No Lives Left: How Common Game Features Can Undermine Persistence, Challenge-Seeking, and Learning to Program

Laura J. Malkiewich, Alison Lee, Stefan Slater, Chenmu Xing, and Catherine C. Chase
ljm2172@tc.columbia.edu, alison.lee@tc.columbia.edu, slater.research@gmail.com,
xing@exchange.tc.columbia.edu, cc3663@tc.columbia.edu
Teachers College, Columbia University

Abstract: Persistence is necessary for learning, yet students have difficulty persisting at academic tasks, especially when faced with challenge. One context where students seem to persist is gameplay. This paper investigates whether an educational computer programming game can enhance student persistence and learning. Students were assigned to play either the Full version of a game, or a Minimal version of the game that ablated several common game features. Contrary to our hypotheses, we found that students who played the Full Game persisted less at challenging tasks, wrote less challenging code during gameplay, and were less willing to learn more about coding in the future. Ultimately, players of the Full Game also learned less than students who played the Minimal Game. Results suggest that certain design features of educational games can negatively impact a player's approach to challenging learning tasks, which in turn can negatively impact learning.

Keywords: games, computer programing, persistence, self-efficacy, challenge-seeking

Introduction

Persistence is critical for learning, yet students frequently fail to persist enough for real learning to occur, often giving up when academic tasks become too challenging (Chase, 2013; Diener & Dweck, 1978). Though students struggle to persist in academic contexts, they seem more willing to persist while playing video games. Children and adolescents spend an average of 90 mins/day playing video games (Rideout et al., 2010), and over half of adolescents find online games "addictive" (Yee, 2006). These statistics suggest that children are persistent during gameplay even though failure occurs often during gameplay, and people find failure uncomfortable (Juul, 2013). Furthermore, a study by Ventura, Shute and Zhao (2013) showed that adult subjects who reported playing video games frequently spent more time attempting to solve an impossible task. These results suggest that educational video games might be a context where students could develop persistent behaviors, which in turn could benefit learning. Still, while educational games might promote student persistence, there is little direct evidence that games enhance persistence during challenging tasks. Furthermore, little empirical research explains how games promote persistence more generally, making it difficult to ascertain what game-like elements, if any, might help students persist during or outside of gameplay.

One way in which games might promote player persistence is by increasing motivation. Many argue that games foster high intrinsic motivation or enjoyment (Habgood & Ainsworth, 2011; Lepper & Malone, 1987; Dickey, 2007; Fernandez, 2008), so children want to keep playing even when the game gets challenging. For instance, work on self-determination theory (Ryan & Deci, 2000) suggests that games promote intrinsic motivation because they satisfy needs for competency, relatedness, and autonomy through common game elements like the use of fantasy, structured rules and goals, sensory stimuli, challenge, mystery, and control (Ryan, Rigby, & Przybylski, 2006; Garris, Ahlers, & Driskell 2002; Birk & Mandryk, 2013). Other work suggests that games can launch players into a state of flow where they feel completely immersed in gameplay (Csikszentmihalyi, 1990), perhaps by providing clear goals, immediate feedback, and a sense of autonomy (Cowley, Charles, Black, & Hickey, 2008; Chen, 2007; Sweetser , Wyeth, 2005). However, these motivational theories focus mostly on how game elements make gameplay enjoyable. Persistence in the face of challenging tasks may be more related to students' perceptions of their own competence.

We take a novel viewpoint in this paper by proposing that games promote persistence by increasing self-efficacy (Bandura,1993; Ketelhut, 2007), a feeling of competence for a task that is enhanced by experiences of success and diminished by experiences of failure. While some findings are mixed (Schunk, 1989, 1991), many studies have demonstrated the link between high self-efficacy and persistence in problem solving (Relich, Debus, & Walker, 1986), writing (McCarthy, Meier, & Rinderer, 1985; Pajares & Johnson, 1994; Schunk, 2003), coping behavior (Bandura, 1977), and even the pursuit of scientific majors (Lent, Brown, & Larkin, 1984). However, the literature has yet to review how self-efficacy affects persistence during video gameplay. We argue that games

enhance self-efficacy by altering players' perceptions of success and failure. For instance, games may enhance self-efficacy by providing challenges that gradually increase in difficulty over time, in line with the player's ability and progress, providing learners with a series of more and more impressive successes (Przybylski, Rigby, and Ryan, 2010). Moreover, players may view failure in games as acceptable and normal since it is "just a game," whereas failure in school contexts where the stakes are higher can be discouraging (Litts & Ramirez, 2014). Finally, games provide functional feedback in the form of metrics like badges and points. Past work shows that performance feedback after success and failure can improve student self-efficacy (Bandura, 1993; Vallerand & Reid, 1984). Therefore, there seem to be many common game features that could boost players' self-efficacy for gameplay, which in turn could promote student persistence at game challenges. However, limited empirical research links self-efficacy to persistence in the context of an educational game.

By promoting persistence, games could lead students to learn deeply. While still a hotly contested topic (Hilton & Honey, 2011), several recent meta-analyses have concluded that educational games enhance learning over standard instructional activities (Wouters et al., 2013; Clark, Tanner-Smith, & May, 2013). Perhaps these games are more effective than traditional instruction because they promote students' self-efficacy at in-game learning tasks, which enhances persistence at challenging learning tasks and ultimately improves learning.

We believe that educational games are particularly well suited for teaching academic content that is highly iterative, where students face a lot of failure during the learning process and persistence is necessary for success. One such domain is computer programming, as coding requires students to effectively debug code that fails to function properly (Papert, 1980; Pea & Kurland, 1984b). The plethora of recent commercially developed games that aim to teach computer programming such as CodeSpark, Code Combat, Space Chem, and many others supports the notion that games might be a good avenue for teaching computer programming.

In this paper, we ask whether an educational game can enhance persistence at challenging, iterative learning tasks in comparison to a learning environment with few game features. Students played one of two versions of a game designed to teach computer programming: a Full Game and a Minimal Game (Figure 1). The Full Game was a commercially available educational puzzle game about computer programming with typical game features; the Minimal Game contained identical scaffolding and content but was stripped of many game features that we hypothesized would affect student self-efficacy and therefore student persistence in turn.

We hypothesized that the Full Game would promote more persistence, particularly in the face of challenge, which would lead to greater learning. We also predicted that Full Game players would have higher self-efficacy for coding and that self-efficacy would be associated with more persistence at a challenging task. Finally, we hypothesized that players of the Full Game would have greater intrinsic motivation for gameplay. Our results supported the link between persistence and learning but did not support the link between self-efficacy and those constructs. Results also suggested that some features of the Full Game were detrimental to both persistence and learning, perhaps by discouraging challenge-seeking behaviors.

Methods

Participants

Thirty-seven fifth grade students (54.1% female) were recruited from an urban charter school after-school program. Prior to the study, 89% of participants had no programming experience; the rest had less than a month's worth of programming experience.

Design

Students were randomly assigned to the Full Game condition (n=18) or the Minimal Game condition (n=19). Both games used block-based code, similar to Scratch (Maloney et al., 2010). In each game level, students solved problems by writing code to navigate an agent over obstacles and reach a goal. Both games contained identical problems and hints. However, the Full Game contained additional standard game features such as a narrative story, high quality graphics, and sound. Additionally, the Full Game highlighted successful or failed attempts at a game level by displaying messages like "OOPS" or "Missed It" after a failure or having the character smile or jump after a success. The Full Game also provided performance metrics in the form of points and stars at the end of each level. Students could earn one, two, or three stars depending on how many hints they asked for, and how many coding blocks they used (where fewer is better) to complete a problem. Finally, the Full Game provided purely fun "bonus" levels where students could collect jellybeans or shoot down asteroids instead of solving coding problems.

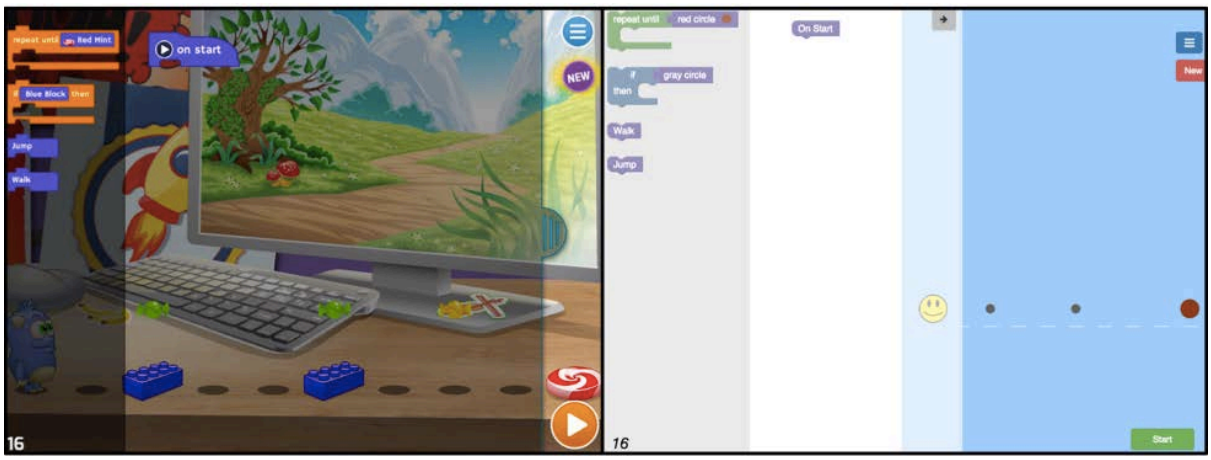

Figure 1. Screenshot of a typical level in the Full Game (left) and the same level in the Minimal Game (right).

Procedure

Each student participated in the study for five, 40-minute sessions: a pretest session, two gameplay sessions, a challenge session, and a posttest session. In the pretest session, students took a survey of their self-efficacy for coding followed by a brief paper pretest of their coding knowledge. In the two gameplay sessions, students played the game individually on iPads. In the next session, students attempted an impossible coding challenge (the challenge task). Finally, in the posttest session, all students were given a survey assessing their self-efficacy for coding and their intrinsic motivation for the game, followed by a paper posttest and a then a measure of persistence at future coding.

Measures

Prior programming knowledge and ability was measured by a paper-based pretest consisting of four test items on constructing code, interpreting code, and planning, a relevant skill to programming (Pea & Kurland, 1984a). Learning outcomes were measured by a 16-question paper-based posttest that assessed programming skills such as conceptualizing, writing, interpreting, and debugging code.

Persistence in the face of failure was measured by the time students voluntarily spent on the challenge task without giving up. The challenge task was an impossible, novel level embedded in the game the students were playing, and therefore had all the challenge level had all the same features of a typical level from each respective game (Full or Minimal). The challenge level was locked prior to the challenge task day, so students could not access the challenge level during the first two days of gameplay. Students were told that to complete the challenge level, they had to solve the challenge task using no more than a certain number of blocks. Unbeknownst to the students, this number of blocks was less than the minimum required to successfully solve the problem. As a result, students experienced failure at every attempt; even when students thought that they solved the problem, they were told by the researchers that they used too many blocks in their solution and they could try again. At any time, students were given the option to continue trying to solve the challenge level (for up to 40 minutes), or quit to explore computer-based science simulations.

Students' self-efficacy for programming was measured by a 7-point Likert-scale survey before and after gameplay, in which students rated their confidence in their ability to write, interpret, debug, and choose correct code. Students were asked questions like, "If something is wrong with your code, how confident are you that you can fix it?". Student intrinsic motivation for the game was also measured using a 7-point Likert-scale survey that asked students how interesting, exciting, enjoyable, and fun they found the game in general, across the two days of gameplay and the challenge level. To measure persistence in future coding, students were given the option of taking a handout that referred them to various websites where they could learn more about coding.

Gameplay data was collected using screen recordings and embedded log capture. Due to technical difficulties, some video and log data was lost, so all measures of student coding behaviors and time spent coding only involve a portion of the total study sample (n=9 for each group). We did an exploratory analysis of the gameplay data to find in-game behaviors that indicated students' approach and response to challenge during gameplay. This analysis helped us identify one measure of in-game persistence, and one measure of challenge-seeking during gameplay. In-game persistence was measured by time spent playing coding levels vs. bonus levels. Challenge-seeking during gameplay was measured by how often students chose to engage in challenging learning behaviors by contrasting the percentage of levels that students completed with either basic,

intermediate, or complex code. Students had access to two types of coding blocks: "action" blocks make the character walk or jump, while "parameter" blocks include repeating loops and if/then conditional commands that need to be filled with action blocks in order to work. Codes that only used action blocks were labeled "basic", codes that implemented rote usage of parameter blocks were labeled "intermediate", and codes that adapted parameter blocks to suit the given problem were labeled "complex". We coded each game level's difficulty by indicating how parameter or action blocks could be used in that level. Easy levels required only using action blocks, medium levels allowed students to use action blocks and one parameter block, and hard levels allowed students to use action blocks and multiple parameter blocks.

Findings

Learning outcomes

Students in both conditions started with equivalent prior knowledge, and learned from pretest to posttest, but the Full condition acquired less programming knowledge from gameplay. T-tests found no significant difference between conditions on the coding skills pretest, $t(35) = 0.12$, $p = .91$, or mean GPA scores, $t(35) = 1.63$, $p = .11$. An ANCOVA with pretest as covariate revealed that students in the Full condition performed worse on the posttest than the Minimal condition, $F(1,34) = 6.50$, $p = .02$, $\eta_p^2 = 0.16$. When controlling for prior knowledge, Full Game students scored 22% lower than Minimal Game students at posttest (Table 1). These results imply that, contrary to our hypotheses, students who played the Full Game learned less than students who played the Minimal Game.

Persistence measures

Also contrary to our hypothesis, measures of persistence indicated that students who played the Full Game were less persistent at coding tasks.

On the challenge task, where students were given an impossible coding problem that they could quit at any time, students who played the Full Game persisted about 8 mins or 30% less than those who played the Minimal Game (Table 1), $t(35) = 2.61$, $p = .01$; $d = 0.88$. So students in the Full condition gave up sooner on a hard challenge set within the context of the game than students who were in the Minimal condition.

Table 1: Student learning and persistence scores

*_Significant difference in scores, $p < .05$._ Challenge times are in minutes.

		Pretest		Posttest		Challenge Time	
Game	*n*	*M*	*SE*	*M*	*SE*	*M*	*SE*
Full	18	1.97	0.18	7.58*	0.61	19.08*	2.31
Minimal	19	1.99	0.15	9.76	0.60	27.01	1.98

While persistence on the challenge task shows students' willingness to persist within their respective game environments, we also wanted to assess whether students would persist at coding in general, beyond the confines of the study. We found that students who played the Full Game were less likely to want to pursue future coding activities. A chi-square test demonstrated that fewer students in the Full Game condition voluntarily took a handout explaining where they could learn more about coding, $X^2(1, N = 37) = 4.56$, $p = .03$.

Students in the Full Game also failed to persist at coding during gameplay. Students who played the Full Game spent more time playing bonus levels and less time on coding levels during the second gameplay session. A repeated measures ANOVA of time on coding levels showed that there was a significant interaction between condition and gameplay session, $F(1,16) = 5.18$, $p = .04$. On the first day of gameplay, students across both conditions spent the same amount of time playing coding levels ($M_{Full} = 35.47$, $SE_{Full} = 0.90$; $M_{Minimal} = 34.95$, $SE_{Minimal} = 0.90$) even though students in the Full version of the game had access to bonus levels with no learning content. However, during the second play session, students who played the Full Game spent less time on coding levels than students who played the Minimal Game ($M_{Full} = 26.95$, $SE_{Full} = 2.11$; $M_{Minimal} = 33.90$, $SE_{Minimal} = 2.11$), $t(16) = 2.33$, $p = .03$, $d = 1.17$. This implies that by the second day of gameplay, when the novelty of the game had worn off, students who played the Full Game persisted less at coding and instead spent more time on non-coding, bonus levels.

Students in the Full condition were also less likely to engage in challenging coding behaviors, particularly in challenging situations. For instance, students in the Full condition tended to shy away from writing

complex code on hard problems. A repeated measures ANOVA showed that condition had a significant effect on the level of complexity of the code that students used in their first attempt to solve hard problems, $F(2,32) = 4.29$, $p = .02$, $\eta_p^2 = 0.21$. Specifically, students who played the Full Game were more likely to use intermediate code in their first attempt at a hard problem, $t(16) = 2.23$, $p = .04$, $d = 1.12$, and less likely to use complex code in their first attempt at a hard problem, $t(16) = 2.45$ $p = .03$, $d = 1.23$. These differences were significant after applying Scheffé's method for alpha correction for family-wise Type 1 error rate control. The same pattern appeared when we looked at the type of code students used to complete a problem. A repeated measures ANOVA showed a significant interaction between condition, coding complexity (basic, intermediate, complex), and problem difficulty (medium vs. hard) on percentage of problems completed $F(2,32) = 10.50$, $p < .001$, $\eta_p^2 = 0.40$. Relative to the Minimal condition, the Full condition solved a larger percentage of hard problems with intermediate code, $t(16) = 2.66$, $p = .02$, $d = 1.33$, and a smaller percentage of hard problems with complex code, $t(16) = -3.22$, $p = .01$, $d = 1.61$ (Table 2). Again, these differences were significant after applying Scheffé's method for alpha correction for family-wise Type 1 error rate control. These results indicate that students who played the Full Game were less willing to take risks in challenging situations; they chose to attempt less complicated code when first trying to solve hard problems and when completing them.

Table 2: Percent of levels completed with various sophistications of code

	Medium Levels		Hard Levels	
	Minimal Game	Full Game	Minimal Game	Full Game
Complex	15%	22%	34%*	9%
Intermediate	18%	16%	31%*	46%
Basic	67%	62%	36%	45%

*_Significant difference in scores, $p < .05$_. Note: all easy levels are solved with basic code.

To test the relationship between persistence, risk taking, and learning, we explored the association between persistence measures and learning outcomes. A regression of persistence times on post-test scores, controlling for condition and pretest score, revealed that persistence on the challenge task trended towards predicting learning, $\beta = 0.46$, $t(32) = 1.952$, $p = .06$. The model predicted a significant amount of the variance in posttest scores $R^2_{adj} = 0.26$, $F(4,32) = 4.15$, $p = .01$. There was no interaction between challenge time and condition and there was no main effect of condition. So regardless of condition, the more students persisted on the challenge task, the better they did on the posttest. Challenge time and percent of hard levels completed with complex code were highly correlated $r(16) = 0.54$, $p = .02$, but willingness to take risks and use complex code on hard problems was more predictive of learning. A linear regression controlling for condition and pretest score found that percent of hard levels solved with complex code predicted a significant amount of the variance in posttest scores $R^2_{adj} = 0.53$, $F(4,13) = 5.84$, $p < .01$. Again, there was no interaction with condition; regardless of condition, students who used challenging code to solve hard problems did better on the posttest. Specifically, every 10% increase in the proportion of hard levels students completed with complex code was associated with a 0.87 point increase in posttest score, $t(13) = 3.27$, $p < .01$. In other words, the more students attempted challenging coding behaviors on hard levels, the more they learned.

Challenge-seeking behaviors also predicted student persistence on the challenge task. A linear regression showed that the percent of hard problems that students completed with complex code predicted the amount of time students spent on the challenge task before giving up, $\beta = 0.54$, $t(16) = 2.59$, $p = .02$. Percent of hard problems completed with complex code also predicted a significant amount of the variance in challenge time $R^2_{adj} = 0.25$, $F(1,16) = 6.731$, $p = .02$. So students who risked getting hard problems wrong by using more complex code, also spent more time persisting at the challenge task. One possible explanation for this finding is that students who engaged in more complex coding during the game persisted longer on the challenge task, because they had developed more strategies (both intermediate and complex codes) to attempt before giving up on the challenge task. In other words, we suspect that the effect of persistence on posttest scores was mediated by challenge-seeking behavior, however we did not have the statistical power to detect these affects, since we only had in-game behavior data for half of our sample (n=9 per condition). Another possibility is that the features of the Full Game are having two independent effects by leading players to engage in both less challenge-seeking and less persistence in the face of challenge, which both negatively affect learning.

Self-efficacy and intrinsic motivation

Conditions did not differ in their coding self-efficacy before game play, $t(35) = 0.939$, $p = .35$. A repeated measures ANOVA on pre and post self-efficacy ratings found a main effect of time $F(1,35)= 84.01$, $p < .01$, $\eta_p^2= 0.71$ and a significant interaction $F(1,35)= 4.30$, $p = .05$, $\eta_p^2= 0.11$. This suggests that while students in both conditions gained self-efficacy from pre to post, the Full condition made significantly smaller gains (Pre-Post Gains: $M_{Full} = 1.90$, $SE_{Full} = 0.40$; $M_{Minimal} = 3.01$, $SE_{Minimal} = 0.36$). Contrary to our hypothesis, self-efficacy was not correlated with persistence measures or posttest scores, p's > .09.

There was no significant difference between conditions in their intrinsic motivation for playing the game. Both conditions found the game equally enjoyable, giving fairly high average ratings on the 7-point intrinsic motivation survey $t(35) = 1.06$, $p = .30$, ($M_{Full} = 5.25$, $SE_{Full} = 0.35$; $M_{Minimal} = 5.75$, $SE_{Minimal} = 0.32$). Intrinsic motivation was not related to persistence measures or posttest scores, p's > .15.

Conclusions and implications

Contrary to our hypotheses, we found that students who played the Full version of a coding game, with many standard game features, showed less persistence at challenging coding tasks compared to students who played a Minimal version of the game, with many game features removed. When given an impossible coding problem within the context of the game environment, students in the Full condition gave up more quickly. Fewer students in the Full condition wanted to pursue future coding after the study ended. During the second gameplay session, after the novelty of the game had worn off, students who played the Full Game were more likely to spend time playing content-free "bonus" levels and less time playing coding levels. In addition to demonstrating less persistence at challenging tasks, exploratory analyses revealed that students who played the Full Game chose to engage in less challenging coding behaviors by using less complex code on hard levels, where the risk of failure was great. This suggests that players of the Full Game engaged in fewer risk-taking behaviors. It is interesting to note that while we had originally focused on persistence at challenging tasks as a key learning behavior, we found that the Full Game discouraged risky learning behaviors during challenging tasks. Overall, this suggests that players of the Full Game had a less productive approach to challenge − they chose to engage in risky coding behaviors less frequently and persisted less.

The Full Game group's less productive approach to challenge may explain why they learned less. Use of complex code on hard problems significantly predicted learning and persistence in the challenge task marginally predicted learning. This was true regardless of condition. So students who had a productive approach to challenge learned more from the game. While the relationship between productive approaches to challenge and learning outcomes is correlational (not causal), it provides a likely explanation for why the Full condition learned less from the game. The Full Game could have hindered learning by hindering a players' use of complex code on hard problems and a player's willingness to persist at the challenge task, both of which seem to be independently pathways for learning. Given the correlational relationship between challenging coding behaviors and persistence on the challenge task, another possibility is that learning more complex code during gameplay enabled learners to persist longer in the challenge task by giving students a larger repertoire of coding behaviors to draw from. So students who were risk-taking and tried using more complex code to solve hard problems during the game might have learned more coding techniques to try while attempting to complete the challenge task. In this case, use of complex code during the game could have mediated the relationship between persistence during the challenge task and learning. We unfortunately do not have enough power from this study to test these two theories.

What caused the Full condition to show a less productive approach to challenge by persisting less at challenging tasks and not choosing to engage in writing complex code on challenging problems? The main mechanism we hypothesized, self-efficacy, does not seem to explain this phenomenon. Although the Full Game students gained less self-efficacy for coding by playing the game, self-efficacy was not predictive of persistent behaviors or challenge-seeking coding behaviors. These findings suggest that other motivational mechanisms were affected by the Full Game, which in turn discouraged challenge-seeking and persistence. One possibility is that the Full Game was not engaging for students. However, both conditions rated their intrinsic motivation for the game equally highly. Another possible explanation is that the Minimal Game may have felt less "game-like" and more like a serious learning environment, causing the Minimal condition students to persist more at learning-relevant behaviors. However, when asked to rate how much the learning environment "felt like a game," both conditions gave equally high ratings.

A more plausible explanation for the Full condition's less productive approach to challenge is that the Full Game made failure more salient. When players failed in the Full Game, a huge "OOPS" pulsed on screen. At the end of each level, star counts and points indicated whether the level was solved with the optimal code, further marking failure when the students' code was non-optimal. The Minimal condition did not contain these standard

game features. While meant to provide clear feedback and encourage revision, these features may have inadvertently intensified perceived student failure. We are currently running a second study where we are manipulating failure feedback in the game to test how salience of failure impacts persistence and learning. Even though in the first study the rate of failure was equivalent across conditions, if the Full condition felt more affected by failure, then it stands to reason that they would both persist less at challenging tasks and choose to engage less in challenging coding behaviors, where the risk of failure is great. In fact, the Full condition did show lower gains in self-efficacy in comparison to the Minimal condition, which would be a fitting outcome since self-efficacy is primarily boosted by feelings of success. Though we did not find a significant relationship between self-efficacy and persistence, it is possible that more proximal measures of self-efficacy (e.g. given immediately after failure in the game) would better predict persistence.

Finally, the Full condition may have shown a less productive approach to challenge because the Full Game's feedback and reward system dis-incentivized students from tinkering. Feedback and reward for students was operationalized by their star count, shown at the end of each level and on the main level screen so that students could see their performance on completed levels. Even if students solved a level with the optimal code, their star count would decrease if they took too many attempts while trying to solve a level. This reward mechanism could have discouraged students from trying more complex code, as using more complicated code makes it harder to get a problem right on the first try. This could explain why students who played the Full Game used less complex code when beginning and solving hard problems, while students who played a game without this reward system were more willing to attempt using complex code on hard levels. This may also explain why Full Game players were less persistent at the challenge task. If students playing the Full Game were discouraged from tinkering, they may have developed fewer coding strategies to try on the challenge task, causing them to run out of ideas and quit sooner.

We do not mean to imply that all games are bad for persistence and learning. Obviously, this study is limited in that it only investigated a single game that encapsulated multiple standard game features. Rather, we interpret these results to mean that *some* commonly-used game design elements *can be* implemented in such a way that they negatively affect persistence and challenge-seeking behaviors, which can in turn hindering learning. Future work can focus on isolating which game features discourage productive learning behaviors, and under what conditions.

References

Bandura, A. (1977). Self-efficacy: toward a unifying theory of behavioral change. *Psychological Review*, *84*(2), 191-215.

Bandura, A. (1993). Perceived self-efficacy in cognitive development and functioning. *Educational Psychologist*, *28*(2), 117-148.

Birk, M., & Mandryk, R. L. (2013, April). Control your game-self: effects of controller type on enjoyment, motivation, and personality in game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 685-694). ACM.

Chase, C.C. (2013). Motivating expertise: Equipping novices with the motivational tools to move beyond failure. In J.J. Staszewski (Ed.), *Expertise and skill acquisition: The impact of William G. Chase*. New York: Psychology Press.

Chen, J. (2007). Flow in games (and everything else). *Communications of the ACM*, *50*(4), 31-34.

Clark, D. B., Tanner-Smith, E. E., & May, S. K. (2013). *Digital Games for Learning: A Systematic Review and Meta-Analysis (Executive Summary)*. Menlo Park, CA.

Cowley, B., Charles, D., Black, M., & Hickey, R. (2008). Toward an understanding of flow in video games. *Computers in Entertainment (CIE)*, *6*(2), 1-28.

Csikszentmihalyi, M. (1990). *Flow: The psychology of optimal performance*. New York: Cambridge University Press.

Dickey, M. D. (2007). Game design and learning: A conjectural analysis of how massively multiple online role-playing games (MMORPGs) foster intrinsic motivation. *Educational Technology Research and Development*, *55*(3), 253-273.

Diener, C. L, & Dweck, C. S. (1978). An analysis of learned helplessness: Continuous changes in performance, strategy, and achievement cognitions following failure. *Journal of Personality and Social Psychology*, *36*, 451-462.

Fernandez, A. (2008). Fun Experience with Digital Games: A Model Proposition. In O. Leino, H. Wirman & A. Fernandez (Eds.), *Extending Experiences: Structure, Analysis and Design of Computer Game Player Experience* (pp. 181-190). Rovaniemi, Finland: Lapland University Press.

Garris, R., Ahlers, R., & Driskell, J. E. (2002). Games, motivation, and learning: A research and practice model. *Simulation & gaming*, *33*(4), 441-467.
Habgood, M. J., & Ainsworth, S. E. (2011). Motivating children to learn effectively: Exploring the value of intrinsic integration in educational games. *The Journal of the Learning Sciences*, *20*(2), 169-206.
Hilton, M., & Honey, M. A. (Eds.). (2011). *Learning science through computer games and simulations*. Washington: National Academies Press.
Hsu, C. L., & Lu, H. P. (2004). Why do people play on-line games? An extended TAM with social influences and flow experience. *Information & Management*,*41*(7), 853-868.
Juul, J. (2013). *The art of failure: An essay on the pain of playing video games*. Cambridge, MA: MIT Press.
Ketelhut, D. J. (2007). The impact of student self-efficacy on scientific inquiry skills: An exploratory investigation in River City, a multi-user virtual environment. *Journal of Science Education and Technology*, *16*(1), 99-111.
Lepper, M. R., & Malone, Th. W. (1987). Intrinsic motivation and instructional effectiveness in computer-based education. In R. E. Snow & M. J. Farr (Eds.), *Aptitude, learning, and instruction: Vol. 3. Conative and affective process analyses* (pp. 255-286). Hillsdale, NJ: Lawrence Erlbaum.
Litts, B., Ramirez, D. (2014) Making people fail: Failing to learn through games and making. Proceedings of the Games Learning Society (GLS) Conference. Madison, Wisconsin.
Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The scratch programming language and environment. *ACM Transactions on Computing Education (TOCE)*, *10*(4), 16:1-16:15.
Pajares, F., & Johnson, M. J. (1994). Confidence and competence in writing: The role of self-efficacy, outcome expectancy, and apprehension. *Research in the Teaching of English*, *28,* 316-334.
Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books, Inc.
Pea, R. D., & Kurland, D. M. (1984a). *Logo programming and the development of planning skills*. (Technical Report No. 16). New York: Columbia University.
Pea, R. D., & Kurland, D. M. (1984b). On the cognitive effects of learning computer programming. *New Ideas in Psychology*, *2*(2), 137-168.
Przybylski, A. K., Rigby, C. S., & Ryan, R. M. (2010). A motivational model of video game engagement. *Review of General Psychology*, *14*(2), 154-166.
Relich, J. D., Debus, R. L., & Walker, R. (1986). The mediating role of attribution and self-efficacy variables for treatment effects on achievement outcomes. *Contemporary Educational Psychology*, *11*(3), 195-216.
Rideout, V. J., Foehr, U. G., & Roberts, D. F. (2010). Generation M^2: Media in the Lives of 8-to 18-Year-Olds. *Henry J. Kaiser Family Foundation*.
Ryan, R. M., & Deci, E. L. (2000). Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *American psychologist*, *55*(1), 68-78.
Ryan, R. M., Rigby, C. S., & Przybylski, A. (2006). The motivational pull of video games: A self-determination theory approach. *Motivation and emotion*, *30*(4), 344-360.
Schunk, D. H. (1989). Self-efficacy and achievement behaviors. *Educational Psychology Review*, *1*(3), 173-208.
Schunk, D. H. (1991). Self-efficacy and academic motivation. *Educational Psychologist*, *26*(3-4), 207-231.
Schunk, D. H. (2003). Self-efficacy for reading and writing: Influence of modeling, goal setting, and self-evaluation. *Reading &Writing Quarterly*, *19*(2), 159-172.
Sweetser, P., & Wyeth, P. (2005). GameFlow: a model for evaluating player enjoyment in games. *Computers in Entertainment (CIE)*, *3*(3).
Vallerand, R. J., & Reid, G. (1984). On the causal effects of perceived competence on intrinsic motivation: A test of cognitive evaluation theory.*Journal of Sport Psychology*, *6*(1), 94-102.
Ventura, M., Shute, V., & Zhao, W. (2013). The relationship between video game use and a performance-based measure of persistence. *Computers & Education*, *60*(1), 52-58.
Voiskounsky, A. E., Mitina, O. V., & Avetisova, A. A. (2004). Playing Online Games: Flow Experience. *PsychNology Journal*, *2*(3), 259-281.
Wouters, P., Van Nimwegen, C., Van Oostendorp, H., & Van Der Spek, E. D. (2013). A meta-analysis of the cognitive and motivational effects of serious games. *Journal of Educational Psychology*, *105*(2), 249-265.
Yee, N. (2006). Motivations for play in online games. *CyberPsychology & Behavior*, *9*(6), 772-775.

Acknowledgments

We thank participating teachers and students. We would also like to thank Joan Cejudo for his programming work on this project.