C++

Стандартная библиотека

Черновик

Кафедра ИВТ и ПМ

2019

План

Outline

Файлы

```
#include <fstream>
using namespace std;
...
// создать объект для записи в файл
// и открыть текстовый файл для записи
ofstream f("myfile");
// запись в файл
// здесь все данные будут записаны слитно. так лучше не делать
f << "qwerty";
f << 123;
f << 3.14;
f << endl; // записать символ перехода на новую строку
f << 42.5;
f.close();
```

Содержимое созданного файла:

```
qwerty1233.14
42.5
```

https://en.cppreference.com/w/cpp/io/basic

Файлы

```cpp
#include <fstream>
using namespace std;

// создать экземпляр класса ifstream (для чтения файлов)
ifstream f1;
// открыть текстовый файл
f1.open("myfile");
if (f1.is_open()){
string s;
f1 >> s; // s = "qwerty1233.14"
...
f1 >> s; // s = "42.5"
float number = stof(s); // строка -> число
f1.close();
}
```

https://en.cppreference.com/w/cpp/io/basic_ifstream

Outline

exception

...

Контейнеры

Некоторые контейнеры

- list - двусвязный список
- vector - динамический массив
- map - ассоциативный массив (словарь)
- stack - стэк
- queue - очередь
- pair - пара

Классы контейнеров объявлены в заголовочных файлах с соответствующими именами. Например класс list объявлен в заголовочном файле list.

```
# include <list>
```

Outline

Умные указатели

Проблема

```
class MyClass{
    //..
public:
    void foo(){}
    int number;          };

void bar(){
    // статическое создание. Объект будет уничтожен после выхода из функции
    MyClass  c1;
    MyClass *c2 = new MyClass();     // Указатель на класс
    C2 = c2;
    c2->number = 22;
    MyClass *c3;                     // Указатель на класс
    c3 = new MyClass();              // память можно выделить и потом
    c3->number = 33;
    C3 = c3;
    // нужно не забыть освободить память по адресу c2 и c3
    // если дальше будет много условных операторов, точек выхода из функции
    // и исключений, то предусмотреть освобождение выделенной памяти не всегда
    // а если получится, то код может быть загромождён
}
int main(){
    bar();
    return 0;   }
```

Умные указатели

Решение

```
#include <memory>
using namespace std;
class MyClass{
    //..
public:
    void foo(){}
    int number;         };

void bar(){
    // Умный указатель - это класс для управления памятью.
    // Экземпляр этого класса создаётся статически. Он сам позаботится о том,
    // чтобы освободить память другого объекта, созданного динамически.
    // Инициализировать умный указатель нужно сразу. RAII
    unique_ptr<MyClass> c4 (new MyClass());  /
    // не смотря на то, что c4 это не MyClass*, используется оператор ->
    c4->foo();
    c4->number = 42;
    // перед уничтожением c4, будет вызван его деструктор, он и освободит памят
    // занимаемую экземпляром класса MyClass, которой он владеет
}

int main(){
    bar();
    return 0;   }
```

Умные указатели

Шпаргалка по использованию умных указателей в C++
https://eax.me/cpp-smart-pointers/

Outline

Outline

vector

Класс **vector** предоставляет удобный интерфейс для работы с динамическим массивом.

```
#include <vector>
using std::vector;

// пустой вектор типа int
vector<int> myVector;
// зарезервировали память под 10 элементов
myVector.reserve(10);
```

vector

```
typedef vector<float> vectorf;  // лучше создать синоним
unsigned n = 128;
vector<float> v; // можно не указывать размер
vector<float> v2(n); // а можно указывать

vectorf v3(128, 0);  // легко инициализировать нулём
vectorf v4 = {1,2,3,4};  // легко инициализировать массивом

v4.resize(10, 9);

//  cout << v3 << endl; // Так печатать нельзя :(
// вывод значений на экран
for (auto i=0; i<v4.size(); i++)
    cout << v4[i] << " ";
cout << endl;
```

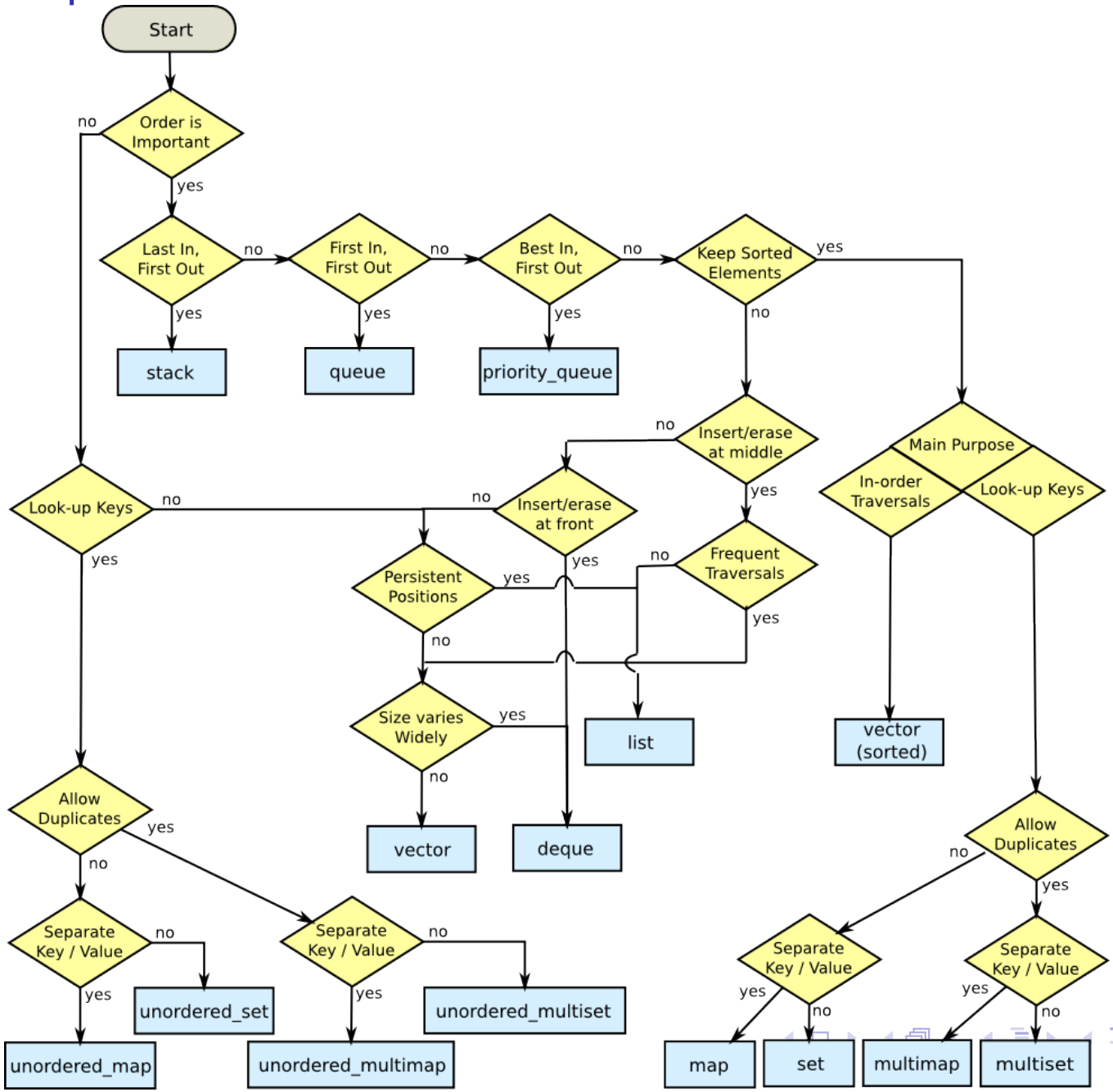
vector. методы

методы и операторы класса vector

- at(индекс) - возвращает элемент по индексу
- индекс возвращает элемент по индексу
- empty() - возвращает true если вектор пуст
- size() - возвращает размер вектора
- clear() - очищает вектор
- pop_back() возвращает последний элемент; элемент удаляется из вектора
- push_back(значение) добавляет значение в конец вектора
- Resize(n, нач_значение) -изменяет размер вектора
- front() - возвращает первый элемент
- back() - возвращает последний элемент

Outline

Какой выбрать



Сравнение

Container	Insertion	Access	Erase	Find	Persistent Iterator
vector / string	Back: O(1) or O(n) Other: O(n)	O(1)	Back: O(1) Other: O(n)	Sorted: O(log n) Other: O(n)	No
deque	Back/Front: O(1) Other: O(n)	O(1)	Back/Front: O(1) Other: O(n)	Sorted: O(log n) Other: O(n)	Pointers only
list / forward_list	Back/Front: O(1) With iterator: O(1) Index: O(n)	Back/Front: O(1) With iterator: O(1) Index: O(n)	Back/Front: O(1) With iterator: O(1) Index: O(n)	O(n)	Yes
set / map	O(log n)	-	O(log n)	O(log n)	Yes
unordered_set / unordered_map	O(1) or O(n)	O(1) or O(n)	O(1) or O(n)	O(1) or O(n)	Pointers only
priority_queue	O(log n)	O(1)	O(log n)	-	-

Outline

Outline

Ссылки и литература

1. **Stepik: Программирование на языке C++**
2. **Б. Страуструп Язык программирования C++.** 2013. 350 страниц. Учебник по языку. Шаблоны. ООП. Проектирование.
3. MSDN: Справочник по языку C++
4. Эффективный и современный C++: 42 рекомендации по использованию C++ 11 и C++14.
   2016. 300 страниц. Просмотреть. Изучить. Использовать как справочник. Неформальный стиль. Много примеров. Хорошее знание C++.
5. www.stackowerflow.com - система вопросов и ответов

Ссылки и литература

Документация по языку:

- ru.cppreference.com - информация по языку и стандартной библиотеке C++. Есть примеры.

Дополнительно:

- habr.com/company/pvs-studio/ Блог компании PVS-Studio. Примеры ошибок в C++ (и не только) коде найденных статическим анализатором кода PVS-Studio.

Ссылки и литература

Ссылка на слайды
github.com/VetrovSV/OOP