порядку убывания
Выберите 4, чтобы записать результаты в файл
Выберите 5, чтобы выйти из программы
2
Sergey Cherkashin 1.5
Artyom Ivanov 1
Vova Ioffe 0.5
Выберите 1, чтобы создать турнир из файла
Выберите 2, чтобы вывести результаты
Выберите 3, чтобы отсортировать результаты по порядку убывания
Строка 5, столбец 6 Пробелов: 4 UTF-8 LF Простой текст Codeium: {...}
```

Запись результата в файл:  
2.txt



The screenshot shows a code editor window with three tabs: 'Tournament.h', 'Tournament.cpp', and '2.txt'. The '2.txt' tab is active, showing a list of names and scores. The text is as follows:

```
Lab4 > 2.txt
1 Sergey Cherkashin 1.5
2 Artyom Ivanov 1
3 Vova Ioffe 0.5
4 |
```

## ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Охарактеризуйте класс `vector`, как происходит обращение к элементам? Класс `vector` в C++ представляет собой динамический массив, который может изменять свой размер. Обращение к элементам осуществляется с помощью оператора индексирования/

2. Как происходит добавление/удаление элементов в классе `vector`? Добавление происходит с помощью метода `push_back()` (Добавляет элемент в конце) и методом `insert(Место, элемент)` (добавляет элемент в любое место).

Удаление происходит с помощью метода `pop_back()` (Удаление с конца, при этом размер сокращается, а ёмкость нет). Удаление из любого места – `erase()`. Удаление всех элементов – `clear()` (Ёмкость также не изменяется, а размер 0)

3. При помощи какого метода можно указать количество элементов в классе `vector`?

Выделение памяти с помощью `reserve()`

Кол-во элементов можно указать методом `size()`, а всю ёмкость массива – `capacity()`.

4. При помощи какого метода можно изменить количество элементов в классе `vector`, как при этом будут происходить изменения?

Чтобы изменить количество элементов в классе `vector`, используется метод `resize(size)`. Если указанный размер больше текущего, новые элементы инициализируются значением по умолчанию. Если меньше, лишние элементы удаляются.

5. Охарактеризуйте класс `array`, как осуществляется его передача в функцию.

Класс `array` в C++ представляет собой статический массив фиксированного размера.

По значению: `void func(std::array<int, 5> arr)` - копия массива.

По константной ссылке: `void func(const std::array<int, 5>& arr)` - эффективный способ, без копирования, но массив нельзя изменить внутри функции.

По ссылке: `void func(std::array<int, 5>& arr)` - позволяет изменять массив внутри функции.

6. Поясните, что из себя представляют контейнеры `set/multiset`, приведите плюсы и минусы использования `set` и `multiset`.

**Set** - упорядоченное множество уникальных элементов. Обеспечивает стандартные операции над множествами (объединение, пересечение, вычитание).

**Multiset** - то же что и `set`, но позволяет хранить повторяющиеся элементы.

Плюсы:

- Автоматическая сортировка.

Минусы:

- Поиск, вставка и удаление медленнее, чем в `'vector'`.

- Чуть больше расходы памяти из-за структуры данных.

- нельзя обратиться по `i` элементу.

7. Охарактеризуйте контейнер `list`. Приведите методы для добавления/удаления и обращения элементам.

`List` - Двусвязный список, элементы которого хранятся в произвольных кусках памяти.

Добавление элемента:

- `push_back(val)`: добавляет значение `val` в конец списка.
- `push_front(val)`: добавляет значение `val` в начало списка.
- `insert(pos, val)`: вставляет элемент `val` на позицию, на которую указывает итератор `pos`. Возвращает итератор на добавленный элемент

Удаление элемента:

- `clear()`: удаляет все элементы.
- `pop_back()`: удаляет последний элемент.
- `pop_front()`: удаляет первый элемент.
- `erase(pos)`: удаляет элемент, на который указывает итератор `pos`. Возвращает итератор на элемент, следующий после удаленного.

8. Что такое итератор, для чего он нужен. Приведите пример (с применением синтаксиса) использования итераторов.

**Итератор** — это класс, объекты которого выполняют такую же роль по отношению к контейнеру, как указатели по отношению к массиву. Указатель может использоваться в качестве средства доступа к элементам массива, а итератор — в качестве средства доступа к элементам контейнера.

```
vector<int> vec = {1, 2, 3, 4, 5};
for (auto it = vec.begin(); it != vec.end(); it++) {
    std::cout << *it << " ";
}
```

9. Охарактеризуйте контейнер `map`. Как происходит работа с контейнером `map`?

`map` — это контейнер, который хранит пары ключ-значение. Доступ к элементам осуществляется по ключу, например: `myMap[key]`. При добавлении нового ключа автоматически создаётся соответствующее значение.

10. Как проверить существование ключа в контейнере `map`?

Существование ключа можно проверить с помощью метода `count()` или с помощью `find()`:

```
if (m.count("apple")) {
    // Ключ существует
}
if (m.find("banana") != m.end()) {
    // Ключ существует
}
```

11. Поясните принцип и работу циклов `foreach`, какая форма записи у данных циклов?

Цикл `foreach` в C++ реализуется с помощью цикла `for` с диапазоном.

`For_each(v.begin(), v.end(), что-то делаем)`

12. Охарактеризуйте методы `substr` и `replace` класса `string`. Поясните принцип объединения строк.

`substr(pos, len)` - возвращает подстроку, начиная с позиции `pos` длиной `len`.

`replace(pos, len, str)` - заменяет подстроку, начиная с позиции `pos` длиной `len`, на строку `str`.

Для объединения строк можно использовать оператор `+` или метод `append()`.

13. При помощи каких методов происходит добавление и вставка фрагмента в классе `string`. Перечислите методы подсчета количества символов строке `string`.

Добавление в строку осуществляется с помощью методов `append` и `push_back`.

Вставка фрагмента — методом `insert`. Для подсчёта символов используется метод `size()` или `length()`. Так же можно использовать `strlen(string)`

14. Приведите синтаксис поиска первого вхождения другой строки (подстроки) в заданную строку.

`find(str)` - возвращает позицию первого вхождения подстроки `str`.

## **ВЫВОДЫ**

Было изучено, что такое библиотека стандартных шаблонов, как использовать шаблоны внутри этой библиотеки, а также написана программа с использованием одного из шаблонов.