OOP上机实验(6)

(Version: 0.76 Date: 2025/05/09 Author: wangxp@fudan.edu.cn)

1 实验目标

- 编写完整的类，熟悉类的构造函数、析构函数、拷贝构造函数；熟悉操作符重载（含输入输出操作符）；熟悉成员函数与非成员函数的区别；熟悉友员函数；熟悉类的静态属性；
- 理解类中的 copy 操作，以及如何定义 copy 行为，包括复制构造函数和复制赋值操作；
- 理解类中的 move 操作，以及如何定义 move 行为，包括移动构造函数和移动赋值操作；
- 理解类对象的创建和释放过程，理解并熟悉操作符 new/delete, new[]/delete[]的重载；
- 理解RVO（return value optimization, 返回值优化）；
- 理解基于测试的程序开发过程。

2 实验内容

2.1 编写vec类

本次实验所给的文件包括：

- vec.h
- vec_test.cpp
- CmakeLists.txt

只需修改文件vec.h，最后也只需上传文件vec.h。 如果有需要说明的内容，请添加一个文本文件，例如vec.txt，然后上传。

vec.h中定义了三个模板类，分别是vec1、vec2和vec3，其中

- vec1类中没有定义任何成员方法和属性。但是在test_vec1_ctor()，我们可以很清楚地看出，对于这样的类，编译器隐含地生成了很多特殊的成员函数；
- 通常把vec2类定义为资源类，因其包含一个T*的动态数据。在vec2类中，定义了构造函数和析构函数；并且在析构函数中，对删除的资源进行了 nullptr 保护。但是很不幸，由于 C++编译器的默认行为生成了很多函数（看 TEST(vec2,*)），包括复制/移动行为的函数。在执行赋值操作(=)后，出现了两个指针指向同一个对象，并且一个指针丢失的现象(见代码TEST(vec2,copy))，导致了程序崩溃(segment fault);
- vec3类通过增加控制copy行为的函数(复制构造函数和复制赋值操作符)，精确地控制复制行为，保证资源的安全复制；同时增加控制move行为的函数(移动构造函数和移动赋值操作符)，保证资源的安全移动。

需要修改的代码集中在vec3中；在最终的测试中，需要把函数中的TEST(vec2, copy_construct)和TEST(vec2, copy_assign)这两个测试函数注释掉：因为vec2的实现会导致这两个函数直接崩溃。

在vec_test.cpp中，最后一个测试**return_value_optimization**，展示了RVO的情形，可以通过调试的方法查看具体的过程。

在源代码vec.h 中，我们增加了若干个 TODO，请大家根据程序的功能增加相应的代码。

2.2 配置googletest

实验中给出了测试代码vec_test.cc，其中使用了goolge的单元测试框架googletest[6]。
按照上面的设置，lab6自带的CMakeLists.txt可以正常工作；在运行之后，如果补充的代码正确，应该输出PASSED。

3 参考资料

[1] 默认构造函数。 https://zh.cppreference.com/w/cpp/language/default_constructor
[2] 复制构造函数。 https://zh.cppreference.com/w/cpp/language/copy_constructor
[3] 移动构造函数。 https://zh.cppreference.com/w/cpp/language/move_constructor
[4] 复制赋值运算符。 https://zh.cppreference.com/w/cpp/language/copy_assignment
[5] 移动赋值运算符。 https://zh.cppreference.com/w/cpp/language/move_assignment
[6] google test。https://google.github.io/googletest/primer.html
[7] RVO。https://zh.cppreference.com/w/cpp/language/copy_elision