附录 A　现有的编程代码测评指标

(1) 正确性指标指编程代码没有语法错误，包括能正确编译与链接、能正常执行、正确终止、正确的输入/输出、通过测试数据、预期输出与实际输出相符、正确的逻辑结构、功能正确、健壮性等，相关文献有 [1–105].

(2) 风格指标指编程代码具有良好的可读性和可维护性，包括适当的缩进、合适的标识符长度、较少的 goto 语句次数、简洁的代码、合理的程序布局、恰当的注释等，相关文献有 [4–5,7–10,12–17,19,21–22,24,26–29,32–33,35–39,41,43,45–48,50,54,58–59,61–62,66–69,72,75,78,80,82–83,86–87,90–95,99–100,106–119].

(3) 复杂度指标指代码性能，包括时间复杂度与空间复杂度. 时间复杂度指估算程序指令的执行次数 (算法执行时间消耗)，通常为算法执行时间随着输入数据规模增长的变化趋势，它通过代码中块重复的次数来衡量，即根据代码中循环或递归的次数增长情况来估量算法的执行时间长度；空间复杂度指代码运行过程中临时占用存储空间的大小，即算法的存储空间随着输入数据规模增长的变化趋势，空间复杂度分为两部分：一部分是固定的空间消耗 (比如代码存储、固定大小变量等)，另一部分是动态的空间消耗 (例如递归栈空间和动态分配的数据空间)，相关文献有[1–2,7,11,13,15,24,27–28,32,35,37–41,46–47,51,55–57,60,64–66,68–69,78–80,83,86,90,94–97,100,104,106,111,115].

(4) 代码相似性指标指计算学生编程代码与正确代码之间的相似程度，主要有编程代码的文本比较、结构特征比较、语法分析比较和语义分析比较. 文本比较包括模式匹配、正则表达式及编辑距离等；结构特征比较主要采用图相似计算的方式，包括对代码控制流图 (Control Flow Graph，CFG)、程序依赖图 (Program Dependence Graph，PDG)、系统依赖图 (System Dependence Graph，SDG)、数据流图 (Data Flow Graph，DFG) 与数据依赖图 (Data Dependency Graph，DDG) 的相似度计算；语法分析比较指计算编程代码的抽象语法树 (Abstract Syntax Tree，AST) 相似度；语义分析比较主要包括结构语义相似度和潜在语义的相似度，也包括语句、运算符、操作数与词法分析，相关文献有[9,23,26,30,33,43,53,55,74,77,81–82,94,101–103,114,116,119–137].

(5) 程序规范指标指编程代码中应严格遵循约定的关键字、知识点、编码标准等规范，相关文献有[8,14,20,23,29,33–36,42,50,52,71–72,79,81–82,91,99,102,138–140].

(6) 复杂性指标指编程代码的基本结构、逻辑和数据依赖关系，包含行数、语句数量、条件语句数量、循环语句数量、函数数量、函数调用数量、保留字数量、运算符数量、操作符数量、操作数数量、数字表示数量、大括号数量、方括号数量、圆括号数量、条件语句的深度、循环语句的深度、大括号的最大深度、方括号的最大深度、圆括号的最大深度等，相关文献有[2,7,9,11,15,39,48,57,67,79,82,85,100,106,118,121,140–145].

(7) 程序设计指标是指编程代码中选择合理的程序结构、数据结构和算法，相关的文献有[8,14,16,27–28,33–34,38,52,68,91,94–95,111–112,114–115].

(8) 模块化设计指标指编程代码中应首先关注结构性、布局与模块之间独立程度的耦合度量. 结构性指代码中各个模块的清晰定义、组织和层次结构，每个模块都有明确的功能；布局指模块之间的布局方式和排列顺序，各个模块之间的关系清晰明了；耦合度是衡量模块之间依赖关系的指标，高耦合度意味着一个模块的改动会对其他模块造成较大影响，低耦合度则表示模块之间相互独立，修改一个模块不会影响到其他模块，代码编写中尽量降低模块之间的耦合度，模块之间的依赖关系较小，更加独立，代码具有良好的可复用性，相关文献有[12,14,16,24,59–60,72,85,107,109,116,118,121,138].

(9) 效率指标指综合了编程过程中所需时间的长短、代码量的多少、算法和实现的优化度、可靠性以及编程的准确快速，旨在评估学生编程解决问题的综合能力. 涵盖了编写程序时所花费的时间、工作量、问题难度，也关注于学生在编程时选择算法和实现代码的效率，还考量编程过程中的准确性、可靠性和速度、解决方案提交的及时性，相关文献有[1–2,7,9,31,33,48,50,67,84,106,111,117].

(10) 抄袭检测指标指检测编程代码中是否存在抄袭与剽窃行为，主要指直接复制别人的代码，相关文献有[16,23,32,50,67,82,96,101,114,121].

(11) 逻辑指标指编程逻辑是否完整、正确，编程代码是否选择恰当的条件逻辑与循环逻辑，是否能够处理各种边界情况，是否能反映问题的需求，是否能够有效地识别并处理编程代码潜在的错误和异常情况，相关文献有[3,75–76,79,81,103,110–111].

(12) 创造力指标指具有问题解决策略的计算思维，设计出具有创新性的问题解决方案，即是否具有创新性和独特性，是否能够解决实际问题或提供新的思路和方法，相关的文献有[8,41,51,84,91,146].

(13) 软件度量指标指对编程代码质量进行量化的评估，包括代码量、循环复杂度 (Cyclomatic Complexity)、Halstead 复杂度 (Halstead Complexity Measures)、代码重用率、注释比例、错误密度 (缺陷数与代码量的比例) 等，相关文献有[68,89,114–115,119,121].

(14) 编程错误指标是指用来评估和识别程序中存在的错误或可疑代码片段的度量标准，如循环控制变量的更新错误、参数类型和使用不一致、未初始化的变量、冗余代码 (即永远不会被执行的代码)、语法错误等，相关文献有[68,114–115].

(15) 可扩展性指标是指在不干扰现有编程代码运行的前提下，程序应当具备灵活性，以便可以顺畅地引入新功能或调整现有功能，相关文献有[47,86].

(16) 异常指标指编程代码编译时异常和运行时异常，相关文献有[53,143].

(17) 变量指标指在编程代码中，学生是否能够正确地使用变量，相关文献有[53,143].

(18) 彻底性指标指测试用例能够覆盖到所有可能的情况和边界条件，但程序的行为可能是无限的，所以完全彻底的测试用例是不可能实现的，因此它是相对的. 测试用例尽可能地

覆盖可能出现的错误情况，以确保测试的有效性，相关文献有[63,139].

(19) 库调用指标包括使用内部库指标与外部库指标. 使用内部库指标指在编程代码中调用允许的库函数或方法来实现特定功能. 使用外部库指标指测评学生的代码是否使用了禁止使用的外部库，以及是否使用了允许使用的外部库并在程序中进行调用，相关文献有[33,53,143].

(20) 编程技能指标指学生编写程序时所展示的技术能力和知识水平，包括正确使用变量和数据类型、正确使用控制结构、正确使用函数和模块等，相关文献有[15,144].

(21) 鲁棒性指标指程序能否在面对不同的数据、期望或需求时保持弹性，相关文献有[27,57].

(22) 概念指标指测评学生在特定概念或技能 (如循环、条件等) 上的得分，检测学生在不同概念上的强项和弱项，相关文献有[79,117].

(23) 类指标指在面向对象程序设计中与类相关的指标，包括类的定义、数量、类中的变量数量、类中的库方法数量等，相关文献有[143–144].

(24) 计划指标指学生在编写代码之前，需要进行的规划和准备工作，包括问题规范、伪代码、结构图等的编写，相关文献有[4].

(25) 子程序使用指标指编程代码中可重复使用的代码片段，可以从程序的不同部分调用. 测评时考虑子程序数量、平均子程序长度、对子程序的调用次数、子程序调用的深度和传递给子程序的参数数量等指标，相关文献有[14].

(26) 安全性指标指在自动测评环境中提交的编程代码，不能含有攻击测评系统的行为，确保不会对系统造成损害或安全漏洞，避免通过作弊或欺诈行为获取不当的成绩，相关文献有[41].

(27) 问题分析指标指评估学生对于所要解决问题的理解能力，反映提交的编程代码在执行既定任务时表现出的有效语义能力，侧重于衡量编程代码是否能准确执行其预期的功能，相关文献有[51].

(28) 准确性指标指学生提交的编程代码是否能够正确地解决所提出的问题，相关文献有[59].

(29) 容差指标通常是一个小数，指学生编程代码的输出结果与标准答案之间的误差，不在容差范围内视为答案不正确，相关文献有[146].

(30) 时间阈值指标指学生提交的编程代码在执行过程中，如果超过了给定时间限制，就会被认为是超时的，评分系统将不会对其进行评分，直接以 0 分处理，相关文献有[146].

(31) 不违反内存指标指不存在可能导致内存冲突或引发运行时错误，相关文献有[60].

(32) 输入输出格式指标指编程代码满足编程问题中对于输入和输出的格式要求，包括大小写、空格等限制. 任何与格式不符的输入或输出都可能导致程序出错，即使算法本身是正确的，相关文献有[62].

(33) 表现力指标指代码的语言表达能力，即代码是否能够清晰地表达其意图和功能，相关文献有[116].

(34) 流控制指标指编程代码执行的顺序和条件，包括循环、条件语句、异常处理等，编程代码的流控制是否合理、清晰和易于理解，相关文献有[116].

(35) 推理指标指学生在使用编程解决问题时运用逻辑推理、抽象与推断的能力. 包括学生运用推理能力将问题拆解成更小、更具体的子问题，并确定问题的关键要素和约束条件的能力；能够阅读、理解和推理他人编写的代码；能够推理编程代码可能出现的异常情况，并正确处理这些异常；能够通过推理来判断编程代码的效率，进而提出优化方案，相关文献有[69].

(36) 数据类型指标指编程代码中使用的数据类型种类和数量，相关文献有[144].

(37) 属性指标指在面向对象程序设计中，类的属性 (字段) 必须满足规定的要求，相关文献有[78].

(38) 内聚性指标指模块内部各个元素之间的相关性和联系程度，用来衡量类数据和方法之间的关系强度，通常将模块分为低内聚性和高内聚性两种. 高内聚性通常指的是具有强大、可重用和可靠的模块，相关文献有[85].

(39) 重复性指标指编程代码中重复出现的代码块的数量和比例，重复代码块可能会导致代码冗余和错误，相关文献有[89].

(40) 表达式指标指编程代码中应该使用明确或精确的表达式，相关文献有[138].

参考文献

[1]FORSYTHE G E, WIRTH N. Automatic Grading Programs[J]. Commun. ACM, 1965, 8(5): 275–278. DOI: 10.1145/364914.364937.

[2]MILLER N E, PETERSON C G. A Method for Evaluating Student Written Computer Programs in an Undergraduate Computer Science Programming Language Course[J]. SIGCSE Bull., 1980, 12(4): 9–17. DOI: 10.1145/989274.989276.

[3]HAMM R W, HENDERSON K D, REPSHER M L, et al. A Tool for Program Grading: The Jacksonville University Scale[J]. SIGCSE Bull., 1983, 15(1): 248–252. DOI: 10.1145/952978.801059.

[4]OLSON D M. The Reliability of Analytic and Holistic Methods in Rating Students' Computer Programs[J]. SIGCSE Bull., 1988, 20(1): 293–298. DOI: 10.1145/52965.53037.

[5]ISAACSON P C, SCOTT T A. Automating the Execution of Student Programs[J]. SIGCSE Bull., 1989, 21(2): 15–22. DOI: 10.1145/65738.65741.

[6]REEK K A. The TRY System -or- How to Avoid Testing Student Programs[J]. SIGCSE Bull., 1989, 21(1): 112–116. DOI: 10.1145/65294.71198.

[7]HUNG S L, KWOK L F, CHAN R. Automatic Programming Assessment[J]. Comput. Educ., 1993, 20(2): 183–190. DOI: 10.1016/0360-1315(93)90086-X.

[8]HOWATT J W. On Criteria for Grading Student Programs[J]. SIGCSE Bull., 1994, 26(3): 3–7. DOI: 10.1145/187387.187389.

[9]BENFORD S D, BURKE E K, FOXLEY E, et al. The Ceilidh System for the Automatic Grading of Students on Programming Courses[C]//ACM-SE 33: Proceedings of the 33rd Annual on Southeast Regional Conference. Clemson, South Carolina: Association for Computing Machinery, 1995: 176–182. DOI: 10.1145/1122018.1122050.

[10]DROMEY R. A model for software product quality[J]. IEEE Transactions on Software Engineering, 1995, 21(2): 146–162. DOI: 10.1109/32.345830.

[11]JACKSON D. A software system for grading student computer programs[J]. Computers & Education, 1996, 27(3): 171–180. DOI: 10.1016/S0360-1315(96)00025-5.

[12]JOY M, LUCK M. Software standards in undergraduate computing courses[J]. Journal of Computer Assisted Learning, 1996, 12(2): 103–113. DOI: 10.1111/j.1365-2729.1996.tb00042.x.

[13]JACKSON D, USHER M. Grading Student Programs Using ASSYST[C]//SIGCSE '97: Proceedings of the Twenty-Eighth SIGCSE Technical Symposium on Computer Science Education. San Jose, California, USA: Association for Computing Machinery, 1997: 335–339. DOI: 10.1145/268084.268210.

[14]JOY M, LUCK M. Effective Electronic Marking for On-Line Assessment[C]//ITiCSE '98: Proceedings of the 6th Annual Conference on the Teaching of Computing and the 3rd Annual Conference on Integrating Technology into Computer Science Education: Changing the Delivery of Computer Science Education. Dublin City Univ., Ireland: Association for Computing Machinery, 1998: 134–138. DOI: 10.1145/282991.283096.

[15]FROSINI G, LAZZERINI B, MARCELLONI F. Performing automatic exams[J]. Computers & Education, 1998, 31(3): 281–300. DOI: 10.1016/S0360-1315(98)00042-6.

[16]LUCK M, JOY M. A Secure On-Line Submission System[J]. Softw. Pract. Exper., 1999, 29(8): 721–740.

[17]SHUKUR Z, BURKE E, FOXLEY E. The automatic assessment of formal specification coursework[J]. Journal of Computing in Higher Education, 1999, 11(1): 86–119. DOI: 10.1007/BF02940843.

[18]DALY C. RoboProf and an Introductory Computer Programming Course[J]. SIGCSE Bull., 1999, 31(3): 155–158. DOI: 10.1145/384267.305904.

[19]JACKSON D. A Semi-Automated Approach to Online Assessment[C]//ITiCSE '00: Proceedings of the 5th Annual SIGCSE/SIGCUE ITiCSEconference on Innovation and Technology in Computer Science Education. Helsinki, Finland: Association for Computing Machinery, 2000: 164–167. DOI: 10.1145/343048.343160.

[20]JONES E L. Grading Student Programs - a Software Testing Approach[J]. J. Comput. Sci. Coll., 2000, 16(2): 185–192.

[21]MCCRACKEN M, ALMSTRUM V, DIAZ D, et al. A Multi-National, Multi-Institutional Study of Assessment of Programming Skills of First-Year CS Students[C]//ITiCSE-WGR '01: Working Group Reports from ITiCSE on Innovation and Technology in Computer Science Education. Canterbury, UK: Association for Computing Machinery, 2001: 125–180. DOI: 10.1145/572133.572137.

[22]LAXER C, MCCRACKEN M, ALMSTRUM V, et al. Report by the ITiCSE 2001 Working Group on Assessment of Programming Skills of First-year CS Students[J]. 2001, 33(4): 125–180.

[23]SAIKKONEN R, MALMI L, KORHONEN A. Fully Automatic Assessment of Programming Exercises[J]. SIGCSE Bull., 2001, 33(3): 133–136. DOI: 10.1145/377435.377666.

[24]KURNIA A, LIM A, CHEANG B. Online Judge[J]. Comput. Educ., 2001, 36(4): 299–315. DOI: 10.1016/S0360-1315(01)00018-5.

[25]ENGLISH J. Experience with a Computer-Assisted Formal Programming Examination[J]. SIGCSE Bull., 2002, 34(3): 51–54. DOI: 10.1145/637610.544432.

[26]HIGGINS C, SYMEONIDIS P, TSINTSIFAS A. The Marking System for CourseMaster[J]. SIGCSE Bull., 2002, 34(3): 46–50. DOI: 10.1145/637610.544431.

[27]RUEHR F, ORR G. Interactive Program Demonstration as a Form of Student Program Assessment [J]. J. Comput. Sci. Coll., 2002, 18(2): 65–78. DOI: 10.5555/771322.771335.

[28]PILLAY N. Developing Intelligent Programming Tutors for Novice Programmers[J]. SIGCSE Bull., 2003, 35(2): 78–82. DOI: 10.1145/782941.782986.

[29]PISAN Y, RICHARDS D, SLOANE A, et al. Submit! A Web-Based System for Automatic Program Critiquing[C]//ACE '03: Proceedings of the Fifth Australasian Conference on Computing Education - Volume 20. Adelaide, Australia: Australian Computer Society, Inc., 2003: 59–68. DOI: 10.5555/858403.858411.

[30]BLUMENSTEIN M, GREEN S, NGUYEN A, et al. GAME: a Generic Automated Marking Environment for programming assessment[C]//International Conference on Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004. Vol. 1. Las Vegas, NV, USA: IEEE, 2004: 212–216. DOI: 10.1109/ITCC.2004.1286454.

[31]HARRIS J A, ADAMS E S, HARRIS N L. Making Program Grading Easier: But Not Totally Automatic[J]. J. Comput. Sci. Coll., 2004, 20(1): 248–261. DOI: 10.5555/1040231.1040264.

[32]TRUONG N, ROE P, BANCROFT P. Static Analysis of Students' Java Programs[C]//ACE '04: Proceedings of the Sixth Australasian Conference on Computing Education - Volume 30. Dunedin, New Zealand: Australian Computer Society, Inc., 2004: 317–325. DOI: 10.5555/979968.980011.

[33]JOY M, GRIFFITHS N, BOYATT R. The Boss Online Submission and Assessment System[J]. J. Educ. Resour. Comput., 2005, 5(3): 2–es. DOI: 10.1145/1163405.1163407.

[34]JUEDES D. Web-based grading: further experiences and student attitudes[C]//Proceedings Frontiers in Education 35th Annual Conference. 2005: F4E–18. DOI: 10.1109/FIE.2005.1612144.

[35]SMITH L, CORDOVA J. Weighted Primary Trait Analysis for Computer Program Evaluation[J]. J. Comput. Sci. Coll., 2005, 20(6): 14–19. DOI: 10.5555/1060405.1060409.

[36]WINTERS T, PAYNE T. Computer Aided Grading with Agar[C]//Frontiers in Education: Computer Science & Computer Engineering. Las Vegas, Nevada, USA: CSREA Press, 2006: 245–251.

[37]KUMAR MANDAL A, MANDAL C, READE C. A System for Automatic Evaluation of Programs for Correctness and Performance[C]//Proceedings of WEBIST 2006 - Second International Conference on Web Information Systems and Technologies: vol. 2. SciTePress, 2006: 196–203. DOI: 10.5220/0001251601960203.

[38]HELMICK M T. Interface-Based Programming Assignments and Automatic Grading of Java Programs[J]. SIGCSE Bull., 2007, 39(3): 63–67. DOI: 10.1145/1269900.1268805.

[39]MANDAL A K, MANDAL C, READE C. A System for Automatic Evaluation of Programs for Correctness and Performance[C]//Web Information Systems and Technologies. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007: 367–380. DOI: 10.1007/978-3-540-74063-6_29.

[40]HERNÁN-LOSADA I, PAREJA-FLORES C, VELÁZQUEZ-ITURBIDE J Á. Testing-Based Automatic Grading: A Proposal from Bloom's Taxonomy[C]//2008 Eighth IEEE International Conference on Advanced Learning Technologies. Santander, Spain: IEEE, 2008: 847–849. DOI: 10.1109/ICALT.2008.224.

[41]JENA S. Authoring and Sharing of Programming Exercises[D]. San Jose State University, 2008. DOI: 10.31979/etd.gevj-k88n.

[42]LINGLING M, XIAOJIE Q, ZHIHONG Z, et al. An Assessment Tool for Assembly Language Programming[C]//2008 International Conference on Computer Science and Software Engineering: vol. 5. IEEE, 2008: 882–884. DOI: 10.1109/CSSE.2008.111.

[43]SKUPAS B, DAGIENE V. Is Automatic Evaluation Useful for the Maturity Programming Exam? [C]//Koli '08: Proceedings of the 8th International Conference on Computing Education Research. Koli, Finland: Association for Computing Machinery, 2008: 117–118. DOI: 10.1145/1595356.1595382.

[44]SZTIPANOVITS M, QIAN K, FU X. The Automated Web Application Testing (AWAT) System[C]//ACM-SE 46: Proceedings of the 46th Annual Southeast Regional Conference on XX. Auburn, Alabama: Association for Computing Machinery, 2008: 88–93. DOI: 10.1145/1593105.1593128.

[45]TREMBLAY G, GUÉRIN F, PONS A, et al. Oto, a generic and extensible tool for marking programming assignments[J]. Software: Practice and Experience, 2008, 38(3): 307–333. DOI: 10.1002/spe.839.

[46]LIANG Y, LIU Q, XU J, et al. The Recent Development of Automated Programming Assessment [C]//2009 International Conference on Computational Intelligence and Software Engineering. Wuhan, China: IEEE, 2009: 1–5. DOI: 10.1109/CISE.2009.5365307.

[47]MONTOYA-DATO F J, FERNÁNDEZ-ALEMÁN J L, GARCÍA-MATEOS G. An Experience on Ada Programming Using On-Line Judging[C]//Reliable Software Technologies – Ada-Europe 2009. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009: 75–89. DOI: 10.1007/978-3-642-01924-1_6.

[48]ROMLI R, SULAIMAN S, ZAMLI K Z. Automatic programming assessment and test data generation a review on its approaches[C]//2010 International Symposium on Information Technology: vol. 3. IEEE, 2010: 1186–1192. DOI: 10.1109/ITSIM.2010.5561488.

[49]LORENZO E J, VELEZ J, PEÑAS A. A Proposal for Automatic Evaluation in a Compiler Construction Course[C]//ITiCSE '11: Proceedings of the 16th Annual Joint Conference on Innovation and Technology in Computer Science Education. Darmstadt, Germany: Association for Computing Machinery, 2011: 308–312. DOI: 10.1145/1999747.1999833.

[50]ROMLI R, SULAIMAN S, ZUHAIRI ZAMLI K. Current Practices of Programming Assessment at Higher Learning Institutions[C]//Software Engineering and Computer Systems. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011: 471–485. DOI: 10.1007/978-3-642-22170-5_41.

[51]ALMAJALI S. Computer-based tool for assessing advanced computer programming skills[C]//2012 International Conference on E-Learning and E-Technologies in Education (ICEEE). Lodz, Poland: IEEE, 2012: 114–118. DOI: 10.1109/ICeLeTE.2012.6333420.

[52]SEPPÄLÄ O, SC D, KORHONEN A, et al. Advances in Assessment of Programming Skills[M]. Aalto University, 2012.

[53]FATIMA A, ASHRAF E. An Intelligent Assessment Tool for Students' Java Submissions in Introductory Programming Courses[J]. Journal of Intelligent Learning Systems and Applications, 2012, 4(1): 59–69. DOI: 10.4236/jilsa.2012.41006.

[54]KLINIK M, KOOPMAN P, van der WAL R. Personal Prof: Automatic Code Review for Java Assignments[C]//CSERC '21: Proceedings of the 10th Computer Science Education Research Conference. Virtual Event, Netherlands: Association for Computing Machinery, 2022: 31–38. DOI: 10.1145/3507923.3507930.

[55]FONTE D, CRUZ D D, GANÇARSKI A L, et al. A Flexible Dynamic System for Automatic Grading of Programming Exercises[C]//Open Access Series in Informatics (OASIcs): 2nd Symposium on Languages, Applications and Technologies: vol. 29. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2013: 129–144. DOI: 10.4230/OASIcs.SLATE.2013.129.

[56]GUTIÉRREZ E D, TRENAS M A, CORBERA F, et al. An Experience of e-assessment in an Introductory Course on Computer Organization[J]. Procedia Computer Science, 2013, 18: 1436–1445. DOI: 10.1016/j.procs.2013.05.311.

[57]LIU X. A new automated grading approach for computer programming[J]. Computer Applications in Engineering Education, 2013, 21(3): 484–490. DOI: 10.1002/cae.20494.

[58]MCCARTNEY R, BOUSTEDT J, ECKERDAL A, et al. Can First-Year Students Program yet? A Study Revisited[C]//ICER '13: Proceedings of the Ninth Annual International ACM Conference on International Computing Education Research. San Diego, San California, USA: Association for Computing Machinery, 2013: 91–98. DOI: 10.1145/2493394.2493412.

[59]RASHKOVITS R, LAVY I. FACT: A Formative Assessment Criteria Tool for the Assessment of Students' Programming Tasks[J]. Lecture Notes in Engineering and Computer Science, 2013, 1: 384–389.

[60]VUJOŠEVIĆ-JANIČIĆ M, NIKOLIĆ M, TOŠIĆ D, et al. Software verification and graph similarity for automated evaluation of students' assignments[J]. Information and Software Technology, 2013, 55(6): 1004–1016. DOI: j.infsof.2012.12.005.

[61]MCCRACKEN M. Evaluation of programs for the iticse working group on programming skill assessment[Z]. Accessed on 2023-05-06. URL https://www.cc.gatech.edu/projects/iticsewg/evaluation.pdf. 2013.

[62]LLANA L, MARTIN-MARTIN E, PAREJA-FLORES C, et al. FLOP: A User-Friendly System for Automated Program Assessment[J]. JUCS - Journal of Universal Computer Science, 2014, 20(9): 1304–1326. DOI: 10.3217/jucs-020-09-1304.

[63]POLITZ J G, KRISHNAMURTHI S, FISLER K. In-Flow Peer-Review of Tests in Test-First Programming[C]//ICER '14: Proceedings of the Tenth Annual Conference on International Computing Education Research. Glasgow, Scotland, United Kingdom: Association for Computing Machinery, 2014: 11–18. DOI: 10.1145/2632320.2632347.

[64]RUBIO-SÁNCHEZ M, KINNUNEN P, PAREJA-FLORES C, et al. Student Perception and Usage of an Automated Programming Assessment Tool[J]. Comput. Hum. Behav., 2014, 31: 453–460.

DOI: 10.1016/j.chb.2013.04.001.

[65]WANG Y N, XIAO L N. Research on Automatic Scoring Methods for Programs Based on Program Understanding[J]. Applied Mechanics and Materials, 2014, 513: 2054–2058. DOI: 10.4028/www.scientific.net/AMM.513-517.2054.

[66]MITTAL H, MANDALIKA S D. Framework for Evaluation of Programming Language Examinations[M]. Singapore: Springer Singapore, 2015: 121–128. DOI: 10.1007/978-981-287-338-5_10.

[67]POŽENEL M, FÜRST L, MAHNIČ V. Introduction of the automated assessment of homework assignments in a university-level programming course[C]//2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). Opatija, Croatia: IEEE, 2015: 761–766. DOI: 10.1109/MIPRO.2015.7160373.

[68]UREEL L C, WALLACE C. WebTA: Automated iterative critique of student programming assignments[C]//2015 IEEE Frontiers in Education Conference (FIE). El Paso, TX, USA: IEEE, 2015: 1–9. DOI: 10.1109/FIE.2015.7344225.

[69]CATETÉ V, SNIDER E, BARNES T. Developing a Rubric for a Creative CS Principles Lab[C]//ITiCSE '16: Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education. Arequipa, Peru: Association for Computing Machinery, 2016: 290–295. DOI: 10.1145/2899415.2899449.

[70]CHRISTIAN M, TRIVEDI B. A Comparison of Existing Tools for Evaluation of Programming Exercises[C]//ICTCS '16: Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies. Udaipur, India: Association for Computing Machinery, 2016. DOI: 10.1145/2905055.2905350.

[71]INSA D, SILVA J. Computer assisted self-assessment of programming code: A report on university students experience and opinion[C]//2016 15th International Conference on Information Technology Based Higher Education and Training (ITHET). Istanbul, Turkey: IEEE, 2016: 1–3. DOI: 10.1109/ITHET.2016.7760727.

[72]SOUZA D M, FELIZARDO K R, BARBOSA E F. A Systematic Literature Review of Assessment Tools for Programming Assignments[C]//2016 IEEE 29th International Conference on Software Engineering Education and Training (CSEET). Dallas, TX, USA: IEEE, 2016: 147–156. DOI: 10.1109/CSEET.2016.48.

[73]STAUBITZ T, KLEMENT H, TEUSNER R, et al. CodeOcean - A versatile platform for practical programming excercises in online environments[C]//2016 IEEE Global Engineering Education Conference (EDUCON). Abu Dhabi, United Arab Emirates: IEEE, 2016: 314–323. DOI: 10.1109/EDUCON.2016.7474573.

[74]ZOUGARI S, TANANA M, LYHYAOUI A. Hybrid assessment method for programming assignments[C]//2016 4th IEEE International Colloquium on Information Science and Technology (CiSt). Tangier, Morocco: IEEE, 2016: 564–569. DOI: 10.1109/CIST.2016.7805112.

[75]PETTIT R, PRATHER J. Automated Assessment Tools: Too Many Cooks, Not Enough Collaboration[J]. J. Comput. Sci. Coll., 2017, 32(4): 113–121. DOI: 10.5555/3055338.3079060.

[76]ALBLUWI I. A Closer Look at the Differences Between Graders in Introductory Computer Science Exams[J]. IEEE Transactions on Education, 2018, 61(3): 253–260. DOI: 10.1109/TE.2018.2805706.

[77]HOUSSEIN S A, PETER Y. Evaluation of Algorithms to Support Novice Programmer[C]//ICETC '18: Proceedings of the 10th International Conference on Education Technology and Computers. Tokyo, Japan: Association for Computing Machinery, 2018: 383–387. DOI: 10.1145/3290511.3290529.

[78]INSA D, SILVA J. Automatic assessment of Java code[J]. Computer Languages, Systems & Structures, 2018, 53: 59–72. DOI: 10.1016/j.cl.2018.01.004.

[79]LIEBENBERG J, PIETERSE V. Investigating the Feasibility of Automatic Assessment of Programming Tasks[J]. Journal of Information Technology Education: Innovations in Practice, 2018, 17: 201–223. DOI: 10.28945/4150.

[80]NANDAGOPAL N, CHANDRASHEKAR S, REDDY S, et al. Flexible Automatic Evaluation of Dynamic and Static Properties of Code[C]//2018 IEEE Tenth International Conference on Technology for Education (T4E). Chennai, India: IEEE, 2018: 214–215. DOI: 10.1109/T4E.2018.00057.

[81]SZABÓ M, NEHÉZ K. Grading Java code submissions in MeMOOC[C]//MultiScience - XXXII. microCAD International Multidisciplinary Scientific Conference. University of Miskolc, 2018. DOI: 10.26649/musci.2018.026.

[82]ULLAH Z, LAJIS A, JAMJOOM M, et al. The effect of automatic assessment on novice programming: Strengths and limitations of existing systems[J]. Computer Applications in Engineering Education, 2018, 26(6): 2328–2341. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cae.21974. DOI: 10.1002/cae.21974.

[83]HAMEER A, PIENTKA B. Teaching the Art of Functional Programming Using Automated Grading (Experience Report)[J]. Proc. ACM Program. Lang., 2019, 3(ICFP): 1–15. DOI: 10.1145/3341719.

[84]HASS B, YUAN C, LI Z. On the Automatic Assessment of Learning Outcome in Programming Techniques[C]//2019 IEEE 14th International Conference on Intelligent Systems and Knowledge Engineering (ISKE). Dalian, China: IEEE, 2019: 274–278. DOI: 10.1109/ISKE47853.2019.9170370.

[85]KALEMBA E, ADE-IBIJOLA A. A Metric for Estimating the Difficulty of Programming Problems by Ranking the Constructs in their Solutions[C]//2019 International Multidisciplinary Information Technology and Engineering Conference (IMITEC). Vanderbijlpark, South Africa: IEEE, 2019: 1–9. DOI: 10.1109/IMITEC45504.2019.9015843.

[86]BERTAGNON A, GAVANELLI M. MAESTRO: a semi-automated Evaluation System for programming assignments[C]//2020 International Conference on Computational Science and Com-

putational Intelligence (CSCI). Las Vegas, NV, USA: IEEE, 2020: 953–958. DOI: 10.1109/CSCI51800.2020.00177.

[87]COSTANTINI U, LONATI V, MORPURGO A. How Plans Occur in Novices' Programs: A Method to Evaluate Program-Writing Skills[C]//SIGCSE '20: Proceedings of the 51st ACM Technical Symposium on Computer Science Education. Portland, OR, USA: Association for Computing Machinery, 2020: 852–858. DOI: 10.1145/3328778.3366870.

[88]MEKTEROVIĆ I, BRKIĆ L, MILAŠINOVIĆ B, et al. Building a Comprehensive Automated Programming Assessment System[J]. IEEE Access, 2020, 8: 81154–81172. DOI: 10.1109/ACCESS.2020.2990980.

[89]PINTÉR Á, SZÉNÁSI S. Automatic Analysis and Evaluation of Student Source Codes[C]//2020 IEEE 20th International Symposium on Computational Intelligence and Informatics (CINTI). Budapest, Hungary: IEEE, 2020: 000161–000166. DOI: 10.1109/CINTI51262.2020.9305819.

[90]ZAMPROGNO L, HOLMES R, BANIASSAD E. Nudging Student Learning Strategies Using Formative Feedback in Automatically Graded Assessments[C]//SPLASH-E 2020: Proceedings of the 2020 ACM SIGPLAN Symposium on SPLASH-E. Virtual, USA: Association for Computing Machinery, 2020: 1–11. DOI: 10.1145/3426431.3428654.

[91]ISMAIL H, LAKULU M M. A Critical Review on Recent Proposed Automated Programming Assessment Tool[J]. Psychology and Education, 2020, 57(8): 1049–1060. DOI: 10.17762/TURCOMAT.V12I3.799.

[92]KURUPPU T, THARMASEELAN J, SILVA C, et al. Source Code based Approaches to Automate Marking in Programming Assignments[C]//Proceedings of the 13th International Conference on Computer Supported Education: vol. 1. SciTePress, 2021: 291–298. DOI: 10.5220/0010400502910298.

[93]NABIL R, MOHAMED N E, MAHDY A, et al. EvalSeer: An Intelligent Gamified System for Programming Assignments Assessment[C]//2021 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC). Cairo, Egypt: IEEE, 2021: 235–242. DOI: 10.1109/MIUCC52538.2021.9447629.

[94]AGRAWAL A, REED B. A SURVEY ON GRADING FORMAT OF AUTOMATED GRADING TOOLS FOR PROGRAMMING ASSIGNMENTS[C]//ICERI2022: ICERI Proceedings. IATED, 2022. DOI: 10.21125/iceri.2022.1912.

[95]CALIFF M E, DUNNE N. Feedback in Context: Using a Code Review Tool for Program Grading [C]//SIGCSE 2022: Proceedings of the 53rd ACM Technical Symposium on Computer Science Education - Volume 1. Providence, RI, USA: Association for Computing Machinery, 2022: 92–97. DOI: 10.1145/3478431.3499402.

[96]COMBÉFIS S. Automated Code Assessment for Education: Review, Classification and Perspectives on Techniques and Tools[J]. Software, 2022, 1(1): 3–30. DOI: 10.3390/software1010002.

[97]RAHMAN M M, WATANOBE Y, MATSUMOTO T, et al. Educational Data Mining to Support Programming Learning Using Problem-Solving Data[J]. IEEE Access, 2022, 10: 26186–26202. DOI: 10.1109/ACCESS.2022.3157288.

[98]ADITI A. CodEval[D]. San Jose State University, 2023. DOI: 10.31979/etd.kj7h-7xxh.

[99]MESSER M, BROWN N C C, KÖLLING M, et al. Machine Learning-Based Automated Grading and Feedback Tools for Programming: A Meta-Analysis[C]//ITiCSE 2023: Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1. Turku, Finland: Association for Computing Machinery, 2023: 491–497. DOI: 10.1145/3587102.3588822.

[100]RAHMAN M M, WATANOBE Y, HAMADA M. A Survey on Automated Code Evaluation Systems and Their Resources for Code Analysis[C]//Advances and Trends in Artificial Intelligence. Theory and Applications: vol. 13926. Cham: Springer Nature Switzerland, 2023: 385–396. DOI: 10.1007/978-3-031-36822-6_33.

[101]修晓杰, 唐红军. C 语言程序评测方法研究[J]. 杭州电子科技大学学报, 2012, 32(03): 57–60. DOI: 10.13954/j.cnki.hdu.2012.03.023.

[102]苏小红, 王宇颖, 王甜甜, 等. 面向综合实践能力考核的 C 语言编程考试自动评分系统[J]. 实验技术与管理, 2010, 27(10): 174–177. DOI: 10.16791/j.cnki.sjg.2010.10.053.

[103]何烨辛. 基于 CEMs 聚类分析与语义结合的程序自动测评方法研究[D]. 西安工程大学, 2022.

[104]THAMVISET W. Automatic Unit Testing-Based Assessments for Online C++ Programming Classroom[C]//2022 19th International Joint Conference on Computer Science and Software Engineering (JCSSE). Bangkok, Thailand: IEEE, 2022: 1–6. DOI: 10.1109/JCSSE54890.2022.9836289.

[105]ZUO F, RHEE J, PARK M, et al. PowerGrader: Automating Code Assessment Based on PowerShell for Programming Courses[C]//2023 IEEE/ACIS 21st International Conference on Software Engineering Research, Management and Applications (SERA). Orlando, FL, USA: IEEE, 2023: 2–7. DOI: 10.1109/SERA57763.2023.10197671.

[106]WEINBERG G M, SCHULMAN E L. Goals and Performance in Computer Programming[J]. Human Factors, 1974, 16(1): 70–77. DOI: 10.1177/001872087401600108.

[107]TASSEL D V. Program Style, Design, Efficiency, DeBugging and Testing[M]. 2nd. USA: Prentice Hall PTR, 1978.

[108]BERRY R E, MEEKINGS B A. A Style Analysis of C Programs[J]. Commun. ACM, 1985, 28(1): 80–88. DOI: 10.1145/2465.2469.

[109]REDISH K A, SMYTH W F. Program Style Analysis: A Natural by-Product of Program Compilation[J]. Commun. ACM, 1986, 29(2): 126–133. DOI: 10.1145/5657.5661.

[110]OMAN P W, COOK C R. A programming style taxonomy[J]. Journal of Systems and Software, 1991, 15(3): 287–301. DOI: 10.1016/0164-1212(91)90044-7.

[111]HUNG S L, KWOK L F, CHUNG A. New Metrics for Automated Programming Assessment[C]//Proceedings of the IFIP WG3.4/SEARCC (SRIG on Education and Training) Working Conference on Software Engineering Education. NLD: North-Holland Publishing Co., 1993: 233–243. DOI: 10.5555/647118.717052.

[112]JUEDES D. Experiences in Web-based grading[C]//33rd Annual Frontiers in Education, 2003. FIE 2003. Vol. 3. Westminster, CO, USA: IEEE, 2003: S3F–27. DOI: 10.1109/FIE.2003.1266003.

[113]ALA-MUTKA K, UIMONEN T, JÄRVINEN H M. Supporting Students in C++ Programming Courses with Automatic Program Style Assessment[J]. JITE, 2004, 3(1): 245–262. DOI: 10.28945/300.

[114]ALA-MUTKA K M. A Survey of Automated Assessment Approaches for Programming Assignments[J]. Computer Science Education, 2005, 15(2): 83–102. DOI: 10.1080/08993400500150747.

[115]IHANTOLA P. Automated Assessment of Programming Assignments: Visual Feedback, Assignment Mobility, and Assessment of Students' Testing Skills[D]. Aalto University, 2011.

[116]STEGEMAN M, BARENDSEN E, SMETSERS S. Towards an Empirically Validated Model for Assessment of Code Quality[C]//Koli Calling '14: Proceedings of the 14th Koli Calling International Conference on Computing Education Research. Koli, Finland: Association for Computing Machinery, 2014: 99–108. DOI: 10.1145/2674683.2674702.

[117]PARIHAR S. Automated Grading Tool for Introductory Programming[D]. Indian Institute of Technology Kanpur, 2015.

[118]De OLIVEIRA M G, NEVES Á O S, LOPES M F S. Automatic Mapping of Student 3D Profiles in Software Metrics for Temporal Analysis of Programming Learning and Scoring Rubrics[M]. Rijeka: IntechOpen, 2018. DOI: 10.5772/intechopen.81754.

[119]GUPTA S, GUPTA A. E-Assessment Tools for Programming Languages: A Review[C]//International Conference on Information Technology and Knowledge Management: vol. 14. 2018: 65–70. DOI: 10.15439/2018KM31.

[120]ROBINSON S S, SOFFA M L. An Instructional Aid for Student Programs[J]. SIGCSE Bull., 1980, 12(1): 118–129. DOI: 10.1145/953032.804623.

[121]LEACH R J. Using Metrics to Evaluate Student Programs[J]. SIGCSE Bull., 1995, 27(2): 41–43. DOI: 10.1145/201998.202010.

[122]LI G, WU W, SUN Y, et al. Transformation-based assessment for C programs[C]//2007 9th International Symposium on Signal Processing and Its Applications. Sharjah: IEEE, 2007: 1–4. DOI: 10.1109/ISSPA.2007.4555367.

[123]RAHMAN K, AHMAD S, NORDIN M J. The Design of an Automated C Programming Assessment Using Pseudo-code Comparison Technique[C]//National Conference on Software Engineering and Computer Systems. University Malaysia Pahang, Pahang, Malaysia, 2007: 1–10.

[124]WU W, LI G, SUN Y, et al. AnalyseC: A Framework for Assessing Students' Programs at Structural and Semantic Level[C]//2007 IEEE International Conference on Control and Automation. Guangzhou, China: IEEE, 2007: 742–747. DOI: 10.1109/ICCA.2007.4376454.

[125]WANG T, SU X, WANG Y, et al. Semantic Similarity-Based Grading of Student Programs[J]. Inf. Softw. Technol., 2007, 49(2): 99–107. DOI: 10.1016/j.infsof.2006.03.001.

[126]TAHERKHANI A, MALMI L, KORHONEN A. Algorithm Recognition by Static Analysis and Its Application in Students' Submissions Assessment[C]//Koli '08: Proceedings of the 8th International Conference on Computing Education Research. Koli, Finland: Association for Computing Machinery, 2008: 88–91. DOI: 10.1145/1595356.1595372.

[127]NAUDÉ K A, GREYLING J H, VOGTS D. Marking student programs using graph similarity[J]. Computers & Education, 2010, 54(2): 545–561. DOI: 10.1016/j.compedu.2009.09.005.

[128]ZEN K, ISKANDAR D N F A, LINANG O. Using Latent Semantic Analysis for automated grading programming assignments[C]//2011 International Conference on Semantic Technology and Information Retrieval. Putrajaya, Malaysia: IEEE, 2011: 82–88. DOI: 10.1109/STAIR.2011.5995769.

[129]BUYRUKOGLU S, BATMAZ F, LOCK R. Increasing the similarity of programming code structures to accelerate the marking process in a new semi-automated assessment approach[C]//2016 11th International Conference on Computer Science & Education (ICCSE). Nagoya, Japan: IEEE, 2016: 371–376. DOI: 10.1109/ICCSE.2016.7581609.

[130]GRIVOKOSTOPOULOU F, PERIKOS I, HATZILYGEROUDIS I. An Educational System for Learning Search Algorithms and Automatically Assessing Student Performance[J]. International Journal of Artificial Intelligence in Education, 2017, 27(1): 207–240. DOI: 10.1007/s40593-016-0116-x.

[131]ARIFI S M, ABBOU R B, ZAHI A. A New Similarity-based Method for Assessing Programming Assignments using Symbolic Execution[J]. International Journal of Applied Engineering Research, 2018, 13(14): 1963–1981.

[132]AKRAM B, AZIZSOLTANI H, MIN W, et al. Automated Assessment of Computer Science Competencies from Student Programs with Gaussian Process Regression[C]//Proceedings of the 13th International Conference on Educational Data Mining, EDM 2020, Fully virtual conference. International Educational Data Mining Society, 2020: 555–560.

[133]薄钧戈, 乔亚男, 齐琪, 等. 基于编辑距离的自适应反馈程序评测方法[J]. 计算机技术与发展, 2022, 32(08): 135–141.

[134]马培军, 王甜甜, 苏小红. 基于程序理解的编程题自动评分方法[J]. 计算机研究与发展, 2009, 46(07): 1136–1142.

[135]王甜甜. 结构语义相似的程序识别方法研究[D]. 哈尔滨工业大学, 2009.

[136]王克朝, 王甜甜, 苏小红, 等. 面向程序理解的系统依赖图构建算法[J]. 哈尔滨工业大学学报, 2013, 45(01): 78–84.

[137]INTURI S, SWAMYDAS M. Programming Assignment Grading Through Control Statement and Program Features[C]//2023 International Conference on Emerging Techniques in Computational Intelligence (ICETCI). Hyderabad, India: IEEE, 2023: 122–129. DOI: 10.1109/ICETCI58599.2023.10331134.

[138]STEGEMAN M, BARENDSEN E, SMETSERS S. Designing a Rubric for Feedback on Code Quality in Programming Courses[C]//Koli Calling '16: Proceedings of the 16th Koli Calling International Conference on Computing Education Research. Koli, Finland: Association for Computing Machinery, 2016: 160–164. DOI: 10.1145/2999541.2999555.

[139]WRENN J, KRISHNAMURTHI S, FISLER K. Who Tests the Testers?[C]//ICER '18: Proceedings of the 2018 ACM Conference on International Computing Education Research. Espoo, Finland: Association for Computing Machinery, 2018: 51–59. DOI: 10.1145/3230977.3230999.

[140]VERMA A, UDHAYANAN P, SHANKAR R M, et al. Source-Code Similarity Measurement: Syntax Tree Fingerprinting for Automated Evaluation[C]//AIMLSystems '21: Proceedings of the First International Conference on AI-ML Systems. Bangalore, India: Association for Computing Machinery, 2021. DOI: 10.1145/3486001.3486228.

[141]CLUTTERHAM D R. A Method for Evaluating Student Progress in Undergraduate Computer Science through Automated Problem Sets[EB/OL]. 1970. https://files.eric.ed.gov/fulltext/ED051665.pdf.

[142]MENGEL S A, YERRAMILLI V. A Case Study of the Static Analysis of the Quality of Novice Student Programs[J]. SIGCSE Bull., 1999, 31(1): 78–82. DOI: 10.1145/384266.299689.

[143]ALSHAMSI F, ELNAGAR A. An automated assessment and reporting tool for introductory Java programs[C]//2011 International Conference on Innovations in Information Technology. Abu Dhabi, United Arab Emirates: IEEE, 2011: 324–329. DOI: 10.1109/INNOVATIONS.2011.5893842.

[144]FERREIRA NOVAIS D, VARANDA PEREIRA M J, RANGEL HENRIQUES P. Profile Detection Through Source Code Static Analysis[C]//Open Access Series in Informatics (OASIcs): 5th Symposium on Languages, Applications and Technologies (SLATE'16): vol. 51. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016: 9:1–9:13. DOI: 10.4230/OASIcs.SLATE.2016.9.

[145]ALPER B, NAZLIOGLU S, KILIC H. ACE-PE: An Automated Code Evaluation Software Tool For Programming Education[C]//2023 11th International Symposium on Digital Forensics and Security (ISDFS). Chattanooga, TN, USA: IEEE, 2023: 1–5. DOI: 10.1109/ISDFS58141.2023.10131776.

[146]SIOSON A A. Experiences on the use of an automatic C++ solution grader system[C]//IISA 2013. Piraeus, Greece: IEEE, 2013: 1–6. DOI: 10.1109/IISA.2013.6623681.