Creative Education Studies 创新教育研究, 2026, 14(1), 693-707
Published Online January 2026 in Hans. https://www.hanspub.org/journal/ces
https://doi.org/10.12677/ces.2026.141085

Hans汉斯

设计思维下高职Python课程教学模式探索与实践

赵美娇[1]，马　冰[2]，沈鸿斌[1]

[1]海南政法职业学院公共安全技术系，海南　海口
[2]海南警察学院网络安全与执法系，海南　海口

收稿日期：2025年11月25日；录用日期：2026年1月20日；发布日期：2026年1月28日

摘　要

针对高职Python课程中存在的知识碎片化、产学脱节与创新能力培养不足等问题，本研究基于设计思维理论，构建了“双项目进阶、五阶段迭代”教学模式。以“猜数字游戏”与“我爱记单词”两项目为载体，系统融入共情、定义、构思、原型、测试五阶段。教学实践表明，该模式显著提升了学生的知识整合与创新能力，实验组综合得分率比对照组提高13.98%，尤其在综合应用与创新设计题型上表现突出($p < 0.05$)。质性分析显示，课堂氛围和学生的创新自信、团队协作与学习动机都有显著增强。本研究为高职信息技术课程从“技能传授”向“思维与能力并重”转型提供了可行路径。

关键词

设计思维，教学模式，Python教学，双项目进阶，创新能力

Exploration and Practice of Teaching Mode for Python Courses in Higher Vocational Education Based on Design Thinking

Meijiao Zhao[1], Bing Ma[2], Hongbin Shen[1]

[1]Department of Public Safety Technology, Hainan Vocational College of Political Science and Law, Haikou Hainan
[2]Department of Cybersecurity and Law Enforcement, Hainan Police College, Haikou Hainan

Received: November 25, 2025; accepted: January 20, 2026; published: January 28, 2026

Abstract

Addressing the issues of knowledge fragmentation, disconnection between production and learning,

文章引用: 赵美娇, 马冰, 沈鸿斌. 设计思维下高职 Python 课程教学模式探索与实践[J]. 创新教育研究, 2026, 14(1): 693-707. DOI: 10.12677/ces.2026.141085

and insufficient cultivation of innovation ability in Python courses in higher vocational education, this study constructs a "dual-project progression, five-stage iteration" teaching model based on design thinking theory. Using the "Guess the Number Game" and "I Love Memorizing Words" projects as carriers, the model systematically integrates the five stages of empathy, definition, ideation, prototyping, and testing. Teaching practice shows that this model significantly enhances students' knowledge integration and innovation ability. The experimental group achieved a 13.98% higher comprehensive score rate compared to the control group, particularly excelling in comprehensive application and innovative design questions ($p < 0.05$). Qualitative analysis reveals that classroom atmosphere, students' confidence in innovation, team collaboration, and learning motivation have all been significantly enhanced. This study provides a feasible path for the transformation of information technology courses in higher vocational education from "skills impartation" to "emphasizing thinking and ability".

Keywords

Design Thinking, Teaching Model, Python Teaching, Dual-Project Progression, Innovation Ability

1. 引言

Python 作为高职信息技术类专业首选的入门语言，其教学目标不仅是语法掌握，更在于计算思维与解决问题能力的培养[1] [2]。然而，现实教学效果与这一目标存在显著差距，主要体现在三个方面：知识传授的碎片化，导致学生难以整合所学以应对综合项目，这种知识与应用的脱节，严重削弱了学生的学习成就感和内驱力；教学目标与产业需求的错位，使学生仅仅局限于语法层面的机械训练，缺乏对真实业务场景的模拟和对用户需求的深度挖掘，缺乏“以用户为中心”的产品思维和创新能力，难以满足当前产业界对技术技能人才的复合型要求；同时，教学过程中创新思维培养的系统性缺失，限制了学生创造性地解决问题的机会，很难培养和锻炼学生在面对不确定问题时的韧性和创新自信。

因此，高职 Python 课程改革的根本需求并非只是教学案例的简单更新，而是一场从教学理念到实施路径的全面重塑。当前，Python 课程改革在技术赋能与模式创新方面已取得显著成效，但相关研究仍主要聚焦于教学工具的升级和知识传递效率的提高，对于如何系统化培养学生应对复杂现实问题的“创新解决问题的能力”仍缺乏足够关注[3]-[6]。当前，项目式学习(PBL)已成为高职编程教学中常用的方法，其强调在真实项目中整合知识与技能。然而，传统 PBL 往往侧重于任务的完成与功能的实现，缺乏对“用户需求洞察”“迭代优化”与“体验设计”的系统性培养[7]。本研究提出的设计思维教学模式，并非对 PBL 的简单替代，而是在其基础上，强化了“以用户为中心”的起点意识、“原型–测试”的迭代闭环，以及“问题定义–方案生成”的创造性过程，从而更贴近数字产品开发的真实流程，也更有利于培养学生面对复杂、开放问题的创新应变能力。为此，我们引入一种能够有效整合知识、对接实际需求并激发创新的方法论——设计思维(Design Thinking)，为重构 Python 教学模式提供新的理论视角与实践框架。将设计思维融入教学，旨在构建以学生为中心、以真实问题为驱动、以迭代优化为路径的新型教学模式，从而有效应对传统教学的局限，为培养符合时代需求的创新型信息技术人才提供可行的实践路径[8]。

2. 设计思维概念解析与教学模型构建

2.1. 设计思维的内涵及其教育应用价值

设计思维的重点是“以用户需求为中心”，是“用户为王”时代一种能够贯穿“理解问题－定义方案－创造实现－评估优化”全过程，用于解决复杂问题的创新方法论[9]。其核心在于强调同理心(深入理解用户)、协作(跨领域团队合作)与迭代(在试错中持续优化)。这与我们培养高职学生解决真实世界问题的目标高度契合。将设计思维应用于编程教育，其价值在于实现三大转变：一是从“语法导向”到“问题导向”的转变：学习的起点不再是某个语法点，而是理解并解决真实世界存在的问题，使编程成为一种有意义的解决问题的工具。二是从“教师中心”到“学生中心”的转变：学生不再是知识的被动接受者，而是主动的探索者、设计者和创造者，教师角色转变为引导者与协作者。三是从“一次性交付”到“持续性迭代”的转变：让学生摒弃“一遍写对”的思维定式，将循环迭代优化视为学习过程中必不可少且极具价值的环节，培养学生应对复杂性的韧性与精益求精的工程精神。

2.2. “五阶段迭代”教学模型的构建

本课程根据教学需要，对经典的斯坦福 D. School 五阶段模型[10]进行了教育化改造，结合高职学生的认知特点与 Python 课程的教学规律，构建了如图 1 所示的教学应用模型。该模型将设计思维的五个经典阶段——共情、定义、构思、原型、测试——转化为可执行、可评估的教学活动序列，构建了一个以“测试”驱动“迭代”的闭环学习流程。

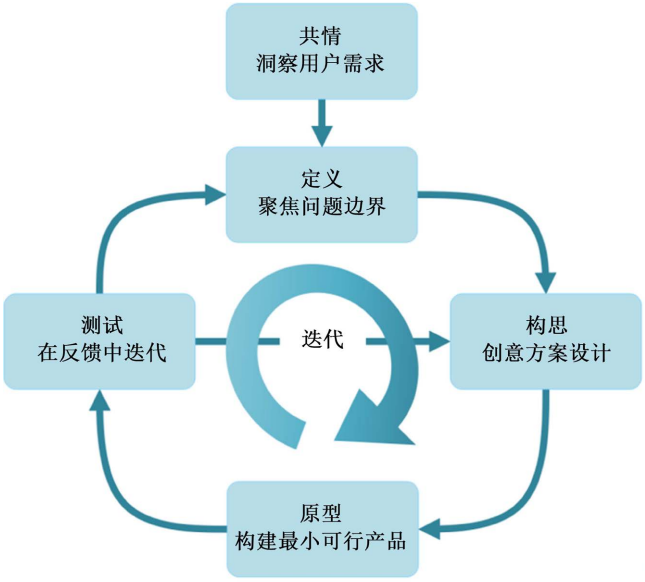

Figure 1. Teaching model of the “Five-Stage Design Thinking” process
图 1. “设计思维五阶段”教学模型

1) 共情阶段：从用户场景中发现问题

共情阶段的目标是引导学生跳出“程序员”视角，以“设计师”的姿态去理解终端用户的真实处境与需求。在教学实施中，我们通过角色扮演、用户访谈、情境观察等方式实现。通过这一过程，使学生深刻体会到，一个优秀程序的价值首先源于它对用户痛点的精准洞察。

2) 定义阶段：将问题转化为功能需求

定义阶段是在充分共情的基础上，学生需要从收集到的、可能杂乱无章的用户需求中，提炼出待解决的核心问题，并将其精确地定义为程序的功能边界。这一阶段是训练学生抽象思维与逻辑归纳能力的关键。最终，学生需要形成一份清晰的、共识性的“产品功能清单”。

3) 构思阶段：从功能到技术方案的创造性转化

构思阶段是连接“做什么”与“怎么做”的桥梁。学生以“功能清单”为着力点，通过头脑风暴、思维导图、方案草图等形式，探索实现各项功能的各种可能技术路径。其核心产出是算法流程图或系统结构图。教师在此过程中鼓励发散性思维，引导学生比较不同方案的优劣，并综合考虑技术可行性、时间成本与用户体验，最终选择一组最优的技术实现路径。这本质上是一个创造性解决问题的过程。

4) 原型阶段：模块化开发与最小可行产品构建

原型阶段是传统教学的重点，但在本模型中，编码实现被赋予了新的意义——构建“可测试的原型”。我们强调模块化编程，指导学生将复杂系统拆分为多个功能单一、接口清晰的函数或类，并分步实现。这不仅能降低开发复杂度，更能够培养学生的工程化思维。在此阶段，学生不再追求一次性实现所有功能，而是优先构建一个包含核心功能的“最小可行产品”(MVP)。

5) 测试阶段：在反馈循环中驱动学习深化

测试阶段是设计思维最具特色的环节，也是传统教学最为缺失的一环。学生需要将自己的 MVP 原型交付给真实用户(同学或其他目标用户)使用，并仔细观察使用过程、收集反馈意见。无论是发现程序漏洞(Bug)，还是接收到“操作不顺手”“提示不清晰”等体验层面的批评，均被视为宝贵的迭代信号。学生基于反馈，重返“构思”或“原型”阶段，进行代码的修订与功能的完善。这个“测试–反馈–修改”的循环可能会进行多次，学生在不断的迭代修改过程中深刻体会到“软件是成长起来的，而非一次建成的”，在此过程中，他们的批判性思维、抗挫折能力和追求卓越的工程素养也得到实质性锤炼。

此五阶段模型构成了一个紧密闭环，它并非简单的流程更新，而是将编程从一种单纯的技能训练，升华为一个完整的、创造性的问题解决与价值创造过程。

3. 基于设计思维的课程教学实施与成效

本研究摒弃了零散案例的堆砌，采用了“双项目进阶”的策略，将图 1 的设计思维五阶段教学模型完整地嵌入到“猜数字游戏”和“我爱记单词”这两个综合性、渐进式的教学项目中。项目的选择基于案例学习理论[11]，并深度契合能力培养与区域产业需求，遵循真实性、渐进性与可迁移性原则。

3.1. 教学实施：“双项目进阶”式教学模式

项目一：“猜数字游戏”。此项目作为入门项目，逻辑清晰、目标明确，初始版本仅有固定数字猜测功能，在共情与测试后，逐步迭代升级到 2.0 版本，包含了提示语、随机数猜测功能，3.0 版本包含了限制猜测次数并显示余额功能，4.0 版本引入了规则难度选择与函数封装的复杂逻辑，完美承载了从变量、分支、循环到函数封装的基础知识。在“共情”阶段，学生需思考“什么样的游戏规则对玩家有吸引力”；在“测试”阶段，学生邀请同学试玩，并根据反馈调整难度、提示信息甚至增加“充值”或“成就”系统，将一个简单的逻辑练习拓展为一个健壮的、充满设计思考的微型产品。其迭代路径清晰地体现了学生能力的进阶。

项目二：“我爱记单词”小程序。此项目作为进阶项目，要求学生处理更复杂的数据结构与用户交互。其培养的“数据处理 + 应用开发”复合能力，能够很好地适应各类数字化应用场景的开发需求。

在共情和定义阶段，我们并不直接布置编码任务，而是要求学生先去访谈至少两位身边的英语学习

者，了解他们在背单词过程中遇到的真实困扰(如“容易遗忘”“过程枯燥”“不知如何检验效果”等)。通过创建用户画像(如英语四六级备考学生)和用户旅程图，精准定位“易遗忘、过程枯燥”等痛点，并明确定义“错题回顾、游戏化学习机制”等功能；针对“如何检验效果”，则可定义为“开发错题回顾与成绩统计功能”。

在构思阶段，一个典型的技术决策点是词库的存储与读取方式——要临时存储还是永久存储？这决定了使用列表、字典还是外部文件。学生通过思维导图分析不同数据结构的优劣，在 1.0 版本必然选择轻量级的列表结构，为后续迭代加入文件读写功能预留出足够空间。

在原型阶段，“我爱记单词”的 MVP 可以只是一个能从词库随机抽取单词并判断对错的控制台程序。面对学生普遍遇到的困难(如怎样将“测试”与“记录错题”两个模块独立开来？)，教师可以通过引导，让学生学会设计独立的测试函数和错题记录函数，并通过全局变量或字典传递数据，初步建立高内聚、低耦合的工程化思想。

测试与迭代阶段，学生使用故事板构建测试场景，以情景剧形式演示小程序如何解决用户问题，并根据反馈对界面、词库算法等进行多轮优化，在后续迭代中将图形界面、用户登录、数据可视化等高级功能加入进去。

这两个项目构成了一个从逻辑训练到系统构建、从控制台程序到潜在 GUI 应用的能力螺旋上升通道，所有语法知识都在解决项目推进过程中的真实需求中被自然引出和反复应用，彻底打破知识壁垒，让所有语法知识能够得到反复锤炼。

3.2. 实施成效

1) 成效评估方法

为科学评估此教学模式的效果，本研究采用解释性序列混合方法设计。研究对象为某高职院校一年级两个平行班，实验组(46 人)采用本文构建的设计思维教学模式，对照组(45 人)采用传统分散案例教学。通过编程基础前测确保两组学生在实验前无显著差异($p > 0.05$)。

量化数据来源于期末统一命题的考试成绩，试卷涵盖基础语法、算法设计、程序实现、综合应用与创新设计五种题型。采用 SPSS 26.0 对成绩进行独立样本 t 检验，并计算效应量(Cohen’s d)以评估实际差异幅度。

质性数据来源于：1) 对实验组 20 名学生的半结构化访谈(依据创新自信、团队协作、学习动机等维度设计提纲)；2) 收集全部 46 名实验组学生的项目反思日志。采用主题分析法对质性文本进行系统编码与分析，以深入解释量化结果背后的内在机制。

2) 实践成效分析

经过一学期的教学实验，量化数据清晰展示了新模式的优越性。如表 1 所示，实验组在各项考核中均优于对照组，尤其在综合应用题和创新设计题上，得分率领先幅度最大(分别达 14.78%和 10.64%)，且独立样本 t 检验显示，实验组和对照组的综合成绩存在统计学上差异显著 ($t(89) = 4.217, p < 0.001$)，且效应量 Cohen’s $d = 0.89$，远高于 0.8 的临界值，表明此教学模式具有巨大的实际效应。这证明了设计思维可以有效提高学生对知识的融会贯通与创造性应用能力，能够解决知识碎片化的核心痛点。

除期末考试成绩外，我们还采集了学习过程数据，发现实验组学生的代码提交频率平均比对照组高 25%，且代码修改中属于“功能增强”和“体验优化”的比例达到 31%，远高于对照组的 8%(后者修改多为修正语法错误)。这表明设计思维模式有效激发了学生持续优化作品的内在动机。

3) 质性分析结果

通过对 20 名实验组学生的半结构化访谈文本和 46 份项目反思日志进行主题分析，我们采用 Braun

& Clarke 提出的六步主题分析法[12]，进行了开放式编码、主题提炼与验证。以下为部分匿名学生原话摘录，用以支撑核心发现：

Table 1. Comparison of score rates between the two student groups (Unit: %)
表 1. 两组学生得分率对比(单位：%)

题型	基础语法题	算法设计题	程序实现题	综合应用题	创新设计题	综合得分	代码迭代深度指数*
实验组	73.70	65.87	74.35	75.00	74.49	73.68	3.2
对照组	66.67	62.67	67.11	60.22	63.85	64.10	1.1
提升幅度	7.03	3.20	7.24	14.78	10.64	13.98	1.91

*注：代码迭代深度指数 = 学生项目平均有效迭代次数(指基于反馈的功能性修改)。

创新自信提升：实验组超过 35%的学生在期末自主项目中，主动探索并应用了课程未要求的第三方库或 API，而对照组该比例仅为 12%。学生 S03 表示：“以前觉得编程就是写代码，现在会先想‘用户需要什么’。我在单词项目里自己加了个‘记忆曲线复习’功能，虽然代码写得慢，但特别有成就感。”学生 S15 则表示：“第一次测试时同学说我的游戏太难，我一开始很沮丧，但后来意识到这是改进的机会，不是失败。”

工程思维形成：学生普遍经历了“原型－测试－修改”的多次循环，走出了“一次就写成”的思维定式。他们开始自发地进行代码审查、撰写文档、规划迭代路线，展现出初级工程师的思维习惯。学生 S08 在日志中表示：“我们小组为‘猜数字’写了三个版本，每次测试后都会开会讨论怎么改。最后一次我们甚至画了用户体验路径图，这是我以前从未想过要做的。”

学习动机与课堂氛围转变：课后访谈中，85%的实验组学生表示对编程抱有持续的学习兴趣，远高于对照组的 62%。学生 S11 反思道：“在传统课堂中，学生遇到报错时普遍表现出沮丧和依赖。而在设计思维课堂中，错误和用户的负面反馈被正常化，学生将其视为‘迭代的信号’，小组内会主动讨论‘我们下一步可以怎么改’，解决问题的自主性和韧性明显增强。”

负面或矛盾案例：值得注意的是，有少数学生(如 S12)在访谈中表示：“迭代过程太耗时，有时觉得在重复劳动。”另有学生(如 S21)反馈：“团队中有人参与度低，导致项目进度不均。”这些反馈提示我们在未来教学中需加强过程辅导与团队管理策略，并优化时间分配。

主题分析最终提炼出四大主题：创新自信、工程思维、协作动机、迭代认知。这些质性发现与量化结果相互印证，揭示了设计思维教学模式影响学生能力与态度的内在机制。

4. 总结与反思

本研究将设计思维引入编程教育，其适用性仍存在一定争议。例如，有学者认为设计思维更适用于开放性、非结构性问题，而编程教学中部分基础语法与算法训练具有较强结构化特征[13]。本研究的实践表明，通过“双项目进阶”设计——从结构清晰的“猜数字游戏”到开放性的“我爱记单词”——设计思维可在不同复杂程度的问题中灵活嵌入，既可用于功能迭代，也可用于体验优化。这表明，设计思维在编程教育中并非万能，但通过合理的教学设计与项目阶梯，可有效弥合“技能训练”与“创新实践”之间的鸿沟。

本研究构建并实践了“双项目进阶、五阶段迭代”的教学模式，为高职 Python 课程从“技能传授”向“思维与能力并重”转型提供了一种有效的教学模式探索。实践表明，该模式能显著提升学生的知识整合与创新能力，尤其在综合应用与创新设计方面表现突出。总体而言，本研究在国际理念的本土化、

从通识工程训练向专业课程教学的深化、以及设计思维教学流程的规范化与工具化方面，做出了具有特色的探索，丰富了设计思维在教育领域应用的理论与实践图景。

然而，实践也反映出三大挑战，并催生了相应的对策思考：

首先，教师角色转型要求教师从“知识的权威”转变为“设计的引导者”和“迭代的教练”。这不仅需要教学理念的更新，更要求教师自身具备一定的项目实战经验和对设计思维工具的熟练运用。院校可以通过组织“设计思维教学工作坊”，鼓励教师参与企业实践，进而提升其非讲授式教学的能力。

其次，迭代过程与固定课时的矛盾确实存在。设计思维的深度迭代需要时间，这与紧凑的教学计划形成冲突。我们探索的解决路径是“课内引导 + 课外迭代”的混合模式。即在课内完成共情、定义、核心构思和 MVP 构建，将测试和后续迭代环节置于课外项目时间，并通过线上平台(如超星学习通、GitHub Classroom 等)进行过程管理与协作。

最后，传统评价体系对设计过程衡量的不足是关键瓶颈。为解决此问题，我们构建了“过程 + 作品”的综合评价体系。过程评价要关注学生在各阶段的产出物(如用户画像、思维导图、迭代日志)，占 40%；作品评价则不仅要看功能实现，更关注其用户体验、代码质量与创新性，占 60%。这一体系可以更全面地反映学生的能力成长。

未来，我们将在以下方面继续探索：一是研究人工智能辅助编程工具(如 GitHub Copilot)在学生的“构思”与“原型”阶段所能扮演的角色，是促进创新还是固化思维。二是深化产教融合，引入来自合作企业的真实、小微型项目需求，让学生在更为逼真的场景中运用和锤炼设计思维，为产业一线输送更多具备创新能力和产品思维的高素质技术技能人才。

附件建议：为增强研究的可复制性与现实指导价值，我们提供了以下配套材料：1) “猜数字游戏”与“我爱记单词”项目教学设计方案；2) 设计思维各阶段产出物评价量表；3) 学生项目迭代日志模板；4) 期末综合评价表。

基金项目

海南省高等学校教育教学改革研究资助项目(项目编号：Hnjg2022-148)；海南政法职业学院院级项目(项目编号：kyyb2009)。

参考文献

[1] 周钢, 李永杰, 郭晖, 等. 以 Python 推动计算思维培养落地的大学计算机基础课程教学[J]. 计算机教育, 2022(2): 79-82.

[2] 仲冰. 结合实例开展 C 语言程序设计教学[J]. 科技视界, 2021(35): 44-45.

[3] Henriksen, D., Gretter, S. and Richardson, C. (2018) Design Thinking and the Practicing Teacher: Addressing Problems of Practice in Teacher Education. *Teaching Education*, **31**, 209-229. https://doi.org/10.1080/10476210.2018.1531841

[4] 刘海鹏. 大语言模型全链赋能的“Python 程序设计”课程教学新范式研究[J]. 无线互联科技, 2025, 22(18): 125-128.

[5] 周丽蓉, 全华凤, 李宜汀, 等. 财经高校 Python 程序设计课程改革实践路径[J]. 计算机教育, 2025(9): 202-207.

[6] 王昕天, 胡畔. 启发式教学在工具类课程中的应用效果与经验启示——以 Python 数据分析课程为例[J]. 高教学刊, 2025, 11(22): 119-124.

[7] Retna, K.S. (2015) Thinking about “Design Thinking”: A Study of Teacher Experiences. *Asia Pacific Journal of Education*, **36**, 5-19. https://doi.org/10.1080/02188791.2015.1005049

[8] 田劲松, 刘新, 李军利, 等. STEM 教育视域下跨学科整合设计思维的教学模式构建与实践研究[J]. 高等理科教育, 2025(5): 96-104.

[9] 朱龙. 设计思维: 一种面向 21 世纪教育创新的实践框架[J]. 数字教育, 2020(1): 32-35.

[10] 韩少文, 田原, 韩鸿远. “超学科”视野下的创意“泛化”: 设计思维课程教学新探索[J]. 设计, 2024, 37(13): 87-90.

[11] Kolodner, J.L. (1993) Case-Based Reasoning. Morgan Kaufmann Publishers.

[12] Braun, V. and Clarke, V. (2006) Using Thematic Analysis in Psychology. *Qualitative Research in Psychology*, **3**, 77-101. https://doi.org/10.1191/1478088706qp063oa

[13] Razzouk, R. and Shute, V. (2012) What Is Design Thinking and Why Is It Important? *Review of Educational Research*, **82**, 330-348. https://doi.org/10.3102/0034654312457429

附件材料：高职 Python 课程设计思维教学模式实施方案

附件一：“双项目进阶”教学设计方案

项目一：猜数字游戏(16 课时)

项目目标：

1) 掌握 Python 基础语法(变量、数据类型、输入输出)

2) 理解分支结构和循环结构

3) 初步体验设计思维五阶段流程

4) 培养基本的用户思维和迭代意识

阶段安排：

阶段	课时	教学活动	产出物	教学目标
共情	2	角色扮演：学生两人一组，分别扮演游戏玩家和开发者，体验不同难度的猜数字游戏，记录玩家的感受和痛点	用户痛点清单(3~5 条)	理解游戏玩家的真实需求
定义	2	小组讨论：基于痛点清单，定义游戏的核心功能和改进方向	功能清单 V1.0 (含优先级排序)	将模糊需求转化为具体功能
构思	2	头脑风暴：如何实现不同难度级别？如何增加趣味性？流程图绘制	系统流程图、技术方案草图	建立从功能到技术的思维桥梁
原型	6	分布编码实现： 1) V1.0：基础版本(固定数字) 2) V1.1：增加随机数和提示 3) V2.0：封装为函数，增加难度选择	1) 可运行的 Python 程序文件(.py) 2) 版本说明文档(记录 V1.0/V1.1/V2.0 功能差异)	掌握模块化编程与版本管理意识
测试	4	交叉测试：小组间互换程序试玩，记录 Bug 和体验问题 迭代优化：基于反馈修改代码	1) 测试反馈表、迭代日志 V1.0、优化后的程序 2) 迭代日志(完整记录 V1.1→V1.2→V2.0→V2.1 的修改内容、原因与效果) 3) 最终可运行程序 V2.1	建立“测试→反馈→迭代”的闭环意识，培养持续优化习惯

技术要点衔接：

第 1~4 课时：input()、print()、int()转换、比较运算符

第 5~8 课时：if-elif-else、random 模块

第 9~12 课时：while 循环、break

第 13~16 课时：函数定义与调用

项目二：我爱记单词(24 课时)

项目目标：

1) 掌握列表、字典、文件读写等数据结构

2) 理解函数封装和模块化设计

3) 深入应用设计思维解决复杂问题

4) 培养工程化思维和团队协作能力

阶段安排：

阶段	课时	教学活动	产出物	教学目标
共情	4	用户访谈：每组访谈 2~3 名英语学习者 创建用户画像和用户旅程图	1) 用户画像×2 2) 用户旅程图 3) 需求洞察报告	深度理解用户场景和痛点
定义	2	需求聚类分析：将访谈结果分类，定义核心功能和扩展功能	产品功能规格说明书(含 MVP 定义)	精准定位产品边界
构思	4	技术方案设计： 1) 词库数据结构选择(列表 vs 字典) 2) 功能模块划分 3) 数据持久化方案	1) 系统架构图 2) 模块接口定义 3) 技术选型说明	培养系统设计能力
原型	8	模块化开发： 1) V1.0：实现控制台版核心背词与测试功能 2) V1.1：增加错题记录模块 3) V2.0：引入文件存储，实现词库持久化	分模块的 Python 代码、版本迭代说明(V1.0→V1.1→V2.0 的功能演进)	掌握工程化开发流程与版本规划能力
测试	6	1) 可用性测试：邀请真实用户(英语学习者)测试 MVP，收集反馈，基于反馈增加功能，迭代高阶版本 2) A/B 测试(V1.1)：对比两种复习算法的用户记忆效果 3) 迭代至 V2.0：根据测试结果优化算法(如实现本地化保存) 4) 扩展功能展望：小组规划 V3.0 方向(如图形界面、数据统计)	1) 可用性测试报告 2) A/B 测试数据分析摘要 3) 完整迭代日志(记录 V1.0→V1.1→V2.0 的关键决策与用户反馈依据) 4) 最终可交付程序 V2.0 + V3.0 功能规划书	建立数据驱动优化的意识，培养从用户反馈中识别产品演进方向的能力

技术进阶路径：

✧ 数据结构：list→dict→json 文件存储

✧ 功能扩展：控制台程序→可考虑 Tkinter 图形界面(选做)

✧ 算法引入：随机抽题→基于遗忘曲线的智能复习算法(选做)

附件二：设计思维各阶段评价量表

1) 共情阶段评价表(满分 20 分)

评价维度	优秀(5 分)	良好(3~4 分)	一般(1~2 分)	得分
用户洞察深度	准确识别 3 个以上核心痛点，有具体场景描述	识别 2~3 个痛点，描述较清晰	痛点识别模糊或脱离实际	
调研方法多样性	使用≥2 种调研方法(访谈、观察、问卷等)	使用 1 种方法但执行充分	调研方法单一或执行不足	
用户画像质量	画像具体、有代表性，包含人口统计特征和行为特征	画像基本完整，但缺乏细节	画像笼统或脱离项目主题	
需求文档规范性	文档结构清晰，痛点分类合理，有优先级标注	文档完整但逻辑性一般	文档杂乱或信息不全	

2) 定义阶段评价表(满分 15 分)

评价维度	优秀(5 分)	良好(3~4 分)	一般(1~2 分)	得分
问题定义精准性	将痛点准确转化为 1~3 个核心问题，表述清晰	问题定义基本合理，但边界稍模糊	问题定义宽泛或与痛点脱节	
功能清单完整性	功能项完整覆盖核心需求，MVP 定义明确	功能清单基本完整，MVP 定义较清晰	功能清单遗漏重要需求	
优先级排序合理性	功能优先级排序逻辑清晰，符合用户价值	有优先级排序但依据不足	缺乏优先级排序或排序不合理	

3) 构思阶段评价表(满分 20 分)

评价维度	优秀(5 分)	良好(3~4 分)	一般(1~2 分)	得分
方案创造性	提出≥2 种创新性技术方案，有独特见解	提出 1~2 种可行方案，较常规	方案单一或缺乏创新性	
技术可行性	方案充分考虑技术约束和实现成本	方案基本可行，但部分细节待验证	方案存在明显技术障碍	
流程图/架构图质量	图表规范、逻辑清晰，涵盖主要流程	图表基本完整，但细节不足	图表不规范或逻辑混乱	
方案比选分析	多方案对比分析全面，选择理由充分	有简单对比但分析深度不足	缺乏方案对比或分析	

4) 原型阶段评价表(满分 25 分)

评价维度	优秀(5 分)	良好(3~4 分)	一般(1~2 分)	得分
MVP 完成度	MVP 完全符合定义，核心功能全部实现	MVP 基本完成，核心功能缺 1 项	MVP 完成度低于 80%	
代码模块化	函数/类划分合理，高内聚低耦合	模块划分基本合理，但接口设计一般	代码结构混乱，模块化不足	
代码规范性	命名规范、注释完整、符合 PEP8	基本规范，但个别地方待改进	规范性较差，影响可读性	
功能完整性	实现所有计划功能，无明显 Bug	实现主要功能，有少量非关键 Bug	功能实现不全或存在严重 Bug	
工程文档	有完整的 README、API 说明、部署指南	文档基本齐全但较简略	文档缺失或严重不足	

5) 测试与迭代阶段评价表(满分 20 分)

评价维度	优秀(5 分)	良好(3~4 分)	一般(1~2 分)	得分
测试计划完整性	测试场景覆盖全面，有明确的测试用例	测试场景基本覆盖，用例较简单	测试计划粗糙或缺失	
用户反馈收集	系统收集≥5 份有效反馈，分类整理	收集 3~4 份反馈，整理一般	反馈收集不足或未整理	
迭代响应有效性	基于反馈进行≥3 次有效迭代，有明显改进	进行 1~2 次迭代，有改进但不足	迭代响应迟缓或无效	
迭代文档记录	迭代日志完整，记录每次修改的原因和效果	有记录但不够系统	缺乏迭代记录	

总分：100 分

- 优秀：85~100 分
- 良好：70~84 分
- 合格：60~69 分
- 待改进：<60 分

附件三：学生项目迭代日志模板

项目名称：_____ 小组编号：_____ 成员：_____

迭代版本	日期	修改内容	修改原因 (基于什么反馈)	效果验证	下一步计划
V1.0		初始 MVP： 1. 2. 3.	无	功能测试通过	
V1.1		1. 2. 3.	用户反馈： 1. 2.	测试结果：	
V1.2		1. 2. 3.	用户反馈： 1. 2.	测试结果：	
V2.0		1. 2. 3.	新增需求： 1. 2.	测试结果：	
最终版本总结		1. 共经历____次迭代 2. 主要改进点： - - 1. 未解决问题： - -			

注：可根据实际情况增加或减少迭代版本。

附件四：期末综合评价表

学生姓名：_____ 学号：_____ 项目：_____

一、过程评价(40%)

评价项目	权重	得分	评语
1. 共情阶段产出物	5%		
2. 定义阶段产出物	5%		
3. 构思阶段产出物	5%		
4. 原型阶段代码质量	10%		
5. 测试迭代记录	10%		
6. 团队协作与参与度	5%		
过程评价小计	40%		

二、作品评价(60%)

评价维度	权重	评分标准	得分
功能实现	20%	- 核心功能完整(10 分) - 扩展功能创新性(5 分) - 功能稳定性(5 分)	
用户体验	15%	- 界面/交互友好度(8 分) - 提示信息清晰度(4 分) - 错误处理友好性(3 分)	
代码质量	15%	- 模块化程度(5 分) - 代码规范性(5 分) - 注释完整性(5 分)	
创新性	10%	- 解决方案创新(5 分) - 技术应用创新(5 分)	
作品评价小计	60%		

三、总分与等级

项目	分数	折算后得分
过程评价	/40	
作品评价	/60	
最终总分	/100	

等级评定：

✧ A (优秀)：85~100 分

✧ B (良好)：75~84 分

✧ C (中等)：60~74 分

✧ D (待改进)：<60 分

教师综合评语：

附件使用说明

本套附件为论文所述“双项目进阶、五阶段迭代”教学模式提供了核心的操作框架与评价工具。其设计遵循完整性、可操作性与可迁移性原则，教师可根据实际教学情境进行适应性调整。

主要调整建议如下：

课时调整：若总课时不足，可适度压缩课时(如项目一调为 12 课时，项目二调为 20 课时)，或将部分迭代环节转为课外任务。

项目替换：可根据专业方向替换项目主题，但替换时应确保两个项目符合“从简单逻辑到复杂系统”的进阶逻辑，以支撑能力的螺旋式上升。

评价侧重：过程性评价与作品评价的占比可根据课程目标微调，建议过程评价比例不低于 40%，以有效引导与衡量思维过程。

教师实施前建议：

为保障教学效果，建议教师首次实施前：1) 通过工作坊等形式熟悉设计思维流程；2) 亲身完整演练两个项目，预判学生难点；3) 结合实际学情与教学大纲，对本材料进行适当优化。