Code Control: Developing a Serious Game to Reinforce Introductory Programming Concepts
SIGCSE Final Report

PI: Devorah Kletenik, Brooklyn College, City University of New York, kletenik@sci.brooklyn.cuny.edu

Co-PI: Deborah Sturm, College of Staten Island, City University of New York, deborah.sturm@csi.cuny.edu

Summary: We developed a 3D game to teach and reinforce fundamental programming concepts. An innovative feature allows instructors to create customized programming challenges that students solve in the context of the game. The game was designed to engage and motivate all students, with a special focus on women and other underrepresented students.

1 Background

Majoring in computer science fields offers a number of benefits to students, including a growing and lucrative job market [30]. However, many students are unable to progress past the first introductory programming course. Introductory programming courses frequently suffer from high drop-out and failure rates of over 30% – 50% [5, 28, 16]. The attrition rate is often significantly higher for female students.

It is commonly observed that learning to program is difficult and that difficulty may account for the poor retention rates in the introductory class [12, 24, 17]. Our experience with our computing students is that learning to program demands significant practice time. Students learn how to program by programming, and the more students program, the easier it is for them to master the material and become successful in the course. However, many introductory courses do not include sufficient programming practice [4].

The literature on *serious games* discusses strategies to motivate students to practice programming. Researchers report that the long-standing formal lecture teaching mode is the least popular with students [27] and less effective than more interactive modes of learning [18]. *Serious games* describes those games that accomplish a goal in addition to entertainment. Research on *serious games for education* shows that, compared to traditional methods of instructor, game-based learning is superior at teaching subject matter [26, 15, 23, 31, 29] and increases both long-term retention and student motivation [26, 23, 13, 29]. Students who learn through games have increased feeling of alertness, activity, and involvement in contrast to standard lectures [11].

We therefore developed a serious educational game to help engage, motivate and improve the learning experience for students taking the introductory programming courses at our institutions and elsewhere. Our game, *Code Control*, is a game that helps reinforce programming concepts. An innovative feature is that instructors can create customized programming challenges that students solve in the context of the game. An automated compiler and testing environment gives students instant feedback, and completing successful programming challenges advances students in the gameplay.

2 Game Design

Code Control is a digital game developed using the Unity 3D platform, that teaches and assesses programming concepts. Instructors can optionally create customized programming challenges for their students as well as provide correct solutions. The programming challenges are then posed to the players in the context of the game.

While a number of serious games for computer science have been created, including some to teach introductory programming (e.g., [1, 2, 3, 8, 10, 9, 22, 19, 20, 21]), all of these games involve fixed programming challenges. In contrast, our approach allows instructors to specify their own challenges that best reinforce what is being taught in class. Additionally, many of these games (e.g. *Gidget* [19]) use a simplified programming language that was created for the game, and are designed to teach abstract programming concepts. In contrast, our game uses standard programming languages and is designed to augment existing programming courses. It is targeted to students who are learning the basics and want practice to solidify their skills.

The game's storyline involves a woman looking for endangered animals who are missing from an animal rescue. The digital name tags contain code that was broken by enemies. The player needs to find the lost roaming pets by solving the code to fix their name tags, to save the pets. The short programming challenges act as mini quizzes based on research of the "testing effect" that indicates that tests actually improve learning [6, 7, 14, 25].

The game currently has two levels, which take place in different areas in the city, to allow players to progress in solving the problems. In addition to solving programming challenges, players can explore the city, engage with non-player characters (NPCs) who offer advice and hints, and take part in recreational activities such as yoga on the street and hopping on the carousel in the park. A map of the area (situated in the bottom-right corner of the screen) allows the player to find their bearings and spot animals; a panel on the upper-left corner keeps track of the progress to date.

In designing our game, we consulted with students, both gamers and non-gamers, men and women, and varied races to appeal to a broad audience. It has been particularly designed for a female audience, featuring a (default) female lead character, avoiding violence and focusing on social goals. The avatar's appearance is fully customizable, including gender, skin color, and outfit, to give players co-ownership and to be able to represent a diverse group of players.

3 Programming Challenges

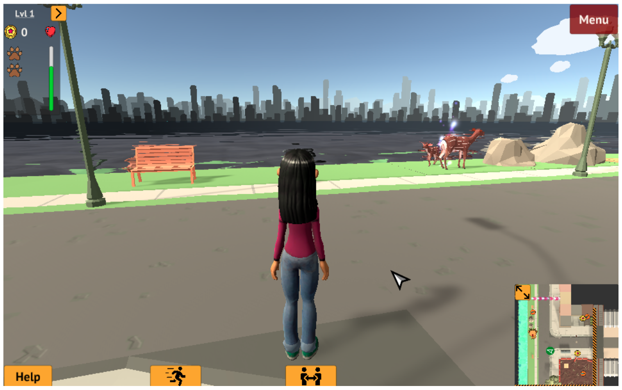
Instructors use our instructor portal to enter programming problems and their solutions. We use the Judge0 API[1], which is a free open-source API for code compilation and execution.

Code windows give players code snippets with missing sections to complete; once the players fill in the missing code, the code is compiled. The students' solutions are then checked for correctness by comparing them to the instructor's solution to see if the output matches. Compiler errors, execution errors, or positive feedback are are also provided. Customized scripts provide syntax highlighting. Judge0 supports 42 programming languages; so that any introductory language will most likely have support in our game. Furthermore, with Judge0 support for multi-file programs, *Code Control* can be used in coding courses at all levels.

CodeControl allows for several types of problems: coding, find and fix the error(s) and tracing problems (e.g. "What is the output of this code?").

A database connection allows us to collect detailed analytics of all users' gameplay (e.g. time spent per challenge, successes and mistakes made, levels completed, score, etc.) This provides a

[1] https://api.judge0.com/



(a) Player approaching an animal in Level 1

(b) Player exploring Level 2

(c) Player near the park in Level 2

(d) Customization of avatar

Figure 1: Gameplay pictures

wealth of information to instructors as to how this game is being played and what students are struggling with and succeeding at. Instructors who adopt this game as a course material are able to use the game as an informal evaluation mechanism to see which concepts their students are successful with and with which they need additional practice.

4 *Code Control:*

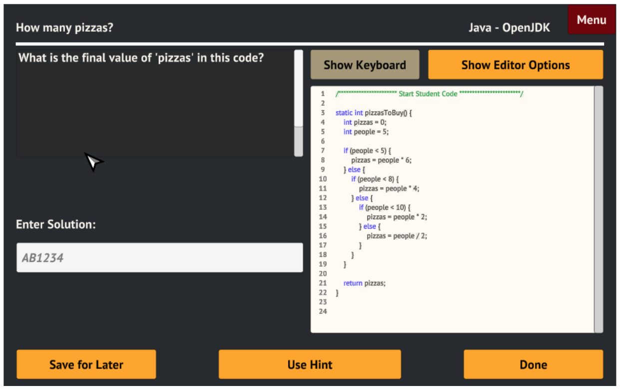
Pictures of *CodeControl* are given in Figures 1 and 2. Figure 1 depicts the gameplay; Figure 2 shows various types of problems as well as the instructor view of a problem.

Code Control is deployed as a WebGL and can be played in a browser without requiring installation. It is available at https://codecontrol.app/. If you are an instructor who is interested in using the game in your course, please contact us for a free instructor's account.

Future work on the game includes the development of more levels and integrating Parson's problems in addition to the coding and tracing problems. An evaluation of the success of *Code Control* in engaging and motivating the students is the topic of an ongoing Master's thesis project.

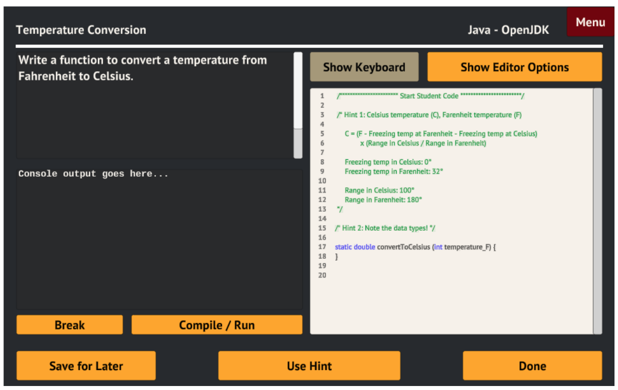
5 Conclusion

The SIGCSE Special Projects funding of $5,000 was used to support the work of an undergraduate student who developed the game.

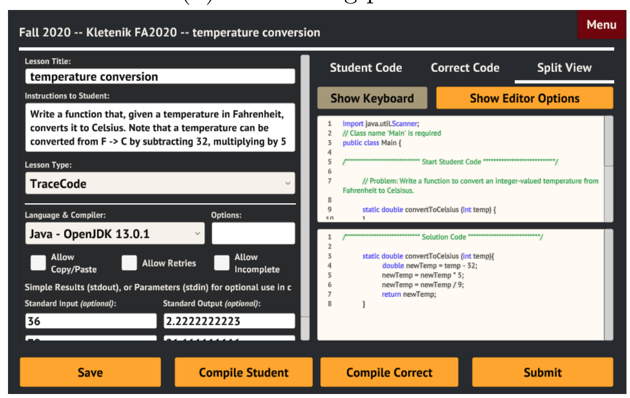


(a) Code tracing problem

(b) Code fixing problem



(c) Code writing problem



(d) Instructor view of a problem

Figure 2: Programming challenges

We are grateful to the SIGCSE Board for supporting our work. We anticipate *CodeControl* having an impact on students taking the introductory programming course at CUNY as well as at other institutions.

References

[1] Tiffany Barnes, Eve Powell, Amanda Chaffin, and Heather Lipford. Game2Learn: improving the motivation of CS1 students. In *Proceedings of the 3rd International Conference on Game Development in Computer Science Education*, pages 1–5. ACM, 2008.

[2] Tiffany Barnes, Heather Richter, Amanda Chaffin, Alex Godwin, Eve Powell, Tiffany Ralph, Paige Matthews, and Hyun Jordan. Game2learn: A study of games as tools for learning introductory programming concepts. *Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education*, 7, 2007.

[3] Tiffany Barnes, Heather Richter, Eve Powell, Amanda Chaffin, and Alex Godwin. Game2Learn: building CS1 learning games for retention. In *Proceedings of the 12th annual SIGCSE conference on Innovation and Technology in Computer Science Education*.

[4] Theresa Beaubouef and John Mason. Why the high attrition rate for computer science students: some thoughts and observations. *ACM SIGCSE Bulletin*, 37(2):103–106, 2005.

[5] Jens Bennedsen and Michael E. Caspersen. Failure rates in introductory programming. *SIGCSE Bull.*, 39(2):32–36, June 2007.

[6] Shana K Carpenter and Edward L DeLosh. Impoverished cue support enhances subsequent retention: Support for the elaborative retrieval explanation of the testing effect. *Memory & cognition*, 34(2):268–276, 2006.

[7] Shana K Carpenter, Harold Pashler, and Edward Vul. What types of learning are enhanced by a cued recall test? *Psychonomic Bulletin & Review*, 13(5):826–830, 2006.

[8] Michael Eagle and Tiffany Barnes. Wu's castle: teaching arrays and loops in a game. In *ACM SIGCSE Bulletin*, volume 40, pages 245–249. ACM, 2008.

[9] Michael Eagle and Tiffany Barnes. Evaluation of a game-based lab assignment. In *Proceedings of the 4th International Conference on Foundations of Digital Games*, pages 64–70. ACM, 2009.

[10] Michael Eagle and Tiffany Barnes. Experimental evaluation of an educational game for improved learning in introductory computing. In *ACM SIGCSE Bulletin*, volume 41, pages 321–325. ACM, 2009.

[11] Michael Grimley, Richard Green, Trond Nilsen, David Thompson, and Russell Tomes. Using computer games for instruction: The student experience. *Active Learning in Higher Education*, 12(1):45–56, 2011.

[12] Mark Guzdial. A biased attempt at measuring failure rates in introductory programming. https://computinged.wordpress.com/2014/09/30/a-biased-attempt-at-measuring-failure-rates-in-introductory-programming, 2014.

[13] Wen-Hao Huang. Evaluating learners' motivational and cognitive processing in an online game-based learning environment. *Computers in Human Behavior*, 27(2):694–704, 2011.

[14] Jeffrey D Karpicke and Henry L Roediger. Repeated retrieval during learning is the key to long-term retention. *Journal of Memory and Language*, 57(2):151–162, 2007.

[15] Mansureh Kebritchi, Atsusi Hirumi, and Haiyan Bai. The effects of modern mathematics computer games on mathematics achievement and class motivation. *Computers & education*, 55(2):427–443, 2010.

[16] Päivi Kinnunen and Lauri Malmi. Why students drop out CS1 course? In *Proceedings of the Second International Workshop on Computing Education Research*, ICER '06, 2006.

[17] Päivi Kinnunen and Lauri Malmi. Why students drop out cs1 course? In *Proceedings of the second international workshop on Computing education research*, pages 97–108. ACM, 2006.

[18] Jennifer K Knight and William B Wood. Teaching more by lecturing less. *Cell biology education*, 4(4):298–310, 2005.

[19] Michael J Lee and Andrew J Ko. Personifying programming tool feedback improves novice programmers' learning. In *Proceedings of the seventh international workshop on Computing education research*, pages 109–116. ACM, 2011.

[20] Michael J Lee and Andrew J Ko. Investigating the role of purposeful goals on novices' engagement in a programming game. In *2012 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pages 163–166. IEEE, 2012.

[21] Michael J Lee, Andrew J Ko, and Irwin Kwan. In-game assessments increase novice programmers' engagement and level completion speed. In *Proceedings of the ninth annual International ACM Conference on International Computing Education Research*, pages 153–160. ACM, 2013.

[22] Michael Jong Lee. *Teaching and Engaging with Debugging Puzzles.* PhD thesis, University of Washington, 2015.

[23] Tsung-Yu Liu and Yu-Ling Chu. Using ubiquitous games in an English listening and speaking course: Impact on learning outcomes and motivation. *Computers & Education*, 55(2):630–643, 2010.

[24] Andrew Luxton-Reilly. Learning to program is easy. In *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education*, ITiCSE '16, pages 284–289, New York, NY, USA, 2016. ACM.

[25] Mark A McDaniel, Janis L Anderson, Mary H Derbish, and Nova Morrisette. Testing the testing effect in the classroom. *European Journal of Cognitive Psychology*, 19(4-5):494–513, 2007.

[26] Marina Papastergiou. Digital game-based learning in high school computer science education: Impact on educational effectiveness and student motivation. *Computers & Education*, 52(1):1–12, 2009.

[27] Paul Sander, Keith Stevenson, Malcolm King, and David Coates. University students' expectations of teaching. *Studies in Higher education*, 25(3):309–323, 2000.

[28] Christopher Watson and Frederick W.B. Li. Failure rates in introductory programming revisited. In *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education*, ITiCSE '14, pages 39–44, 2014.

[29] Pieter Wouters, Christof Van Nimwegen, Herre Van Oostendorp, and Erik D Van Der Spek. A meta-analysis of the cognitive and motivational effects of serious games. *Journal of Educational Psychology*, 105(2):249–265, 2013.

[30] Yi Xue and Richard C Larson. STEM crisis or STEM surplus: Yes and yes. *Monthly Lab. Rev.*, 138:1, 2015.

[31] Ya-Ting Carolyn Yang. Building virtual cities, inspiring intelligent citizens: Digital games for developing students' problem solving and learning motivation. *Computers & Education*, 59(2):365–377, 2012.