C++17起可以使用inline修饰变量,使其为内联变量

在C++17前一个类的静态变量需要在内类声明,内外定义

```
struct Mystruct {
    static std::string msg;
};

std::string Mystruct::msg = "OK";
```

在C++17后可以使用如下方式

```
struct Mystruct {
    inline static std::string msg{"OK"};//OK (自c++17起)
};
```

```
struct Mystruct1 {
//    static int a = 1; //C++不允许在类初始化非常量静态成员
//    const static int a = 10;OK
};
```

根据一次定义原则 (ODR)，一个变量或实体的定义只能出现在一个编译单元内——除非该变量或实体被定义为inline的。

```
struct Mystruct3 {
    static int a;
};

int Mystruct3::a = 10;//如果被多个cpp文件包含会导致链接ERROR,违背ODR规则
```

预处理保护

```
#ifndef MY_HEAD
#define MY_HEAD

class Mystruct4 {
    static int a;
};

int Mystruct4 = 10;//如果被多个CPP包含,预处理器也没用,因为预处理做的是这部分内容不会重复出现
//但是如果在多个CPP文件出现的话在链接时会发生ERROR
#endif
```

C++17扩展

```
struct Mystruct5 {
    constexpr static int a = 0;//C++11/C++14:声明
    //C++17起:定义
    //等价:
    //constexpr inline static int a = 0;
};
    //对静态成员使用inline修饰符后,即使被多个cpp文件包含,也只会有一个全局对象
```