C++: Статические поля и методы

24 мая 2017 г.

Задание

Создайте класс Point для точки на плоскости.

В классе должны быть:

- поля для хранения координат x и y;
- конструктор по умолчанию;
- конструктор с параметрами;
- аксессоры.

Реализуйте возможность подсчитывать количество созданных точек.

Статическое поле

- объявляется с помощью ключевого слова static
- одно на все объекты класса
- определение выносится за пределы определения класса
- может использоваться без объекта класса

```
static int counter;
/* … */
int  MyClass::counter = 0;
```

Статический метод

- объявляется с ключевым словом static
- может использоваться без объекта класса
- определение метода можно задать как в классе, так и вне его
- внутри метода нельзя напрямую обращаться к полям
- метод не может быть константным

```
static int getCounter();
```

Статические поля и методы: пример

```
class MyClass
{
    static int counter;
public:
   MyClass() { ++counter; }
   ~MyClass() { --counter; }
   static int getCounter() { return counter; };
} ;

int MyClass::counter = 0;
```

Задание

Добавьте в класс Point статическое поле, в котором должно храниться количество существующих точек, и статический метод для доступа к этому полю.

Делегирующий конструктор

- вызывает другой конструктор в том же классе, чтобы «перепоручить» работу
- объект считается созданным после выполнения первого конструктора

```
class MyClass {
    public:
        MyClass(double x) { … }
        MyClass(int x) : MyClass(double(x))
        {
                // do something
        }
};
```

Делегирующий конструктор: пример использования

```
class Rectangle {
        size_t width_;
        size_t height_;
public:
        Rectangle(size_t width, size_t height):
           width_(width),  height_(height)
                { std::cout  <<  "Target ctor" << std::endl; }

        Rectangle(size_t width):
           Rectangle(width, width)
                { std::cout  <<  "Delegate ctor" << std::endl; }
};
```

Задание

Добавьте в класс Point делегирующий конструктор для случая, когда пользователь передает только одну координату.

Динамические массивы объектов

В классе должен существовать конструктор без параметров (например, со значениями по умолчанию).

```
class Point{
      int x_, y_;
public:
      Point(int x = 1, int y = 1) : x_(x), y_(y)  {  }
 };

Point* pointArray = new Point[10];
```

Статические массивы объектов

В классе не обязательно должен существовать конструктор без параметров.

```
class Point{
      int x_;
      int y_;
public:
      Point(int x, int y) : x_(x), y_(y)  {  }
};

Point pointArray1[2] = {Point(2, 3), Point(10, 20)};
Point pointArray2[2];
```

Статические массивы объектов (C++11)

Можно создать массив типа std::array.

```
class Point{
      int x_, y_;
public:
      Point() : x_(0), y_(0) { }
      Point(int x, int y) : x_(x), y_(y)  {  }
 };

std::array<Point, 2> pointArray1;
std::array< Point, 2> pointArray2 = {Point(2, 3), Point(10, 20)};
```

std::array

```
#include <array>
std::array<тип_значений, к-во_элементов> имя;
```

- последовательный контейнер фиксированного размера (размер задается при компиляции)
- сохраняет эффективность массивов в языке С
- может использоваться там же, где и С-ый массив

std::array: основные методы

operator[]	обращение по индексу
at()	обращение по индексу (с проверкой)
front()	доступ к первому элементу
back()	доступ к последнему элементу
empty()	проверка, пуст ли массив
size()	получение размера
max_size()	максимальное количество элементов (=size)
begin()	итератор на первый элемент
end()	итератор на элемент, идущий за последним

```cpp
#include <iostream>
#include <array>

void f(){
    std::array<int, 5> arr = {1, 2, 3, 4};
    std::cout << arr.empty() << std::endl;
    std::cout << arr.size() << ' '<< arr.max_size() << std::endl;
    std::cout << arr[arr.size() - 1] - arr[0] << std::endl;
    std::cout << arr.back() - arr.front() << std::endl;

    size_t sum = 0;
    for(std::array<int, 5>::iterator it = arr.begin(); it != arr.end();
                                                        ++it){
        sum += *it;
    }
    std::cout << sum << std::endl;
}
```

Вопросы?