沈阳航空航天大学

软件工程第三次作业

院（系）： 人工智能学院

专　　业： 物联网工程

班　　级： 物联网2101

学　　号： 213428010107

姓　　名： 孙琪轩

带队教师： 张翼飞

2024年04月28日

作业信息	沈阳航空航天大学计算机学院 2024 软件工程作业
课程目标	熟悉一个“高质量”软件的开发过程
作业目标	熟悉代码规范及结对互审

结对伙伴信息			
专业	物联网工程	班级	物联网 2101
学号	213428010113	姓名	刘贤
伙伴 Github 地址：https://github.com/liuxian-top9/-3/blob/main/%E8%BD%AF%E4%BB%B6%E5%B7%A5%E7%A8%8B%E4%BD%9C%E4%B8%9A3			
本人 Github 地址：https://github.com/LuckyE993/Software-Engineering/tree/master/code/Pair_programming1			

1. 参照 C/C++代码审查表（地址：https://blog.csdn.net/wenrenhua08/article/details/39591273）和 Java 代码审查表（地址：https://blog.csdn.net/u012107806/article/details/50765580），根据资料完成检查结果。

序号	重要性	审查项	结论
1		版权和版本声明是否完整?	否
2	重要	头文件是否使用了 ifndef/define/endif 预处理块?	无
3		头文件中是否只存放“声明”而不存放“定义”	无
4			
5	程序的版式		
6	重要性	审查项	
7		空行是否得体?	是
8		代码行内的空格是否得体?	是
9		长行拆分是否得体?	否
10		“{”和“}”是否各占一行并且对齐于同一列?	是
11	重要	一行代码是否只做一件事? 如只定义一个变量，只写一条语句。	否
12	重要	If、for、while、do 等语句自占一行，不论执行语句多少都要加“{}”。	是
13	重要	在定义变量（或参数）时，是否将修饰符 * 和 & 紧靠变量名? 注释是否清晰并且必要?	是
14	重要	注释是否有错误或者可能导致误解?	无
15	重要	类结构的 public，protected，private 顺序是否在所有的程序中保持一致?	无
16			
17	命名规则		
18	重要性	审查项	结论
19	重要	命名规则是否与所采用的操作系统或开发工具的风格保持一致?	是
20		标识符是否直观且可以拼读?	是
21		标识符的长度应当符合“min-length && max-information”原则?	否
22	重要	程序中是否出现相同的局部变量和全部变量?	否
23		类名、函数名、变量和参数、常量的书写格式是否遵循一定的规则?	是

24		静态变量、全局变量、类的成员变量是否加前缀?	否
25			
26	表达式与基本语句		
27	重要性	审查项	结论
28	重要	如果代码行中的运算符比较多，是否已经用括号清楚地确定表达式的操作顺序?	是
29		是否编写太复杂或者多用途的复合表达式?	无
30	重要	是否将复合表达式与“真正的数学表达式”混淆?	否
31	重要	是否用隐含错误的方式写 if 语句? 例如	
32		（1）将布尔变量直接与 TRUE、FALSE 或者 1、0 进行比较。	无
33		（2）将浮点变量用“==”或“！=”与任何数字比较。	无
34		（3）将指针变量用“==”或“！=”与 NULL 比较。	无
35		如果循环体内存在逻辑判断，并且循环次数很大，是否已经将逻辑判断移到循环体的外面?	是
36	重要	Case 语句的结尾是否忘了加 break?	无
37	重要	是否忘记写 switch 的 default 分支?	无
38	重要	使用 goto 语句时是否留下隐患? 例如跳过了某些对象的构造、变量的初始化、重要的计算等。	无
39			
40	常量		
41	重要性	审查项	结论
42		是否使用含义直观的常量来表示那些将在程序中多次出现的数字或字符串?	无
43		在 C++ 程序中，是否用 const 常量取代宏常量?	无
44	重要	如果某一常量与其它常量密切相关，是否在定义中包含了这种关系?	无
45		是否误解了类中的 const 数据成员? 因为 const 数据成员只在某个对象生存期内是常量，而对于整个类而言却是可变的。	无
46			
47			
48	函数设计		
49	重要性	审查项	结论

50		参数的书写是否完整？不要贪图省事只写参数的类型而省略参数名字。	是
51		参数命名、顺序是否合理?	是
52		参数的个数是否太多?	否
53		是否使用类型和数目不确定的参数?	否
54		是否省略了函数返回值的类型?	否
55		函数名字与返回值类型在语义上是否冲突?	否
56	重要	是否将正常值和错误标志混在一起返回？正常值应当用输出参数获得，而错误标志用 return 语句返回。	否
57	重要	在函数体的“入口处”，是否用 assert 对参数的有效性进行检查？	否
58	重要	使用滥用了 assert？ 例如混淆非法情况与错误情况，后者是必然存在的并且是一定要作出处理的。	否
59	重要	return 语句是否返回指向“栈内存”的“指针”或者“引用”?	否
60		是否使用 const 提高函数的健壮性？const 可以强制保护函数的参数、返回值，甚至函数的定义体。“Use const whenever you need”	否
61			
62	内存管理		
63	重要性	审查项	结论
64	重要	用 malloc 或 new 申请内存之后，是否立即检查指针值是否为 NULL？（防止使用指针值为 NULL 的内存）	否
65	重要	是否忘记为数组和动态内存赋初值？（防止将未被初始化的内存作为右值使用）	否
66	重要	数组或指针的下标是否越界?	无
67	重要	动态内存的申请与释放是否配对？（防止内存泄漏）	无
68	重要	是否有效地处理了“内存耗尽”问题?	否
69	重要	是否修改“指向常量的指针”的内容?	无
70	重要	是否出现野指针？例如（1）指针变量没有被初始化;（2）用 free 或 delete 释放了内存之后，忘记将指针设置为 NULL。	是
71	重要	是否将 malloc/free 和 new/delete 混淆使用?	否
72	重要	malloc 语句是否正确无误？例如字节数是否正确？类型转换是否正 确?	否

73	重要	在创建与释放动态对象数组时，new/delete 的语句是否正确无误?	否
74			
75	C++ 函数的高级特性		
76	重要性	审查项	结论
77		重载函数是否有二义性?	无
78	重要	是否混淆了成员函数的重载、覆盖与隐藏?	无
79		运算符的重载是否符合制定的编程规范?	无
80		是否滥用内联函数? 例如函数体内的代码比较长，函数体内出现循环。	无
81	重要	是否用内联函数取代了宏代码?	无
82			
83	类的构造函数、析构函数和赋值函数		
84	重要性	审查项	结论
85	重要	是否违背编程规范而让 C++ 编译器自动为类产生四个缺省的函数:	
86		(1) 缺省的无参数构造函数;	否
87		(2) 缺省的拷贝构造函数;	否
88		(3) 缺省的析构函数;	否
89		(4) 缺省的赋值函数。	否
90	重要	构造函数中是否遗漏了某些初始化工作?	否
91	重要	是否正确地使用构造函数的初始化表?	无
92	重要	析构函数中是否遗漏了某些清除工作?	否
93		是否错写、错用了拷贝构造函数和赋值函数?	否
94	重要	赋值函数一般分四个步骤:	
95		(1) 检查自赋值;	否
96		(2) 释放原有内存资源;	否
97		(3) 分配新的内存资源，并复制内容;	否
98		(4) 返回 *this。是否遗漏了重要步骤?	否
99	重要	是否正确地编写了派生类的构造函数、析构函数、赋值函数?	
100		注意事项:	
101		(1) 派生类不可能继承基类的构造函数、析构函数、赋值函数。	无

102		（2）派生类的构造函数应在其初始化表里调用基类的构造函数。	无
103		(3)基类与派生类的析构函数应该为虚(即加 virtual 关键字)。	无
104		（4）在编写派生类的赋值函数时，注意不要忘记对基类的数据成员重新赋值	无
105			
106	类的高级特性		
107	重要性	审查项	结论
108	重要	是否违背了继承和组合的规则?	
109		(1) 若在逻辑上 B 是 A 的“一种”，并且 A 的所有功能和属性对 B 而言都有意义，则允许 B 继承 A 的功能和属性。	无
110		(2) 若在逻辑上 A 是 B 的“一部分”（a part of），则不允许 B 从 A 派生，而是要用 A 和其它东西组合出 B。	无
111			
112	其它常见问题		
113	重要性	审查项	结论
114	重要	数据类型问题:	
115		（1）变量的数据类型有错误吗?	否
116		（2）存在不同数据类型的赋值吗?	否
117		（3）存在不同数据类型的比较吗?	否
118	重要	变量值问题:	
119		（1）变量的初始化或缺省值有错误吗?	否
120		（2）变量发生上溢或下溢吗?	否
121		（3）变量的精度够吗?	是
122	重要	逻辑判断问题:	
123		（1）由于精度原因导致比较无效吗?	否
124		（2）表达式中的优先级有误吗?	否
125		（3）逻辑判断结果颠倒吗?	否
126	重要	循环问题:	
127		（1）循环终止条件不正确吗?	否
128		（2）无法正常终止（死循环）吗?	否
129		（3）错误地修改循环变量吗?	否
130		（4）存在误差累积吗?	否
131	重要	错误处理问题:	
132		（1）忘记进行错误处理吗?	无
133		（2）错误处理程序块一直没有机会被运行?	无

134		（3）错误处理程序块本身就有毛病吗？如报告的错误与实际错误不一致，处理方式不正确等等。	无
135		（4）错误处理程序块是“马后炮”吗？如在被它被调用之前软件已经出错。	无
136	重要	文件 I/O 问题:	
137		（1）对不存在的或者错误的文件进行操作吗？	无
138		（2）文件以不正确的方式打开吗？	无
139		（3）文件结束判断不正确吗？	无
140		（4）没有正确地关闭文件吗?	无

2. 对同伴的代码写一篇 500 字以上的评论，介绍同伴的优缺点。

同伴的代码是一个多项式处理程序，用于实现多项式的输入、输出、相加、相减以及存储功能。整体上，代码展示了结构化设计的典型特征和一定的面向对象思想。在具体实现中，代码涵盖了多项式的动态数据结构设计、排序算法的应用、文件输入输出操作，以及简单的算术操作。这段代码在实现基本功能方面做得比较完善，能够处理一些基础的多项式运算任务。从软件工程的角度看，代码在内存管理、错误处理和代码优化方面还有很大的改进空间。为了提高代码的健壮性和可维护性，需要重构代码来增加异常处理并优化内存管理。

优点

1. 模块化设计：代码清晰地分为了不同的功能块，如读取多项式、打印多项式、保存多项式到文件、多项式相加与相减等，使得代码易于理解和维护。
2. 函数利用：通过使用标准库函数 sort()来排序多项式的项，对 C++标准库可以有效运用。
3. 用户交互：程序提供了良好的用户交互界面，通过命令行提示用户输入，增强了程序的用户友好性。
4. 动态内存管理：通过动态数组来管理多项式的项，展示了对 C++动态内存管理的应用。

缺点

1. 内存泄漏与管理问题：程序使用了动态内存，但缺乏对异常情况例如输入非法数据时的内存泄漏的处理问题。程序未定义析构函数来释放分配的内存，可能导致严重的内存泄漏。
2. 异常处理不足：在文件操作和用户输入过程中，代码未对潜在的错误进行足够的检查和处理，如不检查文件是否成功打开。
3. 代码复用性低：多项式的相加与相减函数有大量重复代码，可以通过设计更通用的函数或使用多态性来提高代码的复用率。
4. 缺少构造函数和赋值操作的重载：程序没有为 Polynomial 结构提供构造函数、析构函数、拷贝构造函数和赋值运算符重载，涉及到动态内存管理时容易造成数据泄露。
5. 效率问题：在添加和减少多项式时，代码效率可以进一步优化。可以在输入时直接排序和合并同类项，而不是在每次运算后处理。