认知水平与计算思维间的隐含关系挖掘——基于编程行为表征视角

原创 姜强　王利思等 [现代远程教育研究](javascript:void(0);) 1周前

**作者简介**：姜强，博士，教授，博士生导师，东北师范大学信息科学与技术学院（吉林长春　130117）；王利思，硕士研究生，东北师范大学信息科学与技术学院（吉林长春　130117）；赵蔚，博士，教授，博士生导师，东北师范大学信息科学与技术学院（吉林长春　130117）；潘星竹，二级教师，大连市甘井子区奥林小学（辽宁大连　116031）。

**基金项目：**教育部人文社会科学研究青年基金项目“大数据时代在线学习者情感挖掘与干预研究”（16YJC880046） ；东北师范大学教师教育研究基金项目“职前教师信息技术应用能力培养与发展研究”（JSJY20180301）。

引用：姜强,王利思,赵蔚,潘星竹(2020).认知水平与计算思维间的隐含关系挖掘——基于编程行为表征视角[J].现代远程教育研究,32(2):94-103.

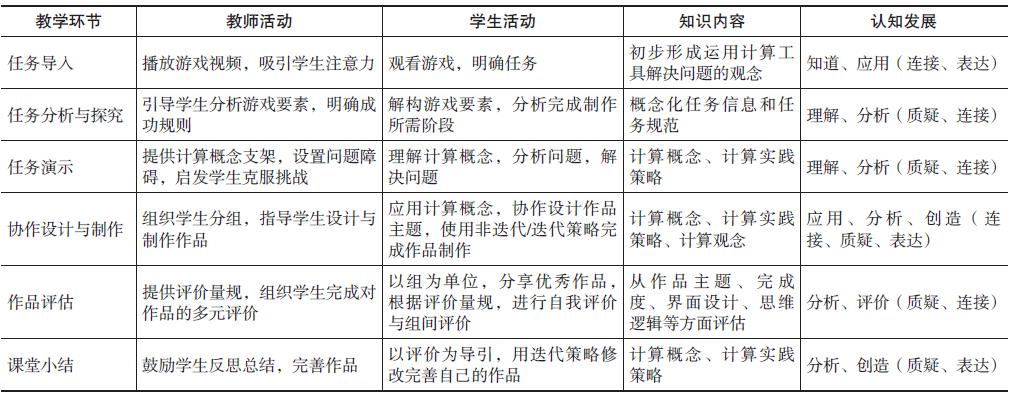
**摘要：**随着人工智能时代的来临，编程教育日益受到重视。学生在编程过程中的行为，实质上反映了其对利用计算工具解决问题的认知水平，也映射出其计算思维的发展过程，挖掘三者间的隐含关系有助于通过优化编程任务设计改善对学生计算思维的培养。基于认知目标分类和计算思维三维框架构建映射关系，以Scratch可视化编程任务作为研究案例，采用编程操作视频分析与学生访谈相结合的方法，从编程行为表征的视角对认知水平与计算思维间的隐含关系进行挖掘后发现：一方面，认知水平与编程行为之间存在密切关系，知道与理解是编程行为的基础性认知，应用与分析是编程行为的核心，评价推动学生对编程行为的反思，而创造可激发编程行为产生新的作品。另一方面，在编程活动中，学生的认知水平与计算思维的发展彼此关联且相互促进，编程行为中的低阶认知目标的达成是形成计算观念的基础，编程实践引发的高阶思维认知需求可促进学生对计算思维的领悟。因此，应当遵循认知水平与计算思维的发展规律，有针对性地将面向各类认知目标的编程任务嵌入到教学设计中，方可有效实现利用编程教育对学生计算思维的培养。

**关键词：**编程教育；计算思维；认知水平；行为表征；Scratch

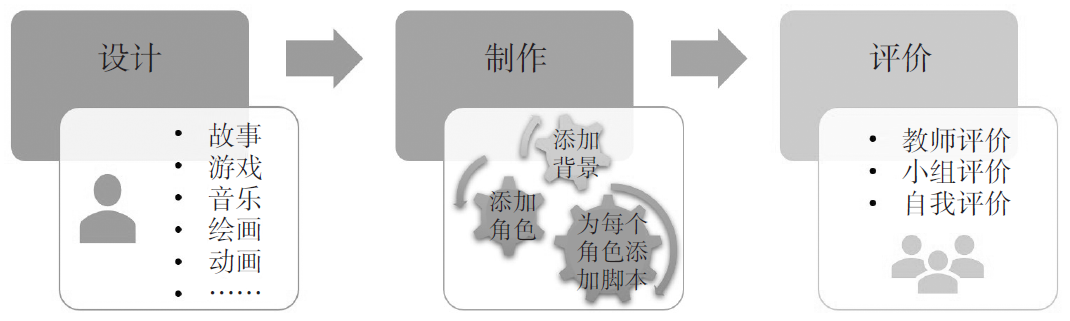
**一、引言**随着人工智能时代的来临，编程教育的课程化成为一种趋势。国务院与教育部分别印发的《新一代人工智能发展规划》和《2019年教育信息化和网络安全工作要点》明确提出要在中小学阶段设置人工智能相关课程、建设人工智能学科、逐步推广编程教育（国务院，2017；教育部办公厅，2019）。编程教育与新时代下所倡导的STEAM教育、创客教育理念不谋而合，有利于学生从科技的消费者变为科技的创造者，进而获得技能转移，使其在具体的编程语言环境中通过运用较复杂的思维技能来培养高阶思维和计算思维，以提高问题解决能力和创新能力。目前，中小学生主要利用Scratch完成编程任务，其作为一种可视化编程工具，以“所见即所得”的编程思想为原则，可形象地表征个体的思维过程，促使个体主动地进行知识建构，形成自己的知识网络。然而，在编程环境中，当学生在通过创建程序解决问题时，其注意力已从操作层面转移到理解编程的认知层面，其利用计算机语言进行计算式思考的过程，实质上反映了其利用计算工具解决问题的一系列认知过程。仅依靠对最终程序或作品的分析往往不能全面解释学生对编程问题的理解及其思维过程，要深度理解“隐形”于计算机操作知识中的计算思维，还需关注学生实践过程中的行为表现和认知水平，以剖析学生的概念应用能力、拆组项目能力、纠错分析能力以及问题解决能力。因此，本文基于修订版布鲁姆认知目标分类体系和美国麻省理工学院（MIT）提出的计算思维三维框架，聚焦于对学生编程行为过程的理解，判断编程实践引发的认知需求，挖掘学生认知水平与计算思维间的关系，旨在揭示编程教育对思维发展的普适性价值，为针对性地设计适合学生思维特点的可视化编程任务，进而从认知角度根本改善计算思维的培养方式提供启示。**二、相关概念与理论框架**作为一种复杂的心智活动，计算思维（Computational Thinking）涉及问题制定、解决方案形成、方案执行和评估的迭代过程，其以抽象和自动化为特征，以计算资源为手段，以问题解决为目标，是新时代人才适应数字化生存的必要技能。Wing（2006；2010）指出计算思维是“21世纪的新素养”，是与听、说、读、写同等重要的基本技能之一，是一套利用计算机科学的基本概念解决问题、设计系统和理解人类行为的完整思维工具。美国计算机科学教师协会（Computer Science Teachers Association，CSTA）提出了计算思维用于问题解决的操作性定义，即以机器协助方式形式化问题，以逻辑方法处理数据、抽象化表示数据、算法化自动解决方案，以有效方式解决问题并将知识和技能迁移至其他问题情境（Chen et al.，2017）。陈国良等（2013）指出计算思维是一种综合性思维，包含数学思维、工程思维和科学思维。李廉（2012）认为计算思维与实证思维、逻辑思维一样，是人类科学思维的重要组成部分。战德臣等（2013）指出计算思维是一种思维方式，学生不应仅着眼于“知识”（即事实）的学习，而应更多地训练“思维”。此外，吴忭等（2019）基于量化民族志的分析方法对计算思维能力进行了评估，张进宝（2019）从计算学科、普适化和认知发展三个视角对计算思维概念进行了系统化分析。目前，各国均十分关注和重视对学生计算思维的培养。英国全面改革中小学计算机课程教学大纲，从侧重于软件应用的信息通信技术教育转向以计算思维培养为重点的计算机科学教育（Department for Education，2014）。美国《K-12计算机科学框架》指出，儿童需要在获得一定的计算机操作技能基础上，构建扎实的计算思维（ACM et al.，2016）。新加坡启动了“智慧国家计划”，注重开发学生的计算思维能力（Seow et al.，2017）。意大利启动Programma il Futuro项目，以Code.org开发的教学材料为基础，强化培养学生的计算思维（Nardelli et al.，2015）。“澳大利亚课程：数字技术”（Australian Curriculum: Digital Technologies）强调让学生开发和运用计算思维技能，在创建、交流和共享信息及管理项目时，使用编程技术和数字化方式来解决特定的问题或需求（ACARA，2012）。芬兰将编程作为小学教育的综合要素之一，大力培养学生的计算思维（Heintz et al.，2016）。爱沙尼亚启动ProgeTiger项目，旨在把编程和机器人技术引入教育领域，将计算思维与人工智能活动相结合，以应对技术世界的挑战（HITSA， 2015）。我国新修订的《普通高中信息技术课程标准》强调计算思维是信息技术学科的核心素养之一，要通过学习计算机科学相关知识与技能，培养学生的逻辑思考、系统化思考能力（任友群等， 2016）。美国教育心理学家Mayer等（1986）研究发现，探索程序设计与思维能力关系的最有效方式，就是把注意力集中在程序设计时的认知思维能力上。多名教育研究专家对布鲁姆认知领域教育目标分类进行了重新修订，Krathwohl（2002）对其修订版进行了概述和解读：新的分类采用“知识”与“认知过程”二维框架，将认知过程修改为“知道、理解、应用、分析、评价和创造”，6个层级的认知过程由低阶思维向高阶思维逐步过渡，各层级相互联系、彼此依存。美国MIT媒体实验室基于Scratch编程活动界定了计算思维三维框架，包括计算概念（设计者在编程中所使用的概念）、计算实践（设计者在编程时所进行的问题解决实践）和计算观念（设计者形成的有关自身和周边世界的观念）（Brennan et al.，2012）。该框架符合Scratch编程特点，为有效表征编程活动中隐含的计算思维提供了理论依据。因此，根据表1所示的认知目标分类与计算思维三维框架（Falloon，2016）间的映射关系，可通过可视化编程行为来论证认知水平和计算思维之间的隐含关系。**表1    研究的理论框架（认知目标分类与计算思维三维框架间的映射关系）**



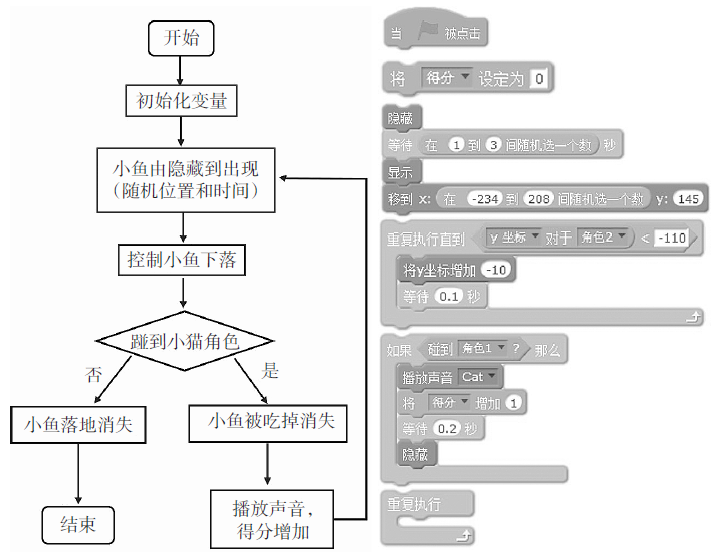
其中，计算概念包括序列（特定任务或活动中由计算机执行产生的一系列步骤或指令）、循环（多次运行相同的序列）、并行（发生在同一时间的不同事件）、事件（可导致另一事件发生，如触发器）、条件（可基于特定条件作出决策，支持多种结果的表达）、操作（支持数学、字符串和逻辑表达式的运算）和数据（可被存储、检索和更新的值、变量和链表）；计算实践包括试验与迭代（开发一部分程序进行尝试，然后根据经验和新想法进一步开发，每一次迭代都可为得到反馈和新想法提供机会）、测试与调试（确保运行正常，当问题产生时能迅速发现并解决）、重用与翻创（在现有项目或想法的基础上进行创作）及抽象与模块化（探索部分与整体间的联系）；计算观念包括表达（把计算视为设计、创造和自我表达的媒介）、连接（意识到一起创造价值和为他人创造价值的力量）和质疑（以计算理解世界，敢于对世界和技术提出疑问）（Brennan et al.，2012）。**三、研究设计**1.研究对象以“信息技术”课程为依托，选取大连市某小学六年级44名学生参与该课程。每节课时长为35分钟，学生平均操作时长约为13分钟，课堂上采用录屏软件收集学生使用Scratch进行可视化编程的操作行为数据。去除机器故障、学生缺席等无效数据后，共获得41名学生的操作数据，总时长为9时7分27秒。2.教学案例设计与创作流程研究以“小猫捉小鱼”游戏制作为任务索引，捕捉了游戏中碰撞、得分等典型元素，采用任务驱动、协作探究教学策略，并辅以讲授法完成教学过程。教学设计注重培养学生的自主思考能力，鼓励学生自主探索，并引导学生将复杂问题拆分、解构成简单问题，以提高学生分析解决问题的能力。同时，整个教学任务的设计围绕着计算思维元素展开，教学方案包括教学环节、教师活动、学生活动、知识内容与认知发展等如表2所示的5个部分。**表2    “小猫捉小鱼”教学案例设计**



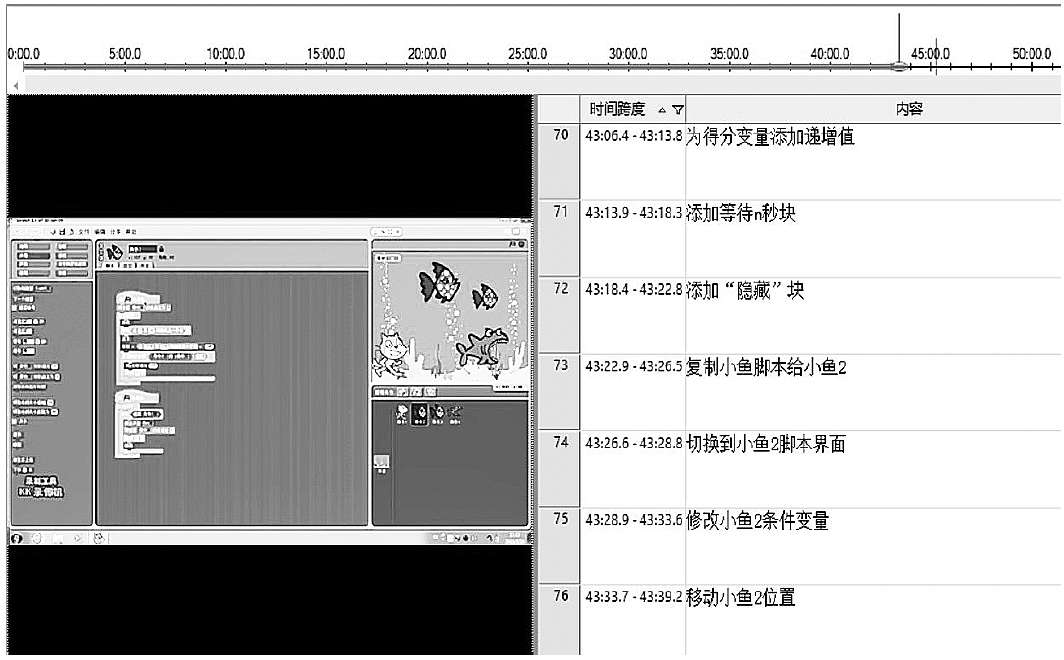
其中，教师活动为创设任务情境、提供概念支架、提问、分组、组织多元评价等；学生活动主要为观察、思考、反馈、设计与制作、评价等；知识内容为游戏要素分解、任务成功标准、Scratch模块功能、计算概念与实践策略等；认知发展主要根据认知目标与计算思维的映射框架，对学生在完成编程任务过程中的思维发展加以评估。案例创作流程分为设计、制作和评价三个阶段，如图1所示。



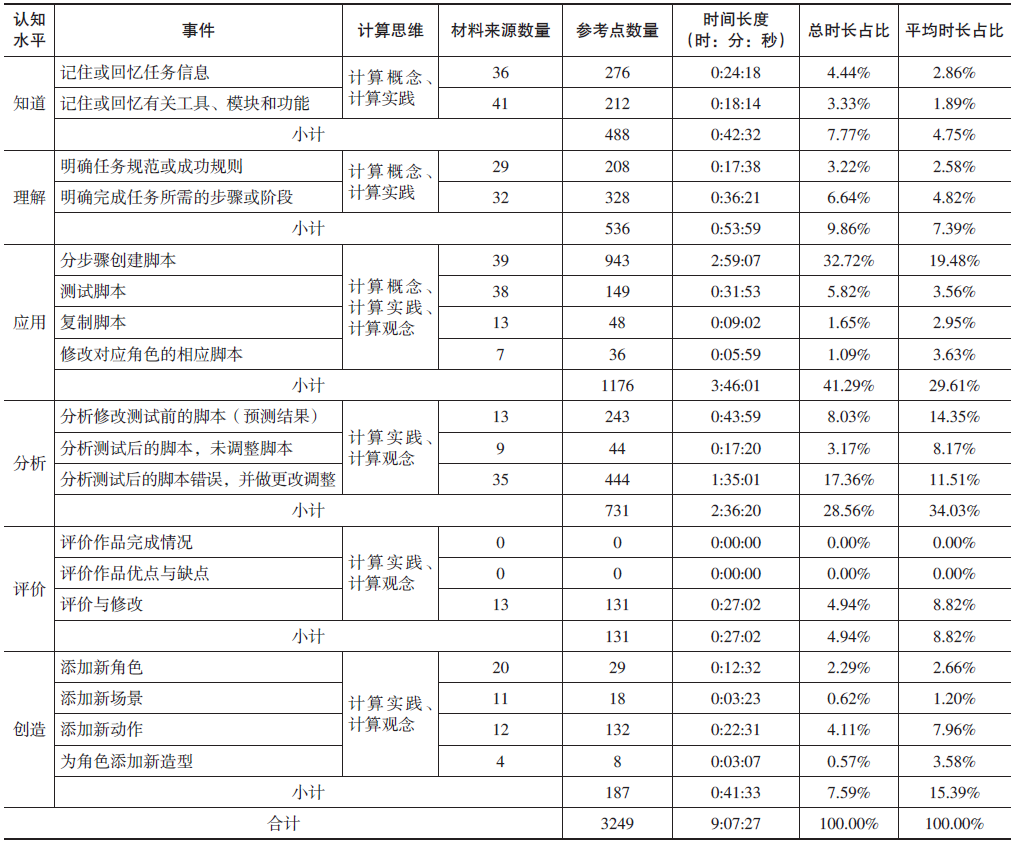
**图1　案例创作总体流程**在设计阶段，充分把握Scratch可视化编程工具特点，选择吸引学生注意力的任务题材，激发学生想象力，引导其创造属于自己的艺术作品。制作阶段是将设计蓝图转变为实物的过程，每一个作品的制作都离不开对背景、角色和脚本的选择和添加，以此思路为学生分析制作复杂作品提供参考。评价阶段是学生对自己作品的反思和再认知，充分利用多元主体评价的优势，帮助学生迭代修改、完善作品，实现作品效果的最优化。其中，作品案例（游戏场景中的小鱼角色）的算法流程及脚本如图2所示。



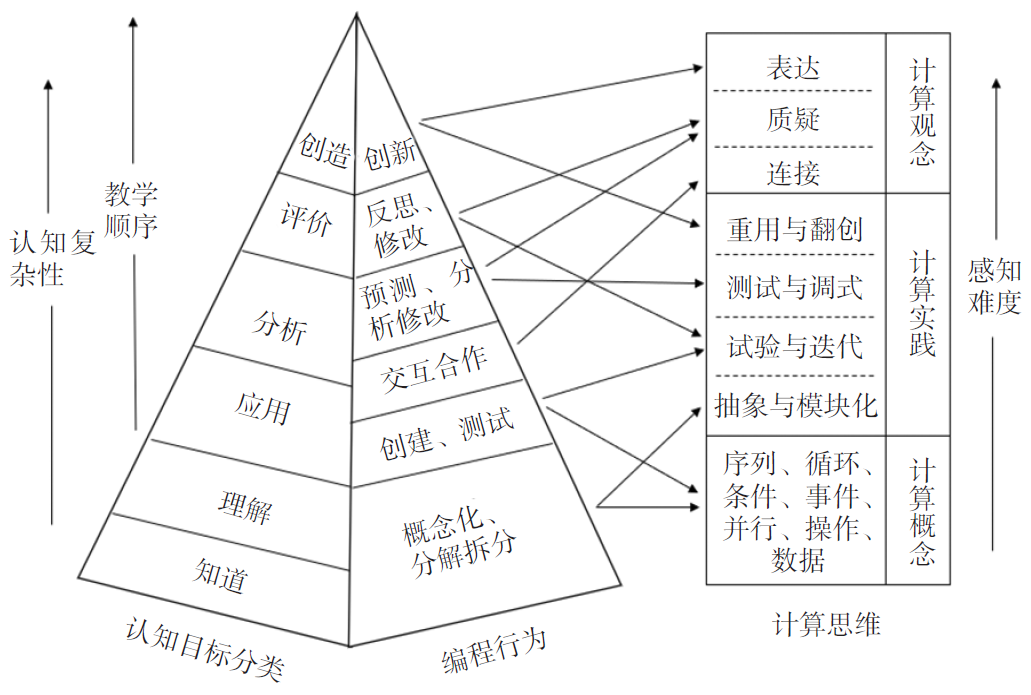
**图2 　“游戏场景中的小鱼角色”算法流程及脚本**3.访谈设计课程结束后，以受访者自愿为原则，随机抽取若干名学生进行半结构化访谈和分析，进而深入挖掘学生在编程过程中表现出的认知水平与计算思维。访谈提纲主要包括：简述作品创作思路（计算实践）、创作中遇到的问题及解决办法（计算观念）、完成编程任务的重难点（计算实践）、作品迭代修改后的变化（计算实践）、对调试结果的预测分析（计算观念）以及对创作完成后的评价及内心体验。**四、数据编码与结果分析**1.数据编码与统计使用质性分析软件NVivo 11对所收集的样本数据进行处理。将收集到的编程操作视频导入NVivo中，对每个视频进行如图3所示的开放性编码。将视频数据转化为文本数据后，根据时间轴工具划分的时间节点，使编程行为形成在时间维度上连续的操作序列。



**图3　视频编码**根据认知目标分类与编程活动的映射关系，对开放型编码后的数据进行再次编码（即主轴编码），以探寻编程行为与认知水平间的关系。为确保样本数据编码的可靠性，数据编码工作由两名研究人员共同完成。识别出的事件共3431项，经讨论后仍存在分歧意见的182项事件被丢弃，最终采用的编码事件为3249项。为进一步分析，再次对二次编码后的事件节点进行了整理，并对各类事件的材料来源数量、参考点数量、时间长度、总时长占比（单类事件的编码时间/所有事件的编码总时间）和平均时长占比（平均时长占比=事件的平均时间/所有事件的平均时间之和，事件的平均时间=事件时长之和/人数）等进行了统计，结果如表3所示。需要特别说明的是，为分析表征学生计算思维的编程行为事件及其对应的认知层次，对各类事件的时间长度、总时长占比进行了统计。此外，由于各类事件的材料来源数量不同，为区分学生个体在各类事件上所用时间的差异，对各类事件的平均时长占比进行了统计。2.结果分析与讨论（1）编程行为与认知水平间的关系第一，知道与理解是编程行为的基础性认知。由表3中事件的材料来源数量可看出，每个学生的编程活动都含有“记住或回忆有关工具、模块和功能”事件，其具体涉及的活动包括打开软件、查看旋转按钮的功能、查看各个模块信息等。学生通过回忆任务信息与原有认知建立连接，概念化所需工具和任务标准，按照任务要求和标准完成增删场景、角色、造型等活动，初步形成了运用计算工具解决问题的观念。在教师指导下理解相关计算概念，共同完成对复杂任务的解构，明确完成任务所需阶段，是完成计算实践的基础。因此，概念化任务和工具、解构模块化任务阶段往往发生在学生实际编程活动之前，多与教师共同完成，因而操作行为只能体现部分认知过程。“知道”与“理解”相关事件的平均时长占比之和为12.14%，且与学生和教师共同完成概念化任务的时长大致相当，因而学生总体上需要花费约五分之一的时间来完成编程前的准备工作。此阶段中，与“知道”和“理解”这两种低阶认知目标相关的行为事件较多，主要涉及概念化成功标准、解构复杂任务、了解相关计算概念、思考如何利用计算工具解决问题等。在教学过程中，应注重引导学生理解任务要求和标准，帮助其解构任务，以思维导图的形式规划解决方案，选择恰当的工具和策略解决复杂问题。**表3    编程行为表征的统计结果**



第二，应用与分析是编程行为的核心。在Scratch编程活动中，与“应用”和“分析”相关的行为事件最多，超过85%的学生会依次为每个角色分步骤创建动作脚本（39人）和测试脚本（38人），并根据测试结果修改脚本（35人）。据观察，动作脚本中学生使用频率较高的相关计算概念有触发器、坐标值、随机数、变量、重复执行等，而关系运算符和判断条件只有少量学生使用。参考点统计结果显示，平均每个学生测试脚本4次，个别学生的测试脚本事件次数达10次，且测试脚本后，学生往往会继续添加脚本或在分析脚本错误后对其作出修改。分步骤创建动作脚本与调试脚本是编程工作的核心。学生在创建脚本时，会长时间集中注意力，将自己的逻辑思维以可视化的方式呈现出来，这是隐性思维的外显化；学生在测试脚本之后，将测试结果与设想结果进行对比，分析差异、识别脚本漏洞或错误并对其进行修改校正，这反映了编程中“应用”与“分析”的连续思维过程。在小组合作中，学生利用试验与迭代、测试与调试的计算实践策略来识别故障原因，反复纠错并不断得到新的反馈，进而完善问题解决方案，在此过程中学生的毅力和协作能力得到了充分锻炼。大约1/3的学生（13人）在测试脚本之前，就对脚本的执行结果进行预测，以质疑视角分析程序，识别和纠正可能的错误，其中多数通过调换模块顺序或修改变量数值来完善已编写好的程序，直到脚本逻辑符合自己的理想状态才进行测试。这一过程是学生预测性思维的表现。预测性思维作为一种复杂的高阶思维，在小学高年级中只有少数学生具备。教师可以适当引导学生对程序的执行结果进行预测，将Scratch编程作为发展学生预测性思维的有效载体。第三，评价推动学生对编程行为的自我反思及作品优化。编程活动中的评价性行为分为两种情况。一种是面向操作，与修改相关联，可以根据操作行为进行表征的评价性行为。有少部分学生对自己的作品细节作出评价并加以修改完善，如一名同学经过测试发现作品中的小鱼下降速度太快，便添加了“等待n秒”模块，让小鱼下落的速度符合自己的标准。另一种是未面向操作，根据一定的评价量规，从多方面对自己的编程结果作出评判的行为。针对此种评价性行为研究采用访谈加以补充。评价与学生所理解的成功标准相关。由于每个学生的主观审美和感受不同，当学生发现结果与预期不一致时，会努力地识别错误，其可能会更改程序，也可能不会更改，因而单纯依靠操作行为很难体现学生对编程结果的最终评价。在访谈中发现，许多学生对自己的作品有一定的评价，但对于作品的不足并没有作出相应的修改与完善，多数同学表示这是因为课堂时间有限，只有个别同学会利用课余时间进行修改。评价与修改程序需要学生运用迭代策略不断对其进行完善，因而教师需要培养学生的恒心和毅力，并从量与质两个维度制定合适的编程评价量规，鼓励学生通过记录日志或心得的方式来进行自评和组评，充分发挥多元评价主体的作用。第四，创造激发编程行为产生新产品。在创造性行为方面，有48.8%的学生（20人）为编程游戏添加了新角色，如螃蟹、老鼠等，甚至有学生完全基于个人想法，重新编排了游戏情节，如猴子摘香蕉、小猫捉老鼠等；超过1/4的学生为游戏设计了新场景（11人）、添加了新动作（12人），如让小鱼横向游动，更换新触发器等；还有个别同学为角色添加了新的造型以实现角色在不同造型之间的切换。创造性思维是认知目标的最高层次，学生在编程的各个环节，均可以通过创造性思维设计新事物、做出新产品。Scratch编程不仅满足了学生的编程需求，还提供了角色造型库和绘画功能，拓展了学生的想象空间，对于培养学生的创造性思维卓有成效。（2）编程行为与计算思维间的关系从表3中的总时长占比结果来看，学生在Scratch编程过程中，“应用”（41.29%）与“分析”（28.56%）相关事件约占总编码时长的70%，其中分步骤创建脚本（32.72%）、分析修改测试后的脚本（17.36%）花费的时间最多。结合对总时长占比和材料来源数量的分析，可以看出，多数学生达到低阶思维层次，学生整体上在低阶思维相关事件上所花时间较多，总时长占比约为59%（其中，“知道”占7.77%，“理解”占9.86%，“应用”占41.29%），而达到复杂高阶思维层次的人数较少。通过平均时长占比数据可以看出，达到高阶思维层次的学生在高阶思维相关事件上的平均时长占比之和约为58%（其中，“分析”占34.03%，“评价”占8.82%,“创造”占15.39%），高于低阶思维相关事件的平均时长占比之和，这表明高阶思维相关事件需要学生花费更多的时间来进行思考和操作。对编程模块的使用情况和操作行为进行统计后发现：对于小学生而言，在计算概念层次上，数据是基础，被普遍使用，而操作（关系运算和逻辑运算）是存在学习困难的概念，只有少数学生使用；序列、条件和循环作为程序设计的三大逻辑结构，学生使用最多的为序列块，条件块多使用单一条件，循环块普遍使用较易掌握的永久重复执行，而相对复杂的条件循环块则使用甚少；学生多使用串行执行结构解决问题，而对并行执行结构的运用还存在较大困难。在计算实践层次上，试验与迭代、测试与调试是学生应用与分析思维中常用的实践策略，但其花费时间较长；对于编程任务整体算法的设计与分解（抽象与模块化）需要教师进行详细的引导和规划；对概念块的使用还不能有效做到举一反三，仍停留在理解与应用等较低层面上，计算实践的整体水平偏低，多为低阶认知目标水平的实践。在计算观念层次上，质疑观念最为突出，而利用计算工具创造性表达自我的观念较薄弱，分享“连接”观念对于概念化任务和错误、调试与迭代的效率提高有显著作用。此外，访谈结果表明，大部分学生能简要描述创作步骤，甚至有学生对全部操作细节记忆深刻。同时，个别学生表示在编程前，会主动画流程图以帮助自己厘清思路，进而更好地完成编程。对于创作中遇到的问题，学生表示会和同学交流沟通、协同解决，且感觉到向他人寻求帮助受益匪浅。至于编程中的重难点，学生则认为自己有大体的制作思路，可是编写程序时总是出错，很难快速达到预想效果，因而认真分析程序并修改错误特别重要。部分学生表示会利用课余时间对自己的作品进行修改完善，如有同学在已完成的作品“猫捉老鼠”游戏中，添加怪兽并赋予怪兽武器与猫制衡。只有个别学生会在调试脚本之前分析预测执行结果，大部分学生表示会直接测试程序，然后再分析修改脚本，觉得自己达不到“预测”的高水平。学生在作品完成后，会反思“有没有做完”“做得如何”“优点与缺点是什么”，且对学习Scratch表现出浓厚兴趣，认为其可以培养自己的逻辑思考能力、设计能力与创造力。（3）编程行为、认知水平与计算思维之间的关系一是编程行为表征出的认知水平与计算思维相互关联映射。基于编程活动行为表征的认知水平与计算思维可以相互映射、彼此促进，其关系模型如图4所示。总体来讲，随着编程行为引发的认知需求复杂性不断加深，学生感知的计算思维难度逐步增强，通过接触计算概念、不断积累计算实践经验后，逐步形成计算观念。计算概念是程序创作的基础，学生在编程任务中对计算概念的应用往往建立在知道和理解基础上，而创建与修改概念块也会反向促进对概念的理解。在计算实践中，抽象与模块化实践引导学生将真实世界的复杂问题抽象成Scratch舞台上的场景、角色和脚本，通过寻找线索并解构任务来达成问题的最终解决，可有效锻炼学生的分析解构能力；试验与迭代、测试与调试实践不仅能有效锻炼学生的查错、纠错、分析和解决问题的能力，还可促进学生对自己作品的评价、反思和再完善；重用与翻创实践对于培养学生的创新意识和能力，进而创造出优秀作品有显著作用。计算观念与高阶认知目标具有内在一致性，学生通过批判性地使用计算工具、创造性地表达自我、与他人一起创造价值来发展高阶思维。



**图4　关系模型**  
二是编程行为中低阶认知目标的达成是形成计算思维的基础。在知道、理解与计算概念、计算实践间的关系方面：以“小猫捉小鱼”游戏制作为例，其所涉及的计算概念有序列、事件、数据、循环、条件、并行和操作，学生必须理解这些概念及其操作，与此相对应的计算实践能力包括抽象与模块化，其具体体现为概念化工具、任务信息与成功标准。在教学过程中，教师需要及时提供概念支架，引导学生理解并使用复杂的计算概念；同时，还应着重提高学生的问题识别、抽象化、分解等计算实践能力，帮助学生形成利用计算工具解决问题的观念。在应用与计算概念、计算实践、计算观念间的关系方面：应用计算概念，设计问题解决方案，分步骤创建动作脚本、测试脚本，将测试结果与成功标准对照，概念化脚本中的漏洞或错误并调试解决，是编程的基本实践过程。在此过程中学生多需要与同伴沟通交流，这种分享信息和策略的行为与“连接”的计算观念相对应，其包含师生交互及生生交互。复杂问题或项目往往需要通过共同协作来解决，因而合作与共享是完成编程任务的必备品质。教师需要通过设置有层次性的、与学生生活经验相关的探究活动，让学生感受到合作解决问题的成就和喜悦。三是编程实践引发的高阶思维认知需求可促进学生对计算思维的领悟。在分析与计算实践、计算观念间的关系方面：分析与调试程序往往是一个多次迭代、螺旋上升的过程，部分学生会在原有作品的基础上进行翻创，每一次迭代修改，都为得到新的反馈提供机会，可有效激发学生的创造力。教师要注重培养学生的耐心和坚毅品质，引导其主动识别和纠正可能的错误，促进其测试与调试、迭代、翻创等计算实践能力的提升。此外，预测性思维与计算观念中的质疑有关，质疑观念可引发学生对方案的反思，从而主动识别和纠正可能的错误。这在一定程度上会促进其对任务的分析进而开发出合理的解决方案。同时，预测性思维也是一把双刃剑，学生在预测错误时需花费较多时间，这会减慢整体编程节奏，因而需要将鼓励预测与鼓励测试相平衡。在评价、创造与计算实践、计算观念间的关系方面：评价创作工具和作品蕴含着质疑的计算观念，这意味着学生敢于对使用的技术提出疑问。访谈结果表明学生有能力评估自己的作品，正确对Scratch编程的利弊进行一定评价，但对中小学生来说，对编程作品的评价需要教师建立一定的评价量规作为支架，而评价后对作品的迭代完善也需要教师的鼓励与支持。创造思维常与设计、想象、批判相关联，学生通过设计新场景、新故事并赋予作品新的意义来不断翻创原有作品，这本质上是在利用计算工具表达自己对世界的看法。教师需要帮助学生恰当地利用技术手段个性化、创造性地表达自我。编程作为一种面向问题解决的计算实践活动，可以长时间凝聚学生注意力，引导学生以程序设计的方式表达想法，而可视化编程可将抽象问题形象化表达，并按照一定逻辑顺序生成解决方案，这为中小学阶段学生形成和表达质疑观念、发展创造性思维提供了有效的手段。**五、结论**编程作为培养高阶思维与计算思维的有效载体，对促进学生的认知发展、协作和自主学习有明显效力。本文通过分析学生可视化编程行为中蕴含的认知水平及其映射的计算思维，实现了内隐计算思维的外显化，并构建了认知水平、编程行为与计算思维间的关系模型。研究发现，在可视化编程活动中，学生认知与计算思维的发展是相互映射的，且其发展方向具有一致性，即在知道、理解与应用等低阶认知层次上主要体现为计算概念思维，在理解、应用、分析、评价与创造等认知层次上主要体现为计算实践思维，在分析、评价与创造等高阶认知层次上主要体现为计算观念思维。研究结果表明只有将有意义的编程任务嵌入到协作学习、问题解决式学习或项目式学习的教学设计中，并制定合适的评价量规作为导引，多给学生提供锻炼高阶思维的编程实践活动，遵循认知水平和计算思维发展规律设计任务，才能实现利用编程进行计算思维培养的价值最大化。编程亦是“人工智能+”教育的重要方式，学生在创建、修改、测试和分享作品的过程中，通过不断调整和反思，既获得了知识技能，又提高了计算思维能力，这符合个体的认知发展规律，有助于对学生信息技术核心素养的培养。进一步的研究可考虑：（1）多层次的数据获取与多任务类型的设计。一方面，面向小学、初中、高中等不同学段群体开展研究；另一方面，除游戏案例外，设计多类型任务（如故事、音乐、动画等），增强研究的普适性。（2）除自身操作性学习行为外，有必要进一步系统研究小组合作、教师评价等因素对学生认知水平、计算思维的深层次影响。（3）针对计算思维迁移的研究。本文是在编程学习平台下开展的研究，而将研究情景迁移到日常的实际问题解决中，可进一步挖掘认知水平与计算思维间的关系。

**参考文献：**

[1]陈国良,董荣胜(2013).计算思维的表述体系[J].中国大学教学,(12):22-26.

[2]国务院(2017).国务院关于印发新一代人工智能发展规划的通知[EB/OL].[2019-01-10].http://www.gov.cn/zhengce/content/2017-07/20/content\_5211996.htm.

[3]教育部办公厅(2019).教育部办公厅关于印发《2019年教育信息化和网络安全工作要点》的通知[EB/OL].[2019-03-15].http://www.moe.gov.cn/srcsite/A16/s3342/201903/t20190312\_373147.html.

[4]李廉(2012).计算思维——概念与挑战[J].中国大学教学,(1):7-12.

[5]任友群,黄荣怀(2016).高中信息技术课程标准修订说明[J].中国电化教育,(12):1-3.

[6]吴忭,王戈(2019).协作编程中的计算思维发展轨迹研究——基于量化民族志的分析方法[J].现代远程教育研究,(2):76-84,94.

[7]战德臣,聂兰顺(2013).计算思维与大学计算机课程改革的基本思路[J].中国大学教学,(2):56-60.

[8]张进宝(2019).计算思维教育: 概念演变与面临的挑战[J].现代远程教育研究,31(6):89-101.

[9]ACARA (2012). Draft Shape of the Australian Curriculum: Technologies[EB/OL]. [2019-06-09].  https://docs.acara.edu.au/resources/\_of\_the\_Australian\_Curriculum\_Technologies\_paper\_March\_2012.pdf.

[10]ACM, Code.org & CSTA et al. (2016). K-12 Computer Science Framework [EB/OL]. [2019-06-09]. https://k12cs.org/.

[11]Brennan, K., & Resnick, M. (2012). New Frameworks for Studying and Assessing the Development of Computational Thinking[C]// Proceedings of the 2012 Annual Meeting of the American Educational Research Association. Vancouver, Canada: AERA:1-25.

[12]Chen, G., Shen, J., & Barth-Cohen, L. et al. (2017). Assessing Elementary Students’ Computational Thinking in Everyday Reasoning and Robotics Programming[J]. Computers & Education, 109:162-175.

[13]Department for Education (2014). National Curriculum in England: Framework for Key Stages 1 to 4 [EB/OL]. [2019-03-07]. https://www.gov.uk/government/publications/al-curriculum-in-england-framework-for-key-stages-1-to-4/al-curriculum-in-england-framework-for-key-stages-1-to-4.

[14]Falloon, G. (2016). An Analysis of Young Students’ Thinking When Completing Basic Coding Tasks Using Scratch Jnr. on the iPad[J]. Journal of Computer Assisted Learning, 32(6):576-593.

[15]Heintz, F., Mannila, L., & Färnqvist, T. (2016). A Review of Models for Introducing Computational Thinking, Computer Science and Computing in K-12 Education[C]// Frontiers in Education Conference. PA,USA: IEEE: 1-9.

[16]HITSA (2015). ProgeTiger Programme 2015-2017[EB/OL]. [2019-04-15].  https://www.hitsa.ee/it-education/educational-

 programmes/progetiger.

[17]Krathwohl, D. R. (2002). A Revision of Bloom’s Taxonomy: An Overview[J]. Theory into Practice, 41(4):212-218.

[18]Mayer, R. E., Dyck, J. L., & Vilberg, W. (1986). Learning to Program and Learning to Think: What’s the Connection?[J]. Communications of the ACM, 29(7):605-610.

[19]Nardelli, E., & Ventre, G. (2015). Introducing Computational Thinking in Italian Schools: A First Report on “Programma il Futuro” Project[C]// INTED2015 Proceedings (9th International Technology, Education and Development Conference). Madrid, Spain: IATED:7414-7421.

[20]Seow, P., Looi, C. K., & Wadhwa, B. et al. (2017). Computational Thinking and Coding Initiatives in Singapore[C]// Proceedings of International Conference on Computational Thinking Education. Hong Kong: The Education University of Hong Kong:164-167.

[21]Wing, J. M. (2006). Computational Thinking[J]. Communications of the ACM, 49(3): 33-35.

[22]Wing J. M. (2010). Computational Thinking: What and Why?[EB/OL]. [2019-02-13]. http://www.cs.cmu.edu/~CompThink/papers/TheLinkWing.pdf.

收稿日期　2019-06-16　责任编辑　谭明杰

