面向计算思维培养的Python微项目教学研究

李天巨
（兰州市第五十九中学,甘肃兰州 730060）

摘　要：针对传统项目学习在Python教学中存在周期长、门槛高、适配性差等问题，以及学生对Python认知零基础的特点，将微项目教学应用于高中Python课堂，以“校园生活”为核心，创设贴近学生的任务情境，构建主题式微项目，遵循生活化、聚焦化、可视化原则设计教学内容，并通过“情境导入—任务拆解—探究实践—展示评价—拓展延伸”五个环节实施教学。

关键词：微项目；计算思维；Python教学；微项目教学；校园生活
DOI：10.19935/j.cnki.1004-2326.2026.10.039
引用格式：李天巨.面向计算思维培养的Python微项目教学研究[J].实验教学与仪器,2026,43(10):139-141.

随着大数据与人工智能技术的快速发展，计算思维已成为新时代核心素养的重要组成部分，其培育工作在基础教育阶段的重要性日益凸显。对于高中生来说，具备计算思维不仅可以提高解决问题的能力，还能为未来的学习和职业发展打下坚实的基础。编程学习是培养计算思维的一个有效途径。基于此，笔者在高中Python编程教学中，积极探索微项目[1-2]整合学科知识与技能的学习，通过引导学生聚焦并解决实际问题，在实践过程中渗透学科核心素养，提高计算思维。

1　高中Python教学运用微项目的优点

项目式学习虽然可以有效支持学科核心素养的达成，但在其实施过程中存在周期长、时间跨度大、内容多、对师生能力要求高、见效迟等瓶颈问题。近几年，部分一线教师经过探索提出了微项目学习，也就是将项目微小化与中小学常规课堂相结合，实现常态化教学，这为项目式学习实施困难提供了解决思路。在高中Python编程教学中，微项目的优势主要体现在三个方面：一是适配课堂时长，微项目通常可在1~2课时内完成；二是降低学习门槛，微项目更聚焦数量不多的Python核心知识点，尤其适合编程基础薄弱的高中生；三是强化思维训练的针对性，学生在短时间内经历“问题拆解—算法设计—代码实现—调试优化”全过程，精准锻炼计算思维的核心能力。

2　Python微项目设计原则

微项目的设计是实现计算思维培养目标的核心环节，需紧扣“小而精”的特点，同时融入计算思维的四大核心要素——问题分解、模式识别、抽象建模、算法设计，具体遵循以下三项原则。

2.1　任务情境生活化，激活问题拆解意识

高中生对贴近自身生活的场景更具感知力，微项目的任务情境应源于校园生活或社会热点，让学生能快速将陌生的编程任务转化为熟悉的现实问题，进而主动拆解任务。例如，在设计“校园运动会积分统计”微项目时，可引导学生将“统计各班总积分”这一核心任务，拆解为“输入各班参赛项目与得分”“判断得分是否符合规则”“按班级累加积分”“输出积分排名”4个子任务，在拆解过程中培养化繁为简的思维习惯。

2.2　知识载体聚焦化，强化模式识别能力

每个微项目需聚焦1~2个Python核心知识点，避免知识点堆砌导致思维分散。同时，通过设计同类任务变式，让学生在重复练习中识别问题背后的共性模式。以“循环结构”教学为例，可先设计基础微项目“打印1~100以内的偶数”，让学生掌握for循环与条件判断的基本模式；再设计变式项目“打印100~200以内能被3整除的数”，引导学生发现仅需调整循环范围与判断条件，同时保持核心逻辑一致，就可强化模式识别能力。

2.3　成果输出可视化，深化抽象建模与算法设计

抽象建模与算法设计是计算思维的核心，而可视化成果能让学生直观感受抽象逻辑转化为具体结果的过程，增强学习成就感。例如，在“简易校园地图导航”微项目中，可引导学生将“校园建筑位置”抽象为直角坐标系中的坐标点，将“两点间最短路径”抽象为“两点间距离公式”，再用Python的Turtle库绘制校园地图与导航路线——学生在“定义坐标（抽象建模）→编写距离计算代码（算法设计）→绘制导航图（成果可视化）”的过程中，深度理解计算思维的本质。

3 Python微项目教学实施流程

结合高中信息技术课堂特点与微项目的优势，将教学实施流程分为“情境导入—任务拆解—探究实践—展示评价—拓展延伸”5个环节，每个环节均围绕计算思维的培养展开。笔者以计算全班BMI的Python微项目为例展开。

3.1 情境导入：激发问题意识

以真实问题为切入点，通过视频、图片或生活案例创设情境，让学生明确微项目的目的。先展示班主任手动统计30名学生的身高、体重，计算BMI并筛选超标学生，耗时20 min且易算错的案例，然后搭配纸质统计表格图片，提出核心问题：班上有40名同学，手动计算每人的BMI不仅麻烦还容易出错，如何用Python快速、准确地完成全班BMI统计和超标筛选呢？通过真实痛点激发学生的问题意识，为后续任务拆解奠定基础，明确项目意义。

3.2 任务拆解：培养任务分解能力

教师引导学生将微项目的核心任务拆解为若干可操作的子任务，并在此过程中注重示范和引导。此环节重点培养学生化繁为简的问题分解能力。

（1）教师示范拆解核心任务。先向学生明确微项目的核心目标是“输入全班学生数据→计算BMI→筛选超标学生→输出结果”，然后示范拆解第一个子任务：“创建存储学生姓名、身高、体重的容器（列表）”，同时讲解“列表就像我们的点名册一样，能一次性存储很多人的信息”。

（2）学生小组拆解子任务。以4人小组为单位，结合BMI计算逻辑，拆解剩余子任务，教师巡视指导。

（3）全班交流补充，形成最终任务拆解清单。子任务1：设计数据输入方式，获取40名学生的姓名、身高（单位：m）、体重（单位：kg）。子任务2：根据BMI公式（BMI=体重÷身高^2），编写代码计算每位学生的BMI。子任务3：设定BMI超标标准（中国成人标准：BMI≥28，肥胖；24≤BMI<28，超重），筛选出超重和肥胖的学生。子任务4：输出全班学生姓名、身高、体重、BMI，以及超标学生的名单和具体信息。

3.3 探究实践：深化思维训练

此环节是学生自主探究与计算思维落地的核心，教师需扮演引导者而非讲授者，具体分为两步：一是算法设计，学生以小组为单位，针对每个子任务设计算法，并在草稿纸上画出流程图；二是代码实现与调试，学生根据流程图编写Python代码，遇到问题时先小组讨论解决，无法解决再向教师求助。此环节让学生在“设计—实现—调试”中深化抽象建模与算法设计能力。

（1）算法设计。

小组讨论：针对每个子任务设计简单算法，教师用通俗语言引导。

示例引导：以“计算BMI”子任务为例，引导学生明确算法逻辑：先获取某同学的体重和身高，用体重除以身高的平方，最后得到该同学的BMI。

绘制流程图：让学生在草稿纸上画出每个子任务的流程图，教师展示标准简化流程图供参考。

（2）代码实现与调试。

基础代码示范：教师先示范核心代码框架，如创建空列表、简单输入输出，用注释标注关键步骤，方便零基础学生理解。

学生自主完成：小组合作，基于流程图补全代码，针对不同子任务进行重点引导。

常见问题解答：教师在巡视时，针对学生可能遇到的问题进行引导。

问题1：输入身高/体重时误输文字导致报错，引导学生思考“如何确保输入的是数字”，提示可先判断输入类型。

问题2：计算BMI时忘记“身高平方”，引导学生回忆公式，用“height**2”表示平方。

问题3：在筛选超标学生时容易漏判“超重”或“肥胖”情况，提示用“嵌套条件判断”（if-elif-else）覆盖所有情况。

3.4 展示评价：强化思维反思

采用学生展示和师生互评的方式，重点关注思维过程，而不是代码对错。每个小组派代表展示微项目成果，如成绩统计表、导航图，讲解任务及其拆解思路、算法设计逻辑、调试时遇到的问题和解决方法；其他小组与教师从“任务拆解是否合理”“算法是否简洁”“代码是否规范”“计算思维运用是否到位”4个维度进行评价。通过评价，引导学生反思自身的计算思维过程，实现思维的深化。

（1）小组展示：每个小组派1名代表展示代码运行结果，如全班学生BMI数据表、超标学生名单，并进行讲解。

（2）师生互评：其他小组和教师从4个维度进行评价。

任务拆解是否合理，子任务是否覆盖“输入—计算—筛选—输出”全流程。算法是否简洁，BMI计算逻辑、条件判断是否清晰，无冗余步骤。代码是否规范，变量命名是否易懂，是否有必要注释。计算思维运用是否到位，是否用列表抽象存储批量数据，是否能自主调试解决问题。

（3）重点点评：针对零基础学生的亮点给予肯定，如

某小组用列表存储所有学生数据，把具体的姓名、身高转化为可处理的数据对象，体现了抽象建模的计算思维；对共性问题（如条件判断逻辑错误）进行集体讲解。

3.5 拓展延伸：促进思维迁移

在微项目完成后，设计1~2个同类变式任务，让学生将所学的计算思维迁移到新场景中，实现举一反三。

（1）变式任务设计：布置1个同类拓展任务，引导学生迁移计算思维。如何用Python统计全班学生的每日睡眠时间（单位：h），筛选出睡眠时间不足8 h的学生，并计算全班平均睡眠时间。

（2）思维迁移引导：提示学生参考BMI计算的思维模式（输入数据→计算分析→输出结果），将“姓名、身高、体重”替换为“姓名、睡眠时间”，将“BMI计算”替换为“平均睡眠时间计算”，将“BMI超标筛选”替换为“不足8 h筛选”，实现举一反三。

（3）布置课后任务：让学生独立完成拓展任务，下节课分享思路和代码。

4 Python微项目实施需注意的问题

4.1 锚定计算思维目标，避免任务驱动沦为代码操作

在Python微项目教学中，教师常围绕写代码、跑程序设计流程，让学生跟着指令敲代码。学生虽完成了任务，却不懂为何用列表存数据、为何要拆成循环步骤，其本质是练了代码操作却没练计算思维。实施时需先明确每个项目要培养的计算思维维度，如问题分解、抽象建模、算法设计，再围绕维度设计任务。

4.2 用门槛实践承载计算思维，避免Python语法成障碍

Python的变量类型、循环结构、条件判断等基础语法，对于零基础学生来说是首要难点。若先花大量时间讲语法，再做项目，学生易因怕写错代码而失去兴趣，更难聚焦计算思维。实施时需谨记语法应服务于思维，用简单语法实现核心思维训练。同时，提供代码模板降低门槛，比如给出带注释的框架，让学生聚焦补全算法而非从零写代码，在低压力实践中练思维。

4.3 防止问题分解流于形式，按逻辑拆任务、明步骤

计算思维核心是化繁为简，但学生在做Python微项目时，任务拆解混乱：如“统计班级BMI”拆成“输入数据”和“输出结果”，漏了“计算BMI”“筛选超标学生”；或拆成“处理数据”，导致后续代码无法推进。需教会学生运用结构化拆解方法：先定总目标，再拆成可操作、无重叠、全覆盖的子任务，每个子任务明确做什么、用Python怎么实现。

4.4 引导学生自主调试，培养思维闭环

学生在写Python代码时，常因语法错（如少写冒号、变量名写错）或逻辑错（如BMI计算漏平方、筛选条件写反）导致程序报错，很多教师会直接指出错误，学生虽改对了代码，却没学会如何发现错、分析错、解决错，计算思维的调试优化环节被跳过。实施时需引导学生自主调试，让学生通过“看结果→找异常→回溯逻辑→修正代码”的过程，形成从设计到实现再到调试的思维闭环。

4.5 强化思维迁移复用，避免项目教学止步于任务完成

Python微项目的最终教学目标是引导学生将计算思维迁移至同类问题的解决场景，但当前不少教学实践仅停留在“完成当前项目”的浅层次：学生学完“统计BMI”案例后，面对“统计学生睡眠时间、筛选睡眠不足8 h的学生”这类新任务，依然无法独立完成任务拆解与代码设计。其本质原因是学生并未掌握数据处理的通用思维框架。在项目实施的收尾阶段，教师需先提炼Python数据处理类项目的通用思维范式，再通过设计的变式任务引导学生完成迁移训练，让学生复用同一套思维框架解决全新问题，真正实现举一反三，将计算思维从孤立的项目技能转化为可迁移的通用问题解决能力。

5 结语

笔者在高中Python教学实践中发现，微项目绝非复杂任务的简单化，而是以小切口撬动大思维的精准教学载体。它通过构建贴近学生生活的真实情境，让编程学习跳出纯粹的语法训练，回归用技术解决实际问题的实用本质；同时以聚焦化的知识模块为核心，有效降低零基础学生的认知负荷，避免因内容繁杂产生畏难情绪。更重要的是，微项目带来的可视化成果能直观强化学生的学习成就感，推动其主动探索。最终，学生在完成微项目的过程中，不仅能扎实掌握Python编程技能，更能逐步形成计算思维，为适应大数据、人工智能时代的技术需求奠定坚实基础。

作者简介：李天巨（1981—），男，高级教师，研究方向：高中信息技术教学，E-mail：397816068@qq.com。

参考文献

[1] 殷凤杰.高中信息技术课堂微项目教学实践研究：以“算法与程序实现”为例[J].中小学信息技术教育，2024(C1)：110-112.

[2] 林温阳.高中信息技术Python编程教学微项目学习研究[J].福建教育学院学报，2022，23(08)：58-60，66.