ACKNOWLEDGMENTS

This research is supported by the Dutch Research Council (NWO), grant number 023.005.063.

REFERENCES

[1] 2018. PMD. (2018). https://pmd.github.io/pmd-6.9.0/
[2] 2019. IntelliJ IDEA Community Edition 2019.2. (2019). https://www.jetbrains.com/idea
[3] 2020. Ace editor. (2020). https://ace.c9.io/
[4] 2020. Ideas. (2020). http://hackage.haskell.org/package/ideas
[5] Kirsti Ala-Mutka, Toni Uimonen, and Hannu-Matti Jarvinen. 2004. Supporting students in C++ programming courses with automatic program style assessment. *Journal of Information Technology Education: Research* 3 (2004), 245–262.
[6] Hannah Blau and J. Eliot B. Moss. 2015. FrenchPress Gives Students Automated Feedback on Java Program Flaws. In *Proceedings of ITiCSE*. 15–20. DOI: https://doi.org/10.1145/2729094.2742622
[7] Jürgen Börstler, Harald Störrle, Daniel Toll, Jelle van Assema, Rodrigo Duran, Sara Hooshangi, Johan Jeuring, Hieke Keuning, Carsten Kleiner, and Bonnie MacKellar. 2017. "I know it when I see it" Perceptions of Code Quality. In *Proceedings of ITiCSE, Working Group Reports*. 70–85. DOI: https://doi.org/10.1145/3174781.3174785
[8] Dennis Breuker, Jan Derriks, and Jacob Brunekreef. 2011. Measuring Static Quality of Student Code. In *Proceedings of ITiCSE*. 13–17. DOI: https://doi.org/10.1145/1999747.1999754
[9] Neil CC Brown and Amjad Altadmri. 2017. Novice Java programming mistakes: large-scale data vs. educator beliefs. *ACM Transactions on Computing Education (TOCE)* 17, 2 (2017), 7. DOI: https://doi.org/10.1145/2994154
[10] Rohan Roy Choudhury, Hezheng Yin, and Armando Fox. 2016. Scale-Driven Automatic Hint Generation for Coding Style. In *International Conference on Intelligent Tutoring Systems*. Vol. 9684 LNCS. 122–132. DOI: https://doi.org/10.1007/978-3-319-39583-8_12
[11] Tyne Crow, Andrew Luxton-Reilly, and Burkhard Wuensche. 2018. Intelligent tutoring systems for programming education: a systematic review. In *Proceedings of the 20th Australasian Computing Education Conference*. ACM, 53–62. DOI: https://doi.org/10.1145/3160489.3160492
[12] Giuseppe De Ruvo, Ewan Tempero, Andrew Luxton-Reilly, Gerard B. Rowe, and Nasser Giacaman. 2018. Understanding Semantic Style by Analysing Student Code. In *Proceedings of the Australasian Computing Education Conference*. 73–82. DOI: https://doi.org/10.1145/3160489.3160500
[13] Stephen Edwards, Nischel Kandru, and Mukund Rajagopal. 2017. Investigating static analysis errors in student Java programs. In *Proceedings of the 2017 ACM Conference on International Computing Education Research*. ACM, 65–73. DOI: https://doi.org/10.1145/3105726.3106182
[14] Stephen Edwards, Jaime Spacco, and David Hovemeyer. 2019. Can Industrial-Strength Static Analysis Be Used to Help Students Who Are Struggling to Complete Programming Activities?. In *Proceedings of the 52nd Hawaii International Conference on System Sciences*. 7825–7834. DOI: https://doi.org/10.24251/HICSS.2019.941
[15] Martin Fowler. 1999. *Refactoring: improving the design of existing code*. Addison-Wesley Professional.
[16] Alex Gerdes, Bastiaan Heeren, Johan Jeuring, and Thomas van Binsbergen. 2017. Ask-Elle: an adaptable programming tutor for Haskell giving automated feedback. *International Journal of Artificial Intelligence in Education* 27, 1 (2017), 65–100.
[17] B. Heeren and J. Jeuring. 2014. Feedback services for stepwise exercises. *Science of Computer Programming* 88 (2014), 110–129. DOI: https://doi.org/10.1016/j.scico.2014.02.021
[18] Hieke Keuning, Bastiaan Heeren, and Johan Jeuring. 2017. Code Quality Issues in Student Programs. In *ITiCSE*. 110–115. DOI: https://doi.org/10.1145/3059009.3059061
[19] Hieke Keuning, Bastiaan Heeren, and Johan Jeuring. 2019. How Teachers Would Help Students to Improve Their Code. In *ITiCSE*. 119–125. DOI: https://doi.org/10.1145/3304221.3319780
[20] Hieke Keuning, Bastiaan Heeren, and Johan Jeuring. 2020. Student Refactoring Behaviour in a Programming Tutor. In *Proceedings of the 20th Koli Calling International Conference on Computing Education Research*. DOI: https://doi.org/10.1145/3428029.3428043
[21] Hieke Keuning, Johan Jeuring, and Bastiaan Heeren. 2018. A systematic literature review of automated feedback generation for programming exercises. *ACM Transactions on Computing Education (TOCE)* 19, 1 (2018). DOI: https://doi.org/10.1145/3231711
[22] Nguyen-Thinh Le and Niels Pinkwart. 2014. Towards a Classification for Programming Exercises. In *Workshop on AI-supported Education for Computer Science*. 51–60.
[23] Andrew Luxton-Reilly, Paul Denny, Diana Kirk, Ewan Tempero, and Se-Young Yu. 2013. On the Differences Between Correct Student Solutions. In *Proceedings of ITiCSE*. 177–182. DOI: https://doi.org/10.1145/2462476.2462505
[24] Steve McConnell. 2004. *Code Complete: A Practical Handbook of Software Construction, Second Edition*. Microsoft Press.
[25] Emerson Murphy-Hill, Chris Parnin, and Andrew P Black. 2011. How we refactor, and how we know it. *IEEE Transactions on Software Engineering* 38, 1 (2011), 5–18. DOI: https://doi.org/10.1109/TSE.2011.41
[26] Stephen Nutbrown and Colin Higgins. 2016. Static analysis of programming exercises: Fairness, usefulness and a method for application. *Computer Science Education* 26, 2-3 (2016), 104–128. DOI: https://doi.org/10.1080/08993408.2016.1179865
[27] Raymond Pettit, John Homer, Roger Gee, Susan Mengel, and Adam Starbuck. 2015. An Empirical Study of Iterative Improvement in Programming Assignments. In *Proceedings of SIGCSE*. 410–415. DOI: https://doi.org/10.1145/2676723.2677279
[28] Yizhou Qian and James Lehman. 2017. Students' Misconceptions and Other Difficulties in Introductory Programming: A Literature Review. *ACM Transactions on Computing Education (TOCE)* 18, 1, Article 1 (2017), 1:1–1:24 pages. DOI: https://doi.org/10.1145/3077618
[29] Kelly Rivers and Kenneth R Koedinger. 2017. Data-driven hint generation in vast solution spaces: a self-improving python programming tutor. *International Journal of Artificial Intelligence in Education* 27, 1 (2017), 37–64. DOI: https://doi.org/10.1007/s40593-015-0070-z
[30] Leo C. Ureel II and Charles Wallace. 2019. Automated Critique of Early Programming Antipatterns. In *Proceedings of SIGCSE*. ACM, 738âĂŞ744. DOI: https://doi.org/10.1145/3287324.3287463
[31] Kurt VanLehn. 2006. The Behavior of Tutoring Systems. *International Journal of Artificial Intelligence in Education* 16, 3 (2006), 227–265.
[32] Kurt VanLehn. 2011. The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems. *Educational Psychologist* 46, 4 (2011), 197–221. DOI: https://doi.org/10.1080/00461520.2011.611369
[33] Roel Wieringa. 2014. *Design science methodology for information systems and software engineering*. Springer. DOI: https://doi.org/10.1007/978-3-662-43839-8
[34] Eliane S. Wiese, Michael Yen, Antares Chen, Lucas A. Santos, and Armando Fox. 2017. Teaching Students to Recognize and Implement Good Coding Style. In *Proceedings of Learning @ Scale (L@S)*. 41–50. DOI: https://doi.org/10.1145/3051457.3051469
[35] Songwen Xu and Yam San Chee. 2003. Transformation-based diagnosis of student programs for programming tutoring systems. *IEEE Transactions on Software Engineering* 29, 4 (2003), 360–384. DOI: https://doi.org/10.1109/TSE.2003.1191799