

# 全过程非标准化考核改革

## 校级优秀案例申报材料

课程名称及代码：	[52004CC0LA]程序设计基础（C++）		
所属学院/专业团队：	数字艺术与设计学院 数字媒体技术系		
课程负责人：	唐媛		
主要参与教师：	王娇琳、张晨敏		
面向年级/专业：	2025 级/数字媒体技术专业、虚拟现实技术专业		
课程学分/学时：	3/48		
改革亮点 (推荐理由)	把握数字媒体与虚拟现实专业特色，以文化数字化传播为真实场景，构建三维评价、AI 赋能、行业协同的非标准化考核体系。强化高阶能力考查，丰富个性化创新空间，规范工具使用边界，实现知识能力素养协同提升，成效扎实具推广价值。		
课程自评得分	95 分	院级评审得分	分

日期： 2025 年 12 月 25 日

# 一、改革背景与问题导向

## （一）课程原有考核痛点

《程序设计基础（C++）》是数字媒体技术、虚拟现实技术专业的核心基础课，承担着夯实编程根基、培养工程思维、塑造工程操守的关键任务。课程遵循知识、能力、素养的三维培养目标（图1），改革前采用形成性考核与终结性考核各占五成的评价结构，形成性考核包含20分期中考试与30分项目任务，终结性考核以期末考试为主。尽管考核结构兼顾过程塑造与结果验收，且题目设计逐年丰富，但仍存在适配性短板。

**一是**期中与期末考试中记忆类内容占比较高，对复杂逻辑调试、模块化编程等高阶能力的考查权重不足。**二是**项目类考核场景相对单一，文化元素融入与真实行业场景的结合深度不够，难以有效支撑学生运用知识解决专业相关实际问题的能力培养。**三是**评价维度多集中于代码运行结果，对编程规范、工具合规使用及素养指标缺乏明确评判标准，未能充分呼应课程三维素养目标。



图1 程序设计基础C++课程目标及重难点

## （二）改革目标

立足课程定位与原有考核短板，确立改革核心方向为，**一是**优化考核结构与内容设置，强化逻辑分析、场景迁移、创新实践等高阶能力的考查力度。**二是**丰富考核场景设计，结合专业特色融入文化元素与真实应用需求，提升任务的实践适配性与趣味性，引导学生树立文化自信。**三是**完善多元评价体系，补充素养类指标的具体评判标准，实现知识能力素养的全面覆盖与科学评价，推动学生编程能力与职业素养的协同提升。



图 2 考核改革目标

## 二、考核改革设计实施

### （一）创新命题设计（附录 1：非标准化命题改革）

立足数媒专业特色与课程目标，构建基础夯实、能力进阶、文化赋能的阶梯式命题体系。**基础层**保留核心语法考查，确保知识落地；**能力层**围绕数字媒体、虚拟现实专业实际需求，设计模块化编程、复杂逻辑调试等任务，强化知识迁移应用；**拓展层**融入传统文化、非遗元素等作为项目主题，以主题闯关游戏开发为典型案例，将循环控制、随机事件触发等知识点与文化场景深度结合，既提升任务趣味性，又引导学生树立文化自信。命题过程联动校内专业教师与行业技术人员，兼顾教学要求与行业应用场景适配性，遵循企业级流程考核，使考核任务更具实践价值。



图 3 命题设计创新模型

## （二）优化过程评价（附录 2：形成性考核样例）

依托东软智慧教学平台，建立全周期递进式过程评价机制。按需求分析、框架搭建、功能开发、测试优化四个阶段拆解项目考核节点，每个阶段聚焦核心能力要点开展针对性评价。**需求分析**阶段侧重方案合理性审核，**框架搭建**阶段关注逻辑完整性，**功能开发**阶段跟踪代码规范度，**测试优化**阶段评估问题解决效率。形成性考核占比保持五成，细化为 40 分项目任务与 10 分习惯评测，习惯评测涵盖前后测习题、主动性互动、新工具使用等维度，通过课堂点评、线上批注等个性化反馈方式，全程跟踪学生能力成长，及时校准学习方向。



图 4 项目式过程评价流程

## （三）完善评价标准（附录 3：课程考核量规）

对标课程知识、能力、素养三维目标，构建细化的评价量表。**知识维度**聚焦语法应用准确性，**能力维度**侧重逻辑建构、场景迁移与创新实践，**素养维度**明确编程规范、主动探究及辅助工具合理使用的具体要求。量表采用分级描述方式，清晰界定不同水平的评判依据，弥补此前评价标准模糊的短板。考核前公开评价标准与示例，让学生明确努力方向，确保评价过程客观可操作，实现以评促学、以评促改。

## （四）技术赋能支撑（附录 4：AI 赋能 C++编程分层提示词）

规范 AI 工具应用场景，构建辅助不主导的 AI 赋能体系。**命题阶段**借助 AI 生成文化元素转化思路与行业场景参考，拓宽命题维度；学生**学习阶段**允许使用 AI 查询语法规则、排查代码 BUG，同时明确禁止核心逻辑生成，引导自主思考；**评阅阶段**利用 AI 辅助筛查基础语法错误，将教师精力解放出来，聚焦高阶能力与素养维度的深度评价，既提升评阅效率，又保证评价精准度。



三、考核改革成效亮点

（一）量化与质性成效

改革后学生综合能力与学习质量显著提升。**量化层面**，记忆类题目占比由约 50%降至 30%以下，高阶能力考查占比提升，学生代码调试通过率、模块化编程完成率较改革前均有明显提高。课堂互动参与度、课后主动咨询优化方案的学生比例大幅增加，形成良性学习循环。**质性层面**，学生工程思维得到有效培养，能够自主拆解复杂编程任务，从需求分析到测试优化的全流程设计能力增强。文化融入类任务让学生在技术实践中深化了对传统元素的理解，文化自信与数字化传播意识逐步形成。AI 工具使用基本达到规范有序，学生普遍能够合理借助工具解决语法查询、BUG 排查等问题，自主思考与独立解决核心逻辑问题的能力得到保留与强化。

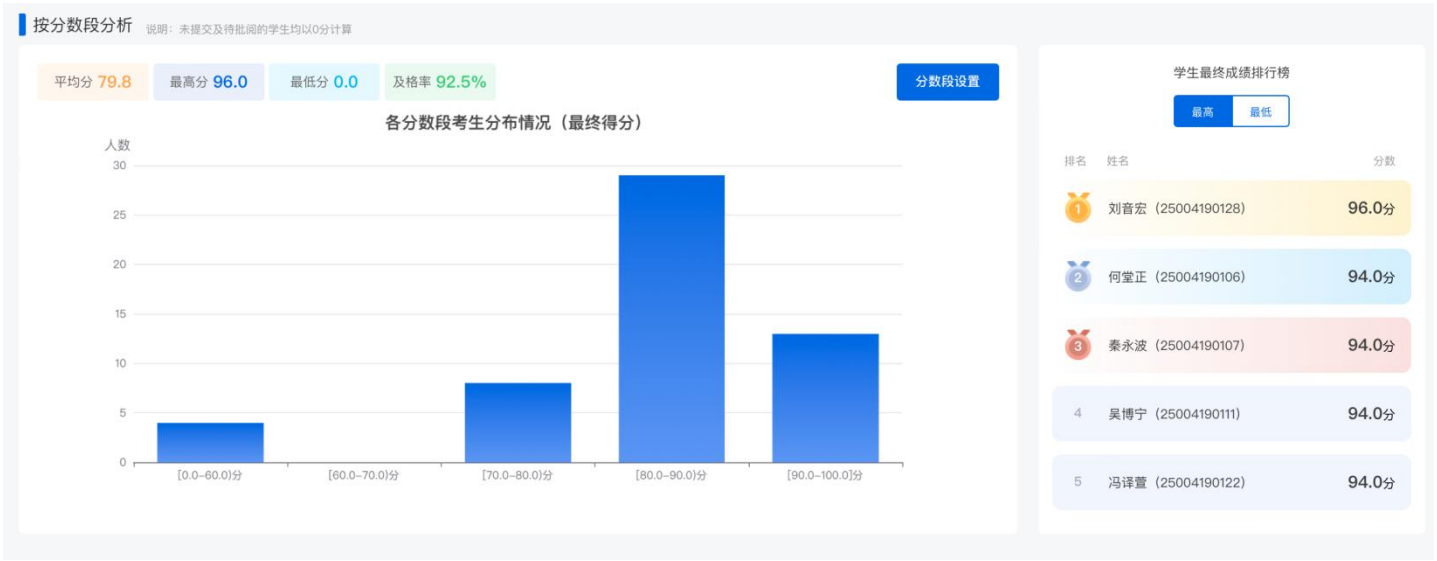


图 5 智慧平台线上测试

表 1 学生项目考核心得体会

学生项目心得	实验中遇到的最大难点是游戏平衡性的把握。初期设计时，年兽的强度设置过高，导致玩家很难通关，这让我意识到游戏设计不仅是技术实现，更需要考虑用户体验。通过反复测试和调整参数，最终找到了合适的难度。另一个挑战是如何将抽象的文化元素转化为具体的游戏机制，比如春联如何体现文采，倒福如何传递福气，这个过程需要创意和对文化内涵的深入理解。
	本次实验让我深刻体会到数字媒体技术在文化传播中的重要作用。将抽象的春节习俗转化为具体的游戏交互，不仅让传统文化活了起来，更让学习者在娱乐中自然吸收文化知识。特别是知识问答后的文化解析环节，既巩固了正确答案，又传播了深厚的文化内涵。通过需求分析→框架搭建→功能开发→测试优化的完整开发流程，我建立了系统的工程化思维。在 8 课时内高效完成任务，锻炼了时间管理和任务拆解能力。BUG 测试修复表的运用，让我认识到系统化测试在软件开发中的关键地位。

	这次的实验经历让我们不仅巩固了 while 循环、rand()随机数等 C++技能，更学会用模块化拆分代码、按时间拆解任务的工程思维；同时把春联、年兽等民俗转化为游戏功能，让我懂了数字媒体是“技术载文化”的载体。过程里虽因变量命名、体验设计踩了坑，但也明白编程既要“实现功能”更要“清晰好用”——这次实践让我摸到了技术+文化+用户体验融合的专业内核。
	理解了循环不仅是重复执行，更是通过条件控制流程推进。初期随机事件逻辑混乱，概率分配与实际交互不匹配。经过反思后，我细化了事件类型，让文化元素与游戏机制结合更紧密，优化界面输出格式提升体验。用编程把年兽、饺子等传统文化元素转化为互动游戏，让文化传播更生动，也体会到技术是传承的工具，整体体会了 C++的模块化能提高开发效率，代码不仅要能运行，还要有逻辑、有温度！
	<p>在完成本次小游戏的实验过程中，我不仅巩固了 C++ 的基础编程知识，还在逻辑思维上有了新的感悟。让我对 C++ 的核心语法有了更透彻的理解。在之前学习 if-else、switch 分支结构和 while 循环时，我只对它们有初步的了解，而在设计小游戏的过程中，我需要根据用户、程序的实际需求将这些结构语法结合起来。</p> <p>在编写代码的过程中，我思考如何使用程序中充满活力的交互设计让玩家在游戏中感受春节的文化氛围。这让我明白，编程不仅仅是与代码和计算机息息相关，还可以成为传承和传播传统文化的载体，通过我们专业的技术手段让传统文化以更有趣、更易接受的形式呈现出来。总的来说，这次春节集福纳祥游戏的编程实验是一次非常有意义的实践！在今后的学习中，我会继续保持这种“学以致用”的态度，尝试将编程与更多实际场景和文化元素结合，不断提升自己的编程能力和创新思维，写出更有价值的程序。</p>

表 2 学生 AI 应用分析

AI 使用场景描述 (学生作业报告提炼)	对 AI 的 依赖程度	核心收获与反馈	对应考核模块
用 AI 查询了 while 循环与 rand 函数的使用格式，排查了关卡推进逻辑中的语法错误；AI 建议用函数封装战斗模块，我们理解后自主设计了传参方式	辅助型 (低依赖)	AI 帮我们快速解决语法问题，节省了调试时间，核心的民俗元素融合与福运评级系统都是自己构思的，学会了合理借助工具	形成性考核 - 项目开发
通过 AI 校验了民俗知识支线任务的分支逻辑正确性，查询了字符串输出的语法规则；尝试让 AI 生成过文化解说模块代码，发现不符合需求，最终自主编写	辅助型 (低依赖)	AI 适合解决基础语法问题，但核心功能还是要自己设计，不然会偏离项目主题，还学会了辨别 AI 输出的合理性	形成性考核 - 项目开发

用 AI 查询了数组与循环结合处理统计数据的语法，排查了数据累加时的逻辑漏洞；AI 提示了代码优化方向，我自主简化了重复计算的代码段	辅助型 (低依赖)	AI 让我不用在语法上花费太多时间，能更专注于数据统计场景的适配，也明白了 AI 的优化建议需要结合实际需求调整	形成性考核 - 场景迁移实践
AI 帮我确认了值传递与引用传递的区别，排查了函数调用时的参数不匹配错误；尝试让 AI 生成函数框架，之后自己填充核心逻辑并调试	辅助型 (中依赖)	刚开始想让 AI 直接生成函数，但发现不符合题目要求的模块化需求，后来改成用 AI 解决细节问题，自主设计整体结构	形成性考核 - 基础能力训练
用 AI 查询了嵌套循环的执行流程，校验了选择逻辑的语法正确性；遇到复杂逻辑问题时，AI 给了思路提示，最终自己梳理清楚并编码	辅助型 (低依赖)	AI 的思路提示很有用，但不能直接照搬，需要自己理解后转化为代码，这种方式既提高了效率，又没影响自主思考	形成性考核 - 阶段性综合应用
AI 提供了民俗元素与编程任务结合的思路，查询了游戏界面相关的语法规则；排查了多模块联动时的 BUG，核心逻辑都是团队讨论后自主编写的	辅助型 (低依赖)	AI 在创意思路和语法排查上帮了大忙，核心功能的设计与实现让我们加深了对编程的理解，也学会了规范使用 AI 工具	形成性考核 - 综合项目开发

（二）改革前后对比

命题设计上，改革前以单一语法记忆与简单应用为主，如直接考查循环语句格式、变量定义规则等，改革后聚焦真实场景下的综合应用，如以春节主题游戏开发为核心任务，需整合循环控制、分支判断、随机数生成等多个知识点，同时融入民俗文化元素，题目综合性与实践价值大幅提升。

评价标准上，改革前仅关注代码是否可运行，评价维度单一模糊；改革后建立三维分级评价量表，明确知识维度的语法准确性、能力维度的逻辑完整性与创新度、素养维度的编程规范与工具使用合规性，每个维度均有清晰可量化的评判依据，评价科学性与客观性增强。

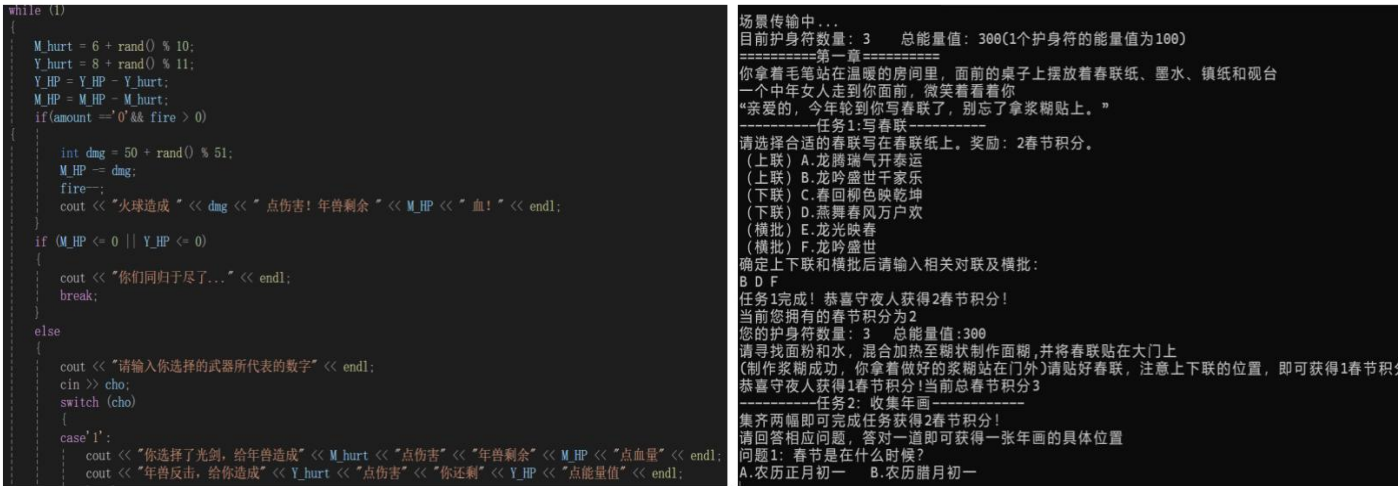
学生成果上，改革前多为同质化的基础代码实现，功能单一；改革后成果呈现个性化与创新性，出现多结局设计、民俗知识问答嵌入、失败重试机制等特色功能，部分作品还具备简单的文化传播落地潜力，成果质量与多样性实现双重突破。

### （三）学生优秀作答/成果

以形成性考核 2 春节守岁祈福闯关游戏设计与开发为例。

董庆玮、陈文道团队开发的《春节奇遇记》。**基础能力**层面，代码逻辑严谨规范，顺利整合 while 循环关卡推进、if-else 分支事件触发、rand 函数随机事件生成等核心知识点，语法应用准确无误，框架搭建合理清晰。**高阶思维**层面，深度融合春节民俗元素，将包饺子放鞭炮等民俗转化为能量增益与福运积累机制，斗夕兽设计呼应驱邪纳福文化内涵，实现技术与文化的有机衔接。**实践创新**层面，自主添加福运值评级系统，根据玩家游戏过程中的选择与表现给出不同评级与文化解说，增加了游戏的趣味性与文化深度。**价值延伸**层面，作品具备完整的交互逻辑与场景闭环，在大二学习游戏引擎后，稍加优化即可作为传统文化数字化传播的轻量化载体，具有一定实践应用价值。

**学生体会：**最初只是想写个春节问答小程序，*随着对传统习俗的了解越深，越想把写春联、祭灶、猜灯谜、包饺子打年兽这些片段串成一条完整的故事线*。于是有了“守夜人”这个设定，玩家不是被动答题，而是在除夕夜里一步步完成仪式，最终亲手击退年兽，让“年味”有了游戏化的落地。



The image shows a side-by-side comparison of C++ code and its runtime output. The code on the left is a snippet from a game, featuring a while loop for a boss fight with a year beast. It uses random number generation for damage and health, and a switch statement for player actions. The output on the right shows the game's narrative progression, including a scene description, a quest to write a couplet, a list of choices, and the resulting game state like energy levels and scores.

图 6 部分程序及结果

以形成性考核 3 承古算之智练函数之技数学系统开发为例。

高睿涵单人开发的《承古算之智练函数之技》展现了扎实的编程功底与深厚的文化融合能力。**基础层面**以自定义函数封装七大核心功能，运用 switch 分支、while 循环等语法，妥善处理边界情况，代码规范、框架清晰。**高阶思维**上，深度挖掘《九章算术》中的正负术、圆田术等古算精髓，将古法算理转化为现代编程逻辑，实现以码复算的跨域整合。**实践创新**方面，自主设计古算学馆主题交互框架，搭配古法话术提示，优化古算逻辑的现代呈现形式，完整记录了从需求分析到迭代优化的全过程。**价值延伸**上，作品既是函数知识的实践成果，更是古算文化传播的有效载体，其文化技术的融合思路，激发了大一同届学生跨学科实践创新，具备较好的示范价值与意义。

**学生体会：**本次承古算之智练函数之技实验，让我在古代数学文化与现代 C++ 编程的碰撞中，*收获了远超单纯代码编写的认知提升*。这不仅是一次编程技能的实操训练，更是一场跨越千年的算学思想对话。在函数设计过程中，最具挑战也最有趣的是“以今释古”的转化过程。起初我以为只需将现代算法



套上古法名称即可，但实际操作中发现，真正的核心是还原古算逻辑的本质。比如约倍术的实现，我没有直接使用常用的欧几里得算法，而是遵循《九章算术》的“更相减损术”——“可半者半之，不可半者，副置分母、子之数，以少减多，更相减损，求其等也”。通过循环实现两数“以少减多”的逻辑，不仅让代码贴合主题，更让我理解了古代数学家化繁为简的思维智慧。而勾股术的“勾自乘、股自乘，再并之，开方除之”，则让我意识到，古代算学早已具备清晰的逻辑拆解思想，这与现代编程中函数的模块化设计不谋而合。

实验也让我对古代数学文化有了全新的认知。以前总觉得《九章算术》《孙子算经》是晦涩难懂的古籍，但通过将其中的算术转化为可运行的程序，我发现这些典籍中蕴含的不仅是具体的计算公式，更是一套完整的逻辑体系。正负术作为古代商贸、工程计算的基础，其对正负数运算的规范，比西方早了数百年；圆田术以“径一周三”改良值  $\pi \approx 3.14$  计算，体现了古人尊重实际、不断优化的务实精神。**这些发现让我深刻体会到，古代数学不是孤立的公式堆砌，而是与生活实践紧密结合、充满智慧的知识体系，是中华优秀传统文化的重要瑰宝。**

在编程能力层面，本次实验让我对函数封装有了更扎实的掌握。七大功能全部通过自定义函数实现，每个函数的命名、参数设计、返回值定义都经过反复斟酌，既保证了代码的可读性，又贴合古算主题。同时，边界情况的处理让我意识到，编程不仅要实现核心功能，更要考虑实际使用中的各种场景，培养了严谨的逻辑思维。此外，使用 Visual Studio 2019 调试代码、用 WPS 整理实验文档的过程，也提升了我的工具应用能力和文档规范化意识。

**作为虚拟现实专业的学生，这次实验让我对“知识融合”有了更深的理解。古代算学的逻辑思维与现代编程的技术手段结合，不仅能实现功能，更能赋予程序文化内涵。未来在开发虚拟现实相关项目时，或许可以将这种文化+技术的思路融入其中，比如设计以古代算学为主题的互动教育类 VR 场景。**



图 6 部分结果

#### 四、总结反思与持续改进

**教师反思**方面，改革有效解决了原有考核中知识考查偏重、场景适配不足等问题，实现了知识能力素养的多元评价。但仍需优化数字媒体技术、虚拟现实技术不同专业的任务差异化设计，让考核更贴合数字媒体技术与虚拟现实技术的专业特色需求。

**学生感悟**集中体现为学习主动性的提升，有学生反馈真实文化场景让编程学习更具意义，通过自主设计功能深化了对知识的理解与应用，也学会了合理运用 AI 工具辅助学习。

**后续**将进一步联动行业专家细化任务场景，增加成果答辩环节强化成果真实性核验，同时优化评价量表的分级精度，推动考核改革与教学实践形成更紧密的闭环。

附录 1：非标准化命题改革

命题导向	知识单元	AI+改革后非标准化题目 / 项目	价值引领
传统文化融合	选 择 结 构 (switch/if)	1. 传统色彩查询系统：输入 1-5 对应朱红、黛青、明黄、月白、石绿，输出色彩文化寓意（如朱红→喜庆吉祥）与应用场景（春联、婚服）	深挖传统文化内涵，实现技术与文化传播结合，增强文化自信
		2. 二十四节气养生：输入节气序号（1-24），判断季节（春 / 夏 / 秋 / 冬），输出对应养生建议（如春季养肝护脾，多吃春笋）	
		3. 传统糕点适配：先判断季节（清爽季 / 温润季），再选择口味（清甜 / 咸香等），输出糕点配方与文化适配理由（如夏季绿豆糕清热解暑）	
传统文化融合	循 环 结 构 (for/while)	1. 春节贴福字：用 for 循环模拟贴 5 个福字，依次对应大门、客厅等位置，输出“福气满满过大年”祝福	传统文化场景化、互动化，让编程成为文化传承载体
		2. 扎染图案绘制：用双层 for 循环绘制 6x6 非遗扎染图案（行列和为偶数显示●，否则○），标注“非遗工艺数字化”	
		3. 古算学馆系统：循环提供正负术（加减乘除）、圆田术（圆周长面积）等 7 大古法功能，支持重复操作直至“归隐”	
传统文化融合	函 数 设计与传参	1. 勾股术函数：输入勾、股长度，用古法公式计算弦长，函数内输出“勾三自乘得九，股四自乘得十六”的古法演算过程	函数设计贴合古法逻辑，实现以码复算、以技传文
		2. 约倍术函数：用更相减损术（古法）设计函数，求两数最大公约数与最小公倍数，函数命名贴合古法逻辑	
		3. 节日祝福函数：输入春节、端午等节日名称，返回对应传统文化祝福语，输入其他节日返回通用文化祝福	
课程思政渗透	基础语法 (变量 / 表达式)	1. 国旗护卫队正步计算：定义总训练步数变量，编写函数返回正步走步数（总步数 1/5），输出“彰显责任与担当”思政话术	将责任担当、环保意识、文明素养融入基础编程，实现隐性德育
		2. 校园节水统计：定义每日用水量变量，计算日均节水量，输出“低碳环保，节约用水”小贴士	
		3. 文明行为判断：输入行为描述（如垃圾分类、礼让师长），用 bool 函数判断是否为文明行为，输出正向引导话术	
课程思政渗透	函 数 设计与传参	1. 节水小贴士函数：void 类型函数循环打印 3 条校园节水建议，强化低碳环保理念	思政目标与函数功能深度绑定，实现知识学习与素养塑造协同
		2. 捐款统计函数：输入捐款总金额与人数，计算人均捐款（保留 2 位小数），输出“爱心传递，温暖同行”正向引导	
		3. 素质积分评级：输入素质积分，用 char 函数返回 A-E 等级，关联“主动探究、严谨规范”等素养目标	
数字媒体游戏逻辑	循 环 结 构 (for/while)	1. RPG 勇者战斗：用 while 循环模拟勇者（100 血）与怪兽（50 血）对战，轮流攻击（随机伤害），打印伤害值与剩余血量，直至分出胜负	贴合数字媒体专业特色，强化游戏交互逻辑、随机事件、场景遍历等核心能力
		2. 宝石矿洞探索：用双层 for 循环遍历 10×10 矿洞，按规则标记宝石、金矿、陷阱，计算总得分并输出探索结果	

		3. 宠物精灵养成：用 while 循环模拟训练升级，每次获得 20-40 经验，100 经验升 1 级，直至 5 级，打印升级进度	
数字媒体 游戏逻辑	综合应用（多 知识点整合）	1. 春节守岁祈福游戏：整合变量、循环、分支、函数，设计 3 个民俗关卡（贴春联、猜灯谜、斗年兽），包含福运值系统、失败重试机制与文化解说	还原数字媒体游戏开发 全流程，培养任务拆解、 模块化设计、交互逻辑构 建能力
		2. 三国武将布阵：用双层 for 循环构建 7×7 阵型，按武力值标记精锐武将、普通武将，统计各类武将数量	
		3. 猜数字游戏：系统随机生成 1-100 数字，用 for 循环限制 5 次猜测机会，打印“第 X 次猜测机会”提示，结束输出游戏结果	
数字媒体 游戏逻辑	选 择 结 构 （if/switch）	1. 侠客状态系统：用 if 判断体力值（>50 正常 /<50 受伤），用 switch 区分受伤程度（轻伤→缓慢移动 / 重伤→无法行动），模拟游戏角色状态切换	聚焦数字媒体交互设计、 场景可视化需求，提升专 业适配能力
		2. 音乐节灯光矩阵：用双层 for 循环设计 6×6 灯光矩阵（行列乘积为偶数亮★，否则暗☆），适配舞台数字化场景	
		3. 咖啡角点单验证：用 while 循环实现饮品编号（1-5）输入验证，无效输入重复提示，直至输入正确并打印“正在制作 XX 饮品”	



## 大连东软信息学院

# 实验报告书

### 一、基本信息

- 实验名称: 春节守岁祈福闯关游戏设计与开发 —C++ 控制台版
- 实验学时: 8
- 实验类别:  
☐验证性 ☐基本应用性 ☒综合性 ☐设计性 ☐创新研究性
- 课程名称: 程序设计基础 C++
- 课程编号: 52004CC0T1
- 授课对象: ☒本科 ☐高职 ☐专升本 ☐中职升本
- 实验场所: ☐课内 ☐课外 ☒课内和课外

### 二、实验目的

- 巩固 C++ 基础编程知识点, 熟练运用变量定义、循环结构 (while)、分支结构 (if-else/switch)、基础类与对象等核心语法, 提升代码编写与调试能力。
- 理解数字媒体技术中数字化交互场景构建的核心逻辑, 掌握控制台游戏的基础开发流程, 培养需求分析→技术实现→优化迭代的工程思维。
- 挖掘春节辞旧迎新、驱邪纳福的传统文化内涵, 实现文化素养与专业技能的深度融合, 增强文化传承意识与正向价值认识。
- 提升问题解决能力, 在 8 课时内高效完成模块化开发, 培养任务拆解与时间管理能力。

### 三、实验任务及原理

#### 第一阶段: 需求分析

小组调研讨论梳理春节核心民俗元素, 筛选 3 个易转化为游戏功能的元素 (例如春联、年兽、贴倒福等), 记录每个元素的文化寓意与游戏转化思路, 形成《春节文化元素清单》。

共同定义游戏核心规则, 明确玩家初始状态、通关条件、失败条件, 用简单流程图描述行动→触发事件→状态更新→胜负判定的基础流程, 形成《游戏核心规则表》。

#### 第二阶段: 模块化技术开发

搭建基础框架：编写 C++ 控制台项目，导入所需头文件，定义能量、烟花、关卡等核心变量并初始化；开发开场界面（输出游戏名称、文化背景、初始状态）与结束判定模块（根据最终状态输出通关或鼓励提示），确保基础框架可独立运行。

实现核心功能：用 `while` 循环构建关卡推进逻辑与事件触发机制，通过 `rand()` 函数分配事件概率；用 `if/switch` 分支实现收集与对战功能，确保功能与规则一致。

### 第三阶段：测试优化与成果整合

功能测试与 BUG 修复：针对正常通关、能量耗尽失败、年兽对战胜负等 3 个核心场景进行测试，记录常见 BUG，通过逐行调试定位并修复，形成《BUG 测试修复表》。

体验与话术优化：调整控制台输出格式，补充文化寓意话术与正向引导话术，可根据体验调整事件概率，提升游戏流畅度与文化氛围。

成果整合：整理可完整运行的最终版代码，撰写实验报告，确保代码注释清晰、报告逻辑完整。

#### 1. 编程技术原理

循环结构：通过 `while` 循环实现关卡推进、事件触发、对战回合的核心逻辑，满足游戏重复交互的需求，核心是通过条件判断控制循环启停（如 `level <= 3 && energy > 0` 控制关卡持续）。

分支结构：用 `if-else` 判断事件概率、通关条件、战斗胜负，用 `switch` 处理多类型信物、考验、年兽，实现多场景分支交互，提升游戏灵活性。

随机数生成：通过 `srand((unsigned int)time(NULL))` 初始化随机种子，`rand()%100` 生成 0-99 随机数，按比例分配事件概率（如 40% 收集、30% 考验、30% 对战），模拟游戏随机性。

#### 2. 数字媒体技术原理

交互设计原理：遵循用户行动→系统反馈→状态更新的交互逻辑，通过控制台 IO 实现输入输出交互，保障用户体验流畅。

数字化文化传承：将春节民俗转化为数字游戏元素，通过技术手段实现文化符号的交互呈现，体现数字媒体技术赋能文化传播的核心价值。

#### 3. 文化导向融合原理

文化内涵融入：将民俗寓意、文化常识植入游戏功能，实现技术学习 + 文化认知同步推进。

价值引领渗透：通过挑战 - 通关的游戏主线、正向话术提示，传递坚持不懈、正义向善的精神价值，实现隐性素质塑造。

## 四、实验环境

实验仪器设备：个人电脑

实验软件：Visual Studio、WPS Office 等

## 五、实验预习任务

复习 C++ 核心语法：重点回顾变量定义与初始化（`int`、`string`、`bool` 类型）、`while` 循环（条件判断、循环启停控制）、`if-else/switch` 分支结构（多条件判断、分支执行逻辑），结合简单案例巩固语法应用，避免开发中因基础不牢影响进度。

预习随机数生成相关知识：掌握 `srand((unsigned int)time(NULL))` 随机种子初始化方法，理解 `rand()%n`（生成 0~n-1 随机数）的概率分配逻辑，尝试编写“随机生成 1~10 数字”的小代码，为游戏事件概率触发功能做准备。

调研春节传统文化元素：通过网络、AI 等搜集春节民俗相关知识，记录 3 个以上感兴趣的文化元素，初步思考其转化为游戏功能的思路，为需求分析阶段快速确定核心元素做准备。

梳理控制台游戏开发流程：结合实验任务，在笔记本上绘制“需求分析→框架搭建→功能开发→测试优化”的简单流程，明确每个阶段的核心输出（如框架搭建需完成变量定义、开场 / 结束界面），建立模块化开发思维，提前规划代码结构。

## 六、实验过程与结果

实验过程填写说明：

- 1.程序设计步骤：记录从需求分析到代码实现的关键过程
- 2.关键代码解析：摘抄程序中的核心代码片段，并说明其作用
- 3.调试过程：记录编写或运行中遇到的问题及解决方法

实验结果填写说明：

- 1.运行截图或文字记录：通过文字描述程序运行的完整流程及输出结果
- 2.结果分析：说明程序是否达到预期目标

## 七、实验心得体会

### 1.知识掌握总结

结合本次实验，说明对某些编程方法的理解加深，总结对编程方法的实际应用经验。

### 2.问题与反思

分析实验中遇到的难点，思考改进方向。

### 3.文化、技术感悟

谈谈通过编程实现项目主题的意义，谈谈对 C++ 编程的感悟。

### 4.AI 使用说明

如实记录本次实验中是否使用 AI 工具（代码生成、原理解释、调试建议等），说明 AI 工具对实验的帮助及自身的独立思考。

附录 3：课程考核量规

一级指标	二级指标	三级指标	优秀（9-10 分）	良好（7-8 分）	合格（5-6 分）	改进（0-4 分）	权重占比	评分说明
知识维度 （30%）	语法应用准确性	基础语法掌握	熟练运用数据类型、变量常量定义、函数声明与调用等核心语法，无任何语法错误	掌握核心语法，仅存在 1-2 处非关键性语法疏漏，不影响程序运行	基本掌握核心语法，存在 3-4 处语法错误，需修改后才能正常运行	核心语法掌握不扎实，存在 5 处及以上语法错误，程序无法正常编译	15%	结合代码编译结果、语法规则应用情况评判
		逻辑结构合理性	选择、循环、函数嵌套等逻辑设计严谨，流程清晰，无冗余或遗漏	逻辑结构基本合理，流程清晰，仅存在 1 处轻微逻辑冗余	逻辑结构可行，但存在 2-3 处逻辑衔接不顺畅或冗余	逻辑结构混乱，存在明显逻辑漏洞，导致程序功能无法实现	15%	依据程序执行流程、逻辑判断准确性评判
能力维度 （40%）	模块化编程能力	函数设计与封装	函数划分合理，功能单一明确，封装性强，传参方式优化得当，可复用性高	函数划分基本合理，具备封装意识，传参方式正确，可复用性较好	能进行简单函数划分，封装性一般，传参方式无原则错误，基本可复用	未进行有效函数划分，无封装意识，传参混乱，代码复用性差	10%	结合函数数量、功能划分、传参合理性评判
		问题解决与调试	能快速定位程序 BUG，独立设计多种调试方案，高效解决复杂逻辑问题	能定位主要程序 BUG，设计 1-2 种调试方案，解决核心问题	需借助少量提示定位 BUG，调试方案单一，能解决基础问题	无法定位关键 BUG，缺乏调试思路，依赖他人协助才能修改错误	10%	依据调试过程记录、问题解决效率评判

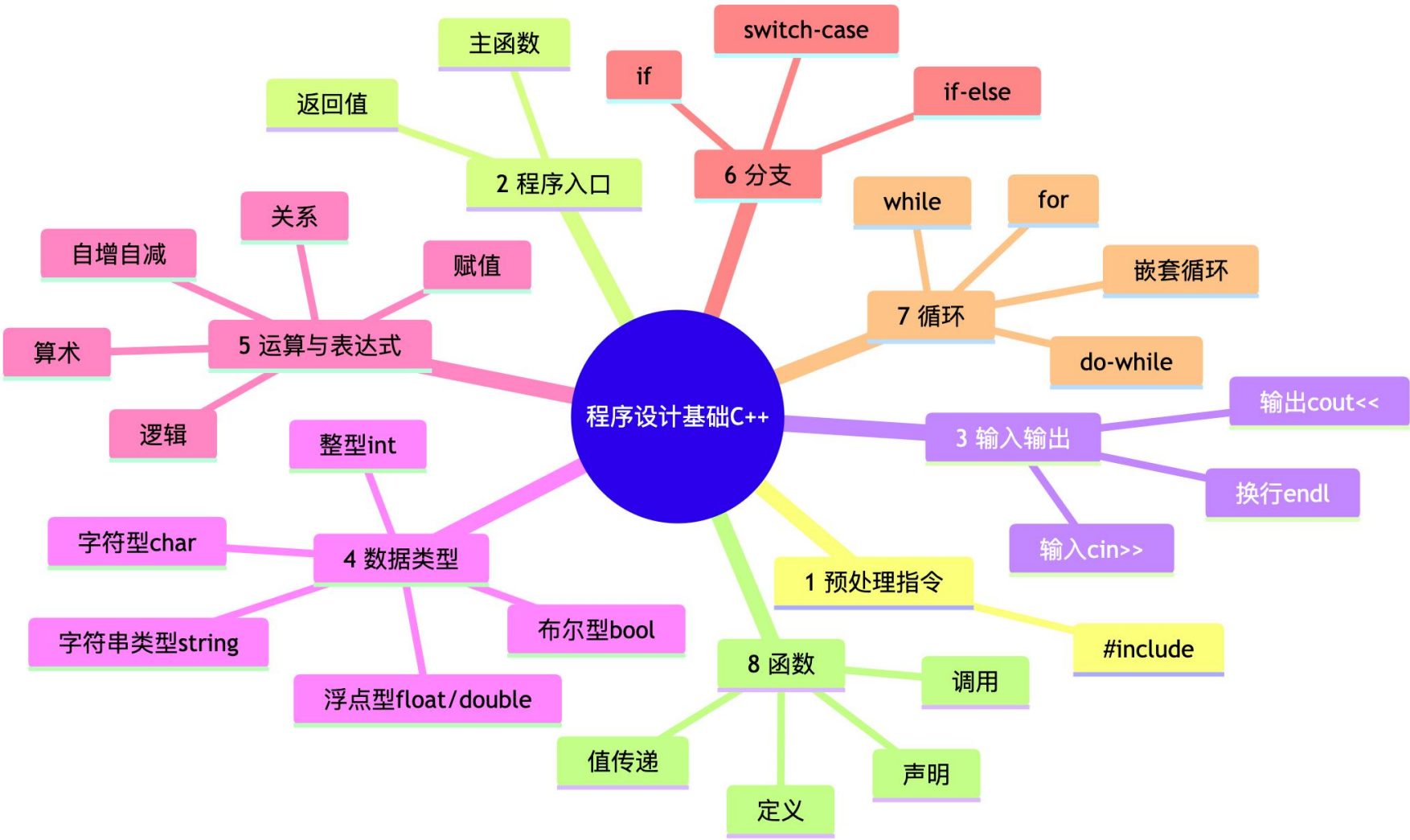


			辑问题			误		
		场景迁移应用	能灵活迁移所学知识，完美适配文化场景或行业需求，实现超出预期的功能	能迁移核心知识适配场景需求，实现基本功能与部分拓展功能	能初步迁移知识适配场景，基本实现核心功能，存在少量适配漏洞	无法将知识与场景结合，仅能实现脱离场景的基础代码，适配性差	10%	结合场景贴合度、功能实现完整性评判
		创新实践能力	自主设计独特功能模块，技术实现有突破，或文化元素融合有创意	能在基础功能上添加 1-2 项个性化功能，具备一定创新意识	能按要求完成基础功能，在细节上有少量个性化调整	仅机械完成基础功能，无任何个性化或创新设计	10%	依据功能创新性、设计独特性评判
素养维度 (30%)	编程规范素养	代码格式与注释	代码缩进统一，命名规范清晰，注释完整详实，可读性极强	代码缩进基本统一，命名规范，注释较完整，可读性较好	代码缩进不统一，命名基本规范，有少量关键注释，可读性一般	代码格式混乱，命名随意，无有效注释，可读性极差	10%	依据代码格式规范性、注释完整性评判
		主动探究意识	主动拓展学习边界，自主查阅资料解决复杂问题，多次优化完善成果	能主动咨询问题，按要求优化成果，有一定探究意愿	需在引导下完成问题解决与成果优化，探究积极性一般	缺乏探究意愿，不主动解决问题，成果无优化痕迹	10%	结合课堂互动、课后咨询、成果优化次数评判

		AI 工具合规使用	严格遵循 AI 使用边界，仅用于语法查询、BUG 排查，核心逻辑完全自主实现	基本遵循 AI 使用边界，偶尔借助 AI 辅助思路，核心逻辑自主实现	存在轻微 AI 使用不规范，核心逻辑有少量 AI 辅助痕迹，需自主修正	违反 AI 使用规则，核心逻辑依赖 AI 生成，存在抄袭风险	10%	依据 AI 使用记录、核心逻辑原创性核查评判
总分	-	-	90-100 分	70-89 分	50-69 分	0-49 分	100%	各项得分累加计算总分，触发否决项直接按不合格判定

附录 4: AI 赋能 C++编程分层提示词

基于 C++ 基础知识点图谱（预处理指令、程序入口、输入输出、数据类型、运算与表达式、分支结构、循环结构、函数 8 大模块），按基础水平（入门阶段）、进阶水平（理解阶段）、挑战水平（应用阶段） 分层设计 AI 提示词，核心目标是引导学生主动思考、自主实践，避免 AI 直接代做，实现 AI 辅助学习而非 AI 替代学习。



# 一、基础水平提示词（适合零基础 / 入门阶段学生）

## 核心定位

帮助理解基础概念、掌握语法格式、完成简单语法验证，提示词需明确已尝试的操作、当前困惑点，引导 AI 提供概念类比、语法纠错方向、简单示例框架。

知识模块	基本提示词（侧重理解、模仿）	挖掘提示词（侧重验证 、拓展）
预处理指令	“我刚学 <code>#include</code> ，知道它要引入头文件，但不知道为什么用能调用 <code>cout</code> ，用能调用 <code>string</code> 类型？能举个生活中“借工具”的例子帮我理解吗？不用直接给代码，先讲清楚逻辑。”	“我试着写了 <code>#include</code> 和 <code>#include "myfile.h"</code> ，发现尖括号和双引号不一样，这两种写法有什么区别？能提示我一个测试方法，比如自己创建一个简单的.h 文件验证吗？”
程序入口	“我写 <code>main</code> 函数时，把 <code>int main ()</code> 写成了 <code>void main ()</code> ，编译器没报错但老师说不对，这两种写法有什么本质区别？能告诉我标准的 <code>main</code> 函数格式里， <code>return 0</code> 的作用是什么吗？”	“我故意把 <code>main</code> 写成了 <code>Main</code> （首字母大写），程序直接报错“找不到入口”，这说明 C++ 对大小写是敏感的对吗？能再举 1 个类似的大小写敏感的例子（比如变量名），我自己测试验证。”
输入输出	“我用 <code>cin&gt;&gt; age</code> 读取年龄时，输入了“二十”而不是 20，程序运行后结果乱了，这是为什么？能提示我怎么判断输入的是不是整数吗？不用给完整代码，讲清思路即可。”	“我发现 <code>cout &lt;&lt; "a" &lt;&lt; "b"</code> 和 <code>cout &lt;&lt; "a" &lt;&lt; endl &lt;&lt; "b"</code> 输出不一样， <code>endl</code> 除了换行还有什么作用？能提示我一个验证方法（比如观察程序运行速度）吗？”
数据类型	“我分不清 <code>int</code> 和 <code>double</code> ，比如存储“身高 175.5 厘米”，用 <code>int</code> 会怎么样？用 <code>double</code> 又会怎么样？能举 2 个具体的数值例子，让我自己计算存储结果吗？”	“我定义了 <code>char c = 'A'</code> ，然后输出 <code>cout &lt;&lt; c</code> 和 <code>cout &lt;&lt; (int) c</code> ，结果不一样，这说明 <code>char</code> 类型本质上存储的是 ASCII 码对吗？能提示我怎么用 <code>char</code> 存储数字字符（比如 '6'）并转成 <code>int</code> 类型的 6 吗？”
运算与表达式	“我计算 <code>5/2</code> 时，结果是 2 而不是 2.5，这和数学里的除法不一样，是因为 <code>int</code> 类型的除法会“截断”小数部分对吗？能告诉我怎么修改才能	“我写了 <code>a = 3; b = ++a;</code> 和 <code>a = 3; b = a++;</code> 发现 <code>b</code> 的结果不一样，这两种自增有什么区别？能提示我一个分步分析的方法，比如先记录 <code>a</code> 的初始值，再看每



	得到 2.5 吗？不用直接给代码，提示变量类型即可。”	一步 a 的变化吗？”
分支结构	“我用 if(score >= 60) 判断及格，想再加一个 “如果 score >= 90 输出优秀” 的条件，应该把新的 if 写在原来的 if 里面还是外面？能画个简单的逻辑图帮我理解吗？”	“我用 switch-case 判断成绩等级时，忘记写 break，结果所有 case 都执行了，这是为什么？能提示我一个测试案例（比如输入 85），让我观察不写 break 时的执行顺序吗？”
循环结构	“我用 for 循环打印 1 到 5，把条件写成了 i < 5，结果只打印到 4，这说明循环条件控制的是 “继续循环的条件” 对吗？能告诉我怎么修改才能打印到 5 吗？”	“我用 while 循环计算 1 到 10 的和，不小心把 sum = 0 写在了循环里面，结果每次循环 sum 都重置为 0，这是变量作用域的问题吗？能提示我怎么调整变量定义的位置吗？”
函数	“我想写一个求两数之和的 add 函数，知道函数名是 add，参数是两个 int，返回值是 int，但不知道函数体里怎么写，能提示我函数体里的核心逻辑（比如用什么运算符）吗？”	“我把 add 函数的声明写在了 main 后面，编译器报错 “未定义标识符 add”，这是为什么？能告诉我函数声明和函数定义的位置关系有什么要求吗？我想自己调整代码验证。”

二、进阶水平提示词（适合有基础 / 理解阶段学生）

核心定位

帮助分析逻辑漏洞、优化代码结构、理解底层原理，提示词需包含已写的代码片段、当前遇到的问题，引导 AI 提供逻辑分析方向、优化思路、原理类比。

知识模块	基本提示词（侧重纠错、 优化）	挖掘提示词（侧重原理、迁移）
预处理指	“我写了 #include 和 using namespace std;，但听说 using namespace std	“我了解到 #define 可以定义宏，比如 #define MAX 100，这和 const int MAX

令	可能有风险，这是为什么？能告诉我怎么不用这句代码也能调用 <code>cout</code> 吗？比如修改 <code>cout</code> 的写法。”	<code>= 100</code> 有什么区别？比如宏在编译时是怎么处理的？能提示我一个测试方法（比如查看预处理后的代码）吗？”
程序入口	“我在 <code>main</code> 函数里调用了其他函数，发现函数调用的顺序会影响结果，比如先调用 <code>a()</code> 再调用 <code>b()</code> ，和先调用 <code>b()</code> 再调用 <code>a()</code> 结果不一样，这说明程序是按代码顺序执行的吗？能再举 1 个类似的顺序影响结果的例子吗？”	“我听说 <code>main</code> 函数的返回值会传给操作系统，比如 <code>return 0</code> 表示正常结束， <code>return 1</code> 表示异常，那操作系统拿到这个返回值后会做什么？能提示我怎么在 <code>Windows</code> 或 <code>Linux</code> 下查看 <code>main</code> 函数的返回值吗？”
输入输出	“我用 <code>cin</code> 读取字符串时，如果输入 “Zhang San”（中间有空格），只能读到 “Zhang”，这是因为 <code>cin</code> 默认以空格为分隔符吗？能告诉我怎么读取包含空格的字符串吗？比如用什么函数替代 <code>cin &gt;&gt;</code> ？”	“我对比了 <code>cout &lt;&lt; endl</code> 和 <code>cout &lt;&lt; "\n"</code> ，发现 <code>endl</code> 会刷新缓冲区，而 <code>"\n"</code> 不会，什么是缓冲区？能举个生活中“快递暂存点”的例子帮我理解吗？刷新缓冲区有什么好处？”
数据类型	“我用 <code>int</code> 存储“人口数量 1400000000”，程序运行后结果不对，查资料发现 <code>int</code> 的最大值是 2147483647，刚好够，但如果存储更大的数（比如 10000000000），应该用什么类型？ <code>long long</code> 吗？能提示我怎么验证 <code>long long</code> 的存储范围吗？”	“我了解到 <code>void</code> 是无类型指针，可以指向任何类型的变量，但为什么不能直接用 <code>void_ptr</code> 访问数据？必须先强制类型转换？能举个例子说明强制类型转换的必要性吗？”
运算与表达式	“我写了 <code>a = (b &gt; c) ? b : c;</code> 这是三元运算符，它和 <code>if-else</code> 实现的逻辑一样，但什么时候用三元运算符更合适？什么时候用 <code>if-else</code> 更合适？能对比 1 个具体的代码场景吗？”	“我计算 <code>a + b</code> 时，如果 <code>a</code> 和 <code>b</code> 都是 <code>int</code> 类型，结果超过 <code>int</code> 范围（比如 <code>a=2000000000</code> ， <code>b=2000000000</code> ），会发生“溢出”，这时候结果会变成什么？能提示我怎么检测溢出吗？比如用什么判断条件？”
分支结构	“我用 <code>if-else if-else</code> 判断成绩等级时，把 “ <code>score &gt;= 90</code> ” 的条件写在了 “ <code>score &gt;= 60</code> ” 后面，结果输入 95 时只输出“及格”，这是逻辑顺序的问题吗？能告诉我分支条件的顺序有什么规则吗？”	“我发现 <code>switch-case</code> 的表达式只能是整数类型（ <code>int</code> 、 <code>char</code> 等），不能是 <code>double</code> 或 <code>string</code> ，这是为什么？从编译器处理的角度，整数类型和非整数类型有什么区别？能简单解释一下吗？”
循环结构	“我用 <code>for</code> 循环遍历数组时，把循环条件写成了 <code>i &lt;= 10</code> ，但数组只有 10 个元素（下标 0-9），结果程序崩溃了，这是“数组越界”吗？能告诉	“我对比了 <code>for</code> 和 <code>while</code> 循环，发现 <code>for</code> 循环更适合“已知循环次数”， <code>while</code> 更适合“未知循环次数”，但其实两者可以互相转换吗？能把 “ <code>for (int i=1;</code>

	我怎么避免数组越界吗？比如用 <code>sizeof</code> 计算数组长度。”	<code>i&lt;=5; i++) { cout &lt;&lt; i; }</code> ”改成 <code>while</code> 循环吗？我想验证转换后的逻辑是否一致。”
函数	“我写了一个函数，参数是 <code>int arr []</code> ，在函数里用 <code>sizeof(arr)</code> 计算数组长度，结果总是 4（32 位系统），而不是数组实际的长度，这是为什么？能告诉我怎么正确传递数组长度吗？”	“我了解到函数可以嵌套调用（比如 <code>a</code> 调用 <code>b</code> ， <code>b</code> 调用 <code>c</code> ），但不能嵌套定义（比如在 <code>a</code> 函数里定义 <code>b</code> 函数），这是 C++ 的语法规则吗？为什么不能嵌套定义？能举个例子说明嵌套定义会导致什么问题吗？”

### 三、挑战水平提示词（适合能力强 / 应用阶段学生）

#### 核心定位

帮助设计综合案例、解决复杂问题、拓展知识边界，提示词需明确项目需求、已有的设计思路，引导 AI 提供方案优化建议、技术选型分析、拓展方向。

知识模块	基本提示词（侧重设计 、解决）	挖掘提示词（侧重拓展 、迁移）
预处理指令	“我想做一个跨平台的程序，在 Windows 下用 <code>#include &lt;windows.h&gt;</code> ，在 Linux 下用 <code>#include &lt;unistd.h&gt;</code> ，怎么用预处理指令（比如 <code>#ifdef</code> ）让程序自动选择引入的头文件？能提示我核心的条件判断逻辑吗？”	“我了解到 <code>#pragma</code> 指令有很多功能，比如 <code>#pragma once</code> 可以防止头文件重复包含，它和 <code>#ifndef ... #define ... #endif</code> 的实现原理有什么区别？哪种效率更高？能提示我一个测试方法（比如编译时间对比）吗？”
程序入口	“我听说 C++ 程序除了 <code>main</code> 函数，还有全局变量的构造函数和 <code>atexit</code> 注册的函数会在 <code>main</code> 前后执行，能告诉我它们的执行顺序吗？比如全局变量 <code>A</code> 、 <code>atexit</code> 注册的函数 <code>B</code> 、 <code>main</code> 函数，执行顺序是 <code>A→main→B</code> 吗？我想自己写代码验证。”	“我了解到有些编译器支持自定义程序入口（比如用 <code>#pragma comment (linker, "/ENTRY:myentry")</code> ），这时候 <code>main</code> 函数还会执行吗？这种自定义入口有什么应用场景？能举 1 个实际的例子吗？”

输入输出	“我想实现一个“带进度条的文件读取”功能，需要实时显示读取的百分比，这时候用 <code>cout</code> 输出会频繁刷新屏幕，影响效率，能告诉我怎么控制 <code>cout</code> 的刷新频率吗？比如用缓冲区的知识优化。”	“我了解到 C++11 以后有 <code>std::cin.ignore()</code> 和 <code>std::cin.clear()</code> 函数，用来处理输入错误（比如输入非数字时清空缓冲区），能详细解释它们的工作原理吗？比如 <code>ignore()</code> 的参数（如 <code>1024, '\n'</code> ）是什么意思？我想自己写一个输入验证的函数。”
数据类型	“我想设计一个“高精度计算器”，能计算超过 <code>long long</code> 范围的整数（比如 100 位的数），不能用现成的库，应该用什么数据类型存储？比如用 <code>char</code> 数组还是 <code>string</code> ？能提示我实现“高精度加法”的核心思路（比如逐位相加 + 进位）吗？”	“我了解到 C++ 有“枚举类型”（ <code>enum</code> ），比如 <code>enum Color { RED, GREEN, BLUE };</code> ，它和 <code>#define</code> 定义的常量有什么区别？比如枚举类型有类型检查，而 <code>#define</code> 没有，能举个例子说明类型检查的好处吗？”
运算与表达式	“我想实现一个“表达式求值”功能，比如输入“ <code>3+4*2-1</code> ”，能计算出正确结果（10），需要处理运算符优先级，这时候应该用什么数据结构？栈吗？能提示我栈的具体用法（比如什么时候入栈、什么时候出栈）吗？”	“我了解到 C++ 有“运算符重载”，比如可以让自定义的类支持 <code>+</code> 、 <code>-</code> 等运算，这和普通的函数重载有什么区别？比如重载 <code>+</code> 运算符时，函数名是什么？能举个“复数类（ <code>Complex</code> ）重载 <code>+</code> ”的简单例子，我想自己扩展减法运算。”
分支结构	“我想做一个“菜单驱动程序”，比如输入 1 执行“添加”，输入 2 执行“删除”，输入 3 执行“退出”，用 <code>switch-case</code> 实现时，发现每个 <code>case</code> 里的代码很多，导致程序臃肿，能告诉我怎么优化吗？比如把每个 <code>case</code> 的逻辑拆分成函数。”	“我了解到“短路求值”，比如 <code>a &amp;&amp; b</code> ，如果 <code>a</code> 是 <code>false</code> ，就不会计算 <code>b</code> ； <code>a</code>
循环结构	“我想实现“冒泡排序”的优化版本，比如当某一轮没有发生交换时，说明数组已经有序，可以提前退出循环，能告诉我怎么修改传统的冒泡排序代码吗？比如添加一个标志变量（如 <code>is_sorted</code> ）。”	“我了解到“递归”和循环可以互相转换，比如用递归计算斐波那契数列，会有重复计算的问题（比如 <code>fib(5)</code> 需要 <code>fib(4)</code> 和 <code>fib(3)</code> ， <code>fib(4)</code> 又需要 <code>fib(3)</code> ），能告诉我怎么用循环优化这个问题吗？或者用“记忆化搜索”的思路？”
函数	“我想做一个“通用的排序函数”，能对 <code>int</code> 数组、 <code>double</code> 数组甚至 <code>string</code> 数组排序，不用写多个重载函数，这时候应该用什么技术？函数模板（ <code>template</code> ）吗？能提示我函数模板的核心语法（比如	“我了解到“函数指针”可以指向不同的函数，比如用一个函数指针变量，既可以指向 <code>add</code> 函数，也可以指向 <code>sub</code> 函数，这在“回调函数”中有应用，能举个例子说明回调函数的用法（比如排序时自定义比较规则）吗？我想自己实现一个带回调

	template ) 吗? ”	的排序函数。”
--	-----------------	---------

## 四、提示词使用规范（必看）

- 1. 拒绝 “裸需求”：不直接问 “帮我写一个求素数的代码”，而是说 “我想写求素数的代码，目前的思路是遍历 2 到  $\sqrt{n}$  判断是否能整除，但不知道怎么优化循环效率，能提示我吗？”
- 2. 提供 “上下文”：必须包含已做的尝试（代码片段、思路描述）和具体困惑（逻辑错误、原理疑问），让 AI 明确辅助方向。
- 3. 控制 “辅助度”：要求 AI 提供思路、方向、方法，而非完整代码、直接答案，比如问“怎么修改循环条件”而不是 “帮我改好代码”。
- 4. 验证 “正确性”：拿到 AI 的提示后，必须自己写代码验证，比如 AI 说 “用 long long 存储大整数”，要自己测试不同数值的存储结果，确认是否正确。

通过以上分层提示词，可根据自身能力和学习进度，合理借助 AI 解决问题，逐步培养独立编程思维，真正实现 AI 为我所用，而非替我所为。





附件 1：《 课程名称 》全过程非标准化考核改革的评分表（院级评审）

一级指标	二级指标	三级指标	指标解读	院评得分	申报材料是否佐证
考核设计（30分）	1.1 目标适配性(10分)	1.1.1 培养目标对齐度	考核任务与课程目标高度一致，覆盖知识、能力、素养三个维度，无单一维度偏废。		
	1.2 场景任务设计真实性（10分）	1.2.1 场景真实性	跳出传统经验命题局限，至少 1 个任务的背景、数据、问题源于真实行业、社会或前沿案例，贴近实践。		
		1.2.2 命题多元性	建立校内教师+AI+行业专家（至少 1 名）等多元主体协同命题机制。		
	1.3 命题高阶性(10分)	1.3.1 任务高阶能力导向	记忆类题目≤30%（根据课程目标调整），提升考查学生批判性思维、知识迁移与应用、创新实践、问题解决等高阶能力考查题目占比，问题具有灵活性、探究性和开放性。		
		1.3.2 任务综合性与挑战度	题目具有整合性，难易梯度清晰，能有效区分不同能力水平学生；无超纲或无意义难题，挑战度贴合学生认知水平与课程要求。		
评价实施（40分）	2.1 评价标准开放性（15分）	2.1.1 指标明确性与可视度	无“千人一面”的标准化要求，建立分级评价量表，每个维度的指标可观察、可考量、可评价、可验证，有效降低评阅人主观偏差，避免主观臆断。		
		2.1.2 开放性与限制性	高阶试题答案具有开放性（无现成答案可找），尊重学生思维和方法差异性，允许选择不同技术路径、研究方法或成果形式，预留创新空间；需在评价标准中明确 AI 工具的使用场景、使用边界及违规判定标准，避免 AI 抄袭。		
		2.1.3 标准公示及时性	评价标准（含评价维度、分级指标、评分权重等）在考核任务启动前向学生完整公示，明确告知“如何评、评什么”，避免学生学习方法偏差以及成果偏离要求，助力实现“评价即学习”。		
	2.2 过程培养有效性（15分）	2.2.1 教学过程互动性与引导性	教学方法善于融入互动、引导探究、组织讨论，调动学生积极性，激发潜能；学习任务具有阶梯式，引导学生能力逐级提升，杜绝“教学空转、期末叠加难度”断层现象。		
		2.2.2 过程支持与技术赋能	依托信息化平台、AI 等工具，赋能学生自学自评与能力提升，通过过程跟踪、成果答辩核验等方式验证成果真实性，避免成果代做。		
		2.2.3 及时个性化反馈	在学习任务关键节点嵌入精准反馈与及时指导，而非仅给出分数或笼统评语，提供学生个体的具体改进建议和学习支持，解决学生能力短板。		

一级指标	二级指标	三级指标	指标解读	院评得分	申报材料是否佐证
	2.3 评分公平性(10分)	2.3.1 评分客观公信力	试批试评校准评分尺度；运用 AI 等技术工具赋能评阅，提升效率与精准度；建立健全多层级评分复核机制，学生成绩复查量低且复查结果零差错。		
		2.3.2 多元评价融合	融合教师评价、学生互评、行业专家点评等多元主体；互评通过制定成果贡献度量化评分项、评价主体回避、异常值筛查等机制规则，避免“搭便车”、印象打分、成果归属或责任划分不清等现象。		
改革成效（30分）	3.1 学生能力提升（15分）	3.1.1 学习主动性激发	学生参与过程性互动研讨、主动优化成果的比例显著提升，形成以考促学、以学促能的良性循环。		
		3.1.2 高阶能力达成	以学生作答及成果为核心证据，有效印证学生在理论联系实际的应用、跨域知识整合、复杂问题拆解、创新方案设计与实践落地等方面成效，无死记硬背、抄袭拼凑现象及 AI 作弊行为。		
	3.2 成果价值（10分）	3.2.1 实践应用价值	学生成果具备明确的问题解决逻辑与实践价值，或具备落地潜力，而非单纯的任务作业。		
		3.2.2 创新性与个性化	成果体现学生独特思考或个性化表达，具有原创突破，无同质化现象。		
	3.3 持续改进（5分）	3.3.1 教与学优化联动	学生满意度不低于 80%，学生自我评估良好；改革有效解决原有考核问题，并基于考核结果调整课程教学策略，实现以考促教的闭环。		
合计得分：					-

说明：1.总分采用 100 分制，按二级指标细化评分，求和计算最终得分。

2.自评与评审给分：二级指标按三档评分——“达标”（**满分**）：完全符合指标要求，核心量化指标达标，无任何偏差、“基本达标”（满分的 **60%—80%**）：符合指标核心要求，次要量化指标存在轻微偏差（不超过 10%），无实质性影响；“不达标”（**0—59%**）”：未满足指标核心要求，或触发相关否决项关联条件。

3.核心否决项：若存在以下情形之一，直接判定为“不达标”（分数不高于 59 分）：

- ①命题开放性不足（如记忆类题目占比超过 30%且未设置开放性或高阶考查题目）；
- ②评价标准缺失，或标准表述模糊、无明确评判依据、无高阶能力评价、不可操作；
- ③过程评价缺乏有效支撑材料证明过程性培养学生能力发展；
- ④学生成果存在抄袭、拼凑行为，或经核查确认存在 AI 作弊情况。