C++基础课程

第???章 —— C++ code style

在编写C++代码时，程序员需要关注代码的可读性、一致性和可维护性。本章提供一个基本的C++代码风格指南，帮助你编写高质量的C++代码。同时，需要注意的是C++的编程风格因人而异，各大公司也都有自己的规定，你可以自行学习，但本文会简要介绍作者的编码风格。

可参考目录

1. 命名约定

- 使用有意义的变量和函数名，避免使用单个字符或缩写，除非它们在上下文中有明确的含义。
    - 注意, 一些特定的广为人知的缩写是允许的, 例如用 `i` 表示迭代变量和用 `T` 表示模板参数.
- 使用小驼峰命名法（lowerCamelCase）或下划线命名法（underscore_case）来命名标识符。
- 类名应该以大写字母开头，使用大驼峰命名法（UpperCamelCase）。
- 不管是静态的还是非静态的, 类数据成员都可以和普通变量一样, 但要接下划线或者开头加上 `m_`:
- 常量应该全部大写，单词间用下划线分隔，例如 `const int MAX_SIZE = 100;`。

  或者以 `k` 开头，采用驼峰命名法: `const int kDaysInWeek = 7;`.
- 枚举的命名应当和 常量 或 宏 一致: `kEnumName` 或是 `ENUM_NAME`.由于枚举值和宏之间的命名可能冲突, 由此, 优先选择常量风格的命名方式

```
 // webcolors.h (third party header)
#define RED   0xFF0000
#define GREEN 0x00FF00
#define BLUE  0x0000FF

// productinfo.h
// The following define product subtypes based on color

enum class Product_info { RED, PURPLE, BLUE };   // syntax error
```

- 通常 *不应该* 使用宏. 如果不得不用, 其命名像枚举命名一样全部大写, 使用下划线 `#define PI_ROUNDED 3.0`

2. 头文件保护

- 使用预处理器保护宏避免头文件的多重包含。

```
// demo.h
#ifndef DEMO_H_
#define DEMO_H_

...

#endif

```

- 头文件应该只包含必要的其他头文件，避免不必要的依赖。
- 也可以使用`#pragma once`来代替传统的宏保护。

3. 格式

- 使用4个空格来缩进代码块，正常情况下，空格和制表符(tab)不得混用，但现代编辑器或IDE都会将tab键替换为四个空格
- 在运算符周围使用空格，例如`int sum = a + b;`。
- 在逗号后面使用空格，例如`int x, y, z;`。
- 在控制结构（if、for、while等）后使用空格，例如`if (condition)`
- 每一行代码字符数最好不超过 80，包含长路径的`#include`语句可以超出80列，头文件保护也可以无视该原则.
- 尽量不使用非 ASCII 字符, 使用时必须使用 UTF-8 编码.
- 花括号你可以自由选择换行或者是不换行，但在同一个项目中不得混用。在现代工程中，换行风格可能更被推崇.

4. 注释

- 使用注释来解释代码的目的，而不是描述代码做了什么。
- 对于重要的函数和类，添加文档注释，描述函数的参数、返回值和作用。
- 避免不必要的注释，代码本身应该尽可能自解释。 // 真正优雅的代码不需要解释!

5. 函数

- 函数应该短小且专一，每个函数应该只做一件事情。
- 使用参数传递而不是全局变量。
- 避免使用递归，除非有必要
- 内联函数不得过长

6. 类

- 类的数据成员应该是私有的，通过公有成员函数访问。
- 提供恰当的构造函数和析构函数。
- 遵循单一职责原则，每个类应该有明确的功能。
- 使用访问修饰符来限制数据成员的访问。