# Using the "Daily Missions" Gamification Strategy To Promote Incremental Progress on Programming Assignments

Stephen H. Edwards
Virgina Tech
edwards@cs.vt.edu

Andrew B. Goldman
Virginia Tech
abg@vt.edu

## Abstract

*Automatic assessment tools are increasingly utilized in undergraduate programming courses to evaluate software solutions, streamlining the grading process for both students and professors. In spite of their benefits of speed and convenience, such online systems for providing instant feedback have the tendency to draw attention to performance-based outcomes while failing to reliably recognize the effort and hard work a student puts into a solution. For the many struggling students who are new to programming, this type of objective feedback can be discouraging and may decrease their motivation to stay engaged towards success. To address this issue, this paper explores strategies for more effectively recognizing student progress on programming assignments and identifying small tasks for students to complete that will steer them in the right direction. Further, this paper presents the design of a "Daily Missions" gamification strategy that was added to an automated program grading system. A pilot study shows feasibility of the approach and suitability of the design, indicating a statistically significant increase in the rate of completing tasks assigned as daily missions, and that students believe daily missions reinforce good practices while giving them better ideas on how to improve their work. An evaluation of experiences from using the strategy in the classroom is presented, along with student perceptions from using the system.*

## 1. Introduction

The use of automated systems for grading student work in computing and information technology classrooms has become increasingly common as class sizes have risen. These tools offer many benefits, including more repeatable, objective assessment and near-immediate feedback. Most importantly, they allow students to apply many cycles of obtaining feedback and making revisions for improvement before completing their work. At the same time, however, these tools simultaneously carry potential disadvantages, particularly in large classrooms where students may pursue their work most often in isolation. Current automated grading tools focus on assessing the quality of the end product (typically the student's program), and draw attention to performance-based concerns relevant in the final product. As a result, early in a student's development process, they may see zero or near-zero scores for incomplete work, and feedback may be viewed more negatively as criticism, rather than as helpful advice. Students who experience a series of low-or-no-scoring submissions may quickly become discouraged, even if such scores are typical when a student is early in the development process, only achieving high scores when near completion.

While many gamification strategies, including badging, leveling, and experience points, have been used in educational contexts, this paper reports on experiences with a less common gamification strategy to student assignments: *daily missions*. This mechanism is inspired by commercially successful video games, including many popular mobile games. Here, a *mission* is a specific task that a participant can complete to earn a stated reward. We focus on tasks related to making progress on a student's assignment. Importantly, these tasks, while measurable and while associated with making forward progress toward a solution, are not actions that would necessarily result in a change in the student's score—instead, the focus is on making progress in small increments that reinforce good work habits. These missions are *daily* because they are assigned once each 24 hours and students have one day to complete them. A set of *daily missions* can then be used to clearly define a set of incremental subgoals that are relevant to the process of completing an assignment, even if the student may not be far along enough to earn a significant score on their partial work.

There are two purposes behind daily missions. First, these missions are intended to help motivate students to stay engaged and mitigate the negative impacts of objective, performance-based feedback, particularly

URI: https://hdl.handle.net/10125/63892
978-0-9981331-3-3

for beginners who are just starting out. Just as in video games, by providing a specific list of daily tasks that are easy to achieve, we can instill a sense of forward progress, while also providing a sense of accomplishment as tasks are performed. When task achievement is recognized, we also communicate value for the process students are following and the effort they are investing, even if their solution is not yet complete. Second, these missions reinforce good practices. The tasks follow recommended practices for code development, self-checking, and design that have been shown successful through research.

Here we describe the design of a daily missions mechanism for an automated grading tool that processes programming assignments, and report on our experiences with it in the classroom. The approach offers a series of 5 small challenges to each student every 24 hours, where these challenges consist of manageable actions that carry the student's work forward and that may help improve the student's solution. We describe the design strategy used to create these missions, and the user interface choices made in presenting them to students. We report on performance data collected from using this approach in a live classroom, and also summarize the results of student perceptions of the mechanism collected via survey.

## 2. Related Work

The approach of using daily missions, while inspired by video games, is also related to goal-setting theory and social learning theory. It is implemented in this context using an automated program grading tool, and is built on prior work in measuring productive effort that students invest in their assignments.

### 2.1. Gamification Approaches

Gamification refers to giving game-like attributes to a non-game entity [1], which can engage users beyond the instrumental activity of the task [2]. In addition to the potential for increased motivation, gamification is advantageous because many students are already familiar with gameplay elements from their years of using video games as teenagers. The enjoyment that young people derive from video games makes a strong case for the theory that gamifying educational tools will make learning more fun and engaging. The other answer lies primarily in psychology. User experience designs are increasingly placing emphasis on elements that target human emotions [3], and gamification in particular appeals to the natural human desires of competition, status, and achievement [4]. In the context of tools that facilitate virtual education and assessment, gamifying certain features aims to incentivize progress by stimulating these intrinsic desires.

To measure the effectiveness of gamification techniques on student motivation and engagement in learning, Poondej and Lerdpornkulrat conducted an experiment with undergraduate students in an information literacy skills course [5]. They compared a non-gamified (control) group with a gamified (treatment) group, the latter of which used an online learning management tool called CourseSites that implemented a system of points, levels, achievement badges, and leaderboards. Each CourseSites activity was set up as a mission that students could complete to earn experience points and increase their rank, and along the way students could see the achievement points they've earned so far and what is required to receive additional rewards. Results indicated that students in the treatment group had a markedly higher engagement in learning than students in the control group.

A similar study was conducted by Kaila et. al [6] where game-like features were implemented in a university-level online programming course. Their design including breaking the assignment into a list of smaller tasks to make it more manageable, using a progress bar to give a sense of how far the student has advanced through the assignment, and using virtual trophies along the bar to communicate a feeling of achievement. The core idea is to make users feel like they are on a journey with many steps along the way, and that they are in control of their own destiny [7].

Tracking progress is a common utility provided by gamified tools, enabling student **self-monitoring**, where a student recognizes their own growth and builds confidence when they see how far they have come [8]. This is an important part of self-regulated learning [9]. This process is characterized by a cycle in which students monitor the effectiveness of their work methods and react to feedback accordingly [10]. From this pattern, students attain a sense of self-actualization that motivates their willingness to learn.

Games also challenge players to achieve goals and outcomes [1]. In gamified educational tools, **rising to challenges** can drive participation among students and stimulate their problem-solving capacities. There is often a delicate balance between challenges that are too hard and challenges that are too simple in order to most effectively motivate students to stay engaged. With an appropriate balance, challenges combined with gameplay can encourage students to practice skills more frequently and view obstacles as opportunities to improve rather than deterrents to their self-esteem.

**Persistence** is another benefit of gamification, which can forge a sense of perseverance in the face of difficulty.

In contrast to traditional learning environments where students can lose motivation easily, gameplay elements reduce the level of discouragement a user may feel when failing to complete a task. Instead, they draw attention to the user's progress toward reaching the next level or mission based on tasks that have been completed [11]. Mistakes are offloaded to the process of play in a way that "minimizes personal association with failure and encourages students to strive for mastery" [9].

Finally, applying rewards to increase student motivation has been widely researched in areas including psychology, education, and video games [12, 13]. Reward strategies of many different kinds are widely used in video games as a result. David explains a number of game reward systems [14]. Wang et al. [15] gave an overview about how reward systems give positive experiences to players in various video games. They recommend that players can have fun with both rewards and reward mechanisms. Rewards and reward mechanisms in video games foster players' intrinsic motivation with sense of fun and pleasure.

## 2.2. Goal-Setting Theory

The idea of daily missions in video and mobile games was inspired by existing practice rather than research [14], where it has been proven effective through use. However, its use in this study is related to both goal-setting theory and social learning theory.

Goal-setting theory explains how the act of setting goals can affect one's future performance [16, **?**]. Edwin Locke proposed five basic principles of goal-setting in this theory: clarity, challenge, commitment, feedback, and task complexity. This theory has been used in other educational games [17], and daily missions here relate to several (but not all) aspects of this theory. As explained in Section 3, daily missions are designed to be clear and measurable, and are implemented in a system that provides instant feedback. They are also selected to be elements of productive behaviors that are still small enough to be accomplished within the one-day window allowed. The mechanism used to present the missions to students is intended to invite their commitment to complete (at least some of) the missions. However, students do not set the missions for themselves—they are presented with a pre-selected set of missions based on their current progress, and only have the agency to select which missions they pursue. Still, multiple principles from goal-setting theory are present in daily missions, although the approach described here may not realize all of the benefits ascribed to goal-setting by the theory.

At the same time, daily missions are intended to serve as a *model* for actions a student can take to make progress on their work in a variety of contexts. Thus, in addition to aiming for some of the benefits of goal-setting, daily missions also incorporate some aspects of social learning theory [18], which is based on the idea that people learn by observing and imitating others, as well as through modeling. Here, daily missions serve as an explicit model of how to set subgoals for how to invest effort on an assignment. Effectively, the daily missions provide a specific model that, through repetition, students may begin to imitate. Indeed, many of the missions are intended to help provide guidance to students that can be applied to improve and further the student's work, even if the student happens to be stuck on a conceptual aspect of an assignment. While these missions are not being performed by another person, they still may serve as a model for students to imitate, by repetitively showing small daily tasks that are held up as (micro-)goals of achievement.

## 2.3. Automated Assessment of Student Programs

The daily missions described here are implemented in an automated grading tool used to assess programming assignments. Ihantola et al [19] provide a comprehensive overview of approaches to automatically grading computer programming assignments. While such systems streamline a number of assessment tasks that would otherwise be manual, focus among these tools has been increasingly placed on going beyond a simple grading system and creating a platform that guides student learning while offering direction and support to those who struggle [19]. Strategies geared toward this objective include identifying subtasks for students to complete, implementing features that raise motivation, and providing incentives for students to engage in practices that increase programming success. Most current assessment tools lack these characteristics, which is why the chief goal of evolving systems is to foster progress in work habits and programming ability.

One of the biggest challenges in making assessment tools more useful for students is being able to generate feedback that not only measures performance but that also measures effort. Assessment approaches in computing are often characterized by points-driven scoring that is easy to be taken in a negative tone due to its focus on mistakes that were made and criteria that are not met [20].

### 2.4. Progress Indicators

To enable daily missions that are driven by appropriate measures of student effort and progress, this paper is based on prior work to address this problem. Edwards and Li [20] developed a series of progress indicators to measure the productive effort that students are expending while they work towards a solution. The goal of these indicators is to recognize many types of advancement in addition to just performance outcomes. While individual measures are noisy and subject to both false positives and false negatives, they propose using triangulation to combine multiple measures, together with the use of rolling averages to identify trends, in order to address these limitations. They divide their measures into 8 that are applicable to any kind of programming project and focus on writing a solution, plus an additional 7 that focus on self-checking practices that are appropriate in assignments where students are required to write software tests to check their own solutions. These measures were implemented for a popular open-source automated grading platform, and so were chosen as one of the building blocks for our daily missions strategy.

## 3. Designing Daily Missions

The concept of daily missions began with the idea of offering successive milestones for students to help them improve their solution and encourage students to work incrementally and avoid marathon programming sessions. At a glance, this system operates by randomly selecting a subset of programming-related missions from a predefined list and offering a virtual reward to students for completing them. The constant presence of daily missions is intended to provide a clear guide to achievable challenges. Tracking completion of these tasks also provides support for self-monitoring. By encouraging students to complete the challenges and by recognizing the progress that results, we reinforce the student's sense of forward progress and sense of achievement as they work, combating potential discouragement from receiving negative feedback on their work.

For our design, we chose "daily" as the period for these missions because this was the most frequently observed period for such missions in most commercially successful video games. In addition, the instructor of the course involved in the study expressed the opinion that this was a preferable timescale based on the pedagogical goals of the missions, providing confirmation in the utility of this choice. Missions reset at midnight each day in the server's home timezone. The daily missions strategy was designed with the following goals in mind, based on the gamification literature.

**Missions should target tasks that help students improve the overall quality of their software.** Gamified challenges are most effective when they induce a intrinsically rewarding feeling upon completion. Students want their software to improve so they can reach a higher score, and so when students recognize the positive impact of completing a mission they will be motivated to attempt more missions.

**Missions should be directly measurable.** In order to recognize when missions have been achieved, it must be possible to measure completion automatically.

**Students should have multiple missions to choose from each day.** Another objective discussed was to prevent students from falling into "stuck" situations while attempting the missions. By offering several alternatives in cases where the student cannot get past a certain mission, students are more likely to keep working to improve their solution. This property mirrors game design features that allow players to choose from multiple paths in order to reach the next level or reward.

**Missions should each focus on a specific software task.** A key property of the Daily Missions is that they are high in numbers and low in individual size rather than consisting of a few missions with multiple criteria per mission. This is advantageous to the student because it breaks up the large assignment into small manageable tasks that are each reinforced by a rewarding experience. By this method, students are recognized for their efforts more frequently and thus more likely to repeat the behavior, according to the reinforcement theories in gamification.

**Missions should clearly indicate what the student must achieve and include the benefit of completing the task.** The sentence structure of the missions has two parts—a description of the action a student must take to complete the mission, and a note in parentheses describing the positive impact of the action. To ensure that everyone interprets the missions in the same way, the action part of the missions indicates specifically the code change that is required, and the impact part provides additional incentive for students to attempt them. This structure keeps mission descriptions short and easy to scan, with the goal of minimizing the likelihood that students will skip reading them. Student feedback reports from most automated tools already contain plenty of information competing for the user's attention, and so it is important to phrase the missions in a way that motivates students to read them. Missions can be thought of as concise instructions that reveal just enough information for users to gain a clear idea of their objective. Too many words risks an information overload that students are unlikely to read, and missions that appear excessively long might give students the impression of missions being too challenging.

Because this strategy is designed to work in conjunction with an automated grading system, it presupposes that students can submit their work in progress at any time to get feedback on how well it matches the assignment's requirements. Students rapidly adapt to such a situation, using the system to gain insights into what works, what doesn't work, and what aspects of the specified requirements are not being met. Indeed, supporting many cycles of feedback, revision, and resubmission is one of the main benefits of automated grading tools. Within this process, on the student's first submission of each day (that is, the first submission at some point after 12:00AM server time), the student is presented with five daily missions selected from a predefined list of 14 missions. These five missions are available until midnight, after which point a new set of missions is generated for the next day. The algorithm for determining which missions are offered on a given day can be summarized as follows:

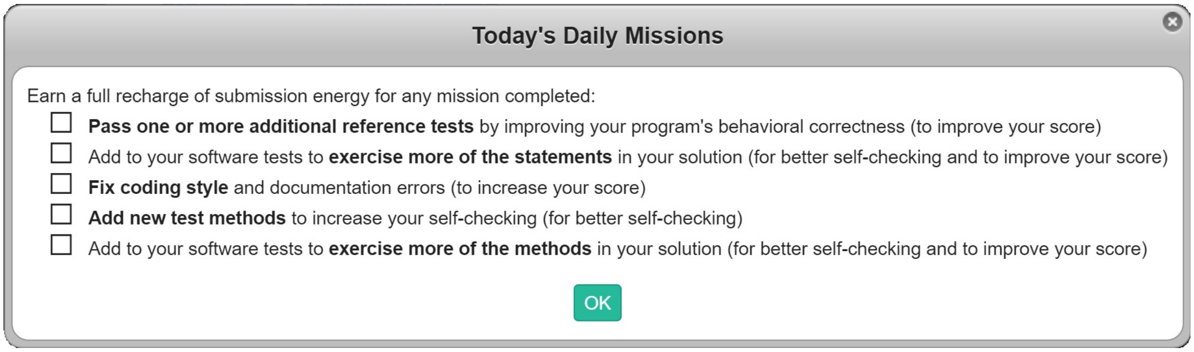
1. From the list of all 14 possible missions, a subset of eligible missions is extracted. In order for a mission to be eligible, it must be possible for the student to complete the mission based on the student's current work. For example, if a student's solution correctly implements all of the assignments expected behaviors, it is not appropriate to ask the student to improve the solution's behavioral compliance.
2. From the list of eligible missions, all missions that were offered on the previous day are removed unless doing so reduces the number of available missions below five.
3. From the remaining list, five missions are selected. If the solution is not 100% behaviorally correct, then improving the solution's correctness according to the reference tests supplied for assessing work is always selected as one of the missions, and the remaining four are randomly selected from the pool of available missions. For solutions that are already 100% behaviorally correct, five missions are randomly selected from the poo instead.

The list of all 14 missions from which each day's missions is selected is given below. Each mission corresponds to a different progress indicator:

1. Add more methods to your solution, to make your methods shorter and easier to understand (to increase readability and improve method design)
2. Fix coding style and documentation errors (to increase your score)
3. Simplify the logic in your solution by removing or consolidating if statements or loops (to increase readability and testability, and reduce potential for bugs)
4. Reduce the length of your methods by breaking them into more manageable pieces (to increase readability and improve method design)
5. Add comments to your solution (to increase readability)
6. Add another class to your solution (for better solution design)
7. Pass one or more additional reference tests by improving your program's behavioral correctness (to improve your score)
8. Add new test methods to increase your self-checking (for better self-checking)
9. Add to your software tests to increase your self-checking (for better self-checking)
10. Add to your software tests to exercise more of the statements in your solution (for better self-checking and to improve your score)
11. Add to your software tests to exercise more of the methods in your solution (for better self-checking and to improve your score)
12. Add to your software tests to exercise more of the logic conditions in your solution (for better self-checking and to improve your score)
13. Add more assertions to your software tests so they check expected behaviors more comprehensively (for better self-checking)
14. Add another test class (for better self-checking)

### 3.1. Rewards

Daily missions aim to encourage students to view and attempt missions until they are completed, where the incentives for doing so include indications of code improvement and virtual rewards. Consideration of rewards for gamification is an important choice. While we did consider assignment gains such as extra credit, this option was not pursued in this design. Some find that extra credit tends to primarily benefit high-achieving students and may provide a misleading basis for rewarding learning [21]. Further, gameplay



Today's Daily Missions

Earn a full recharge of submission energy for any mission completed:

- □ **Pass one or more additional reference tests** by improving your program's behavioral correctness (to improve your score)
- □ Add to your software tests to **exercise more of the statements** in your solution (for better self-checking and to improve your score)
- □ **Fix coding style** and documentation errors (to increase your score)
- □ **Add new test methods** to increase your self-checking (for better self-checking)
- □ Add to your software tests to **exercise more of the methods** in your solution (for better self-checking and to improve your score)

**Figure 1. Daily missions are shown in a modal dialog, displayed via a button or automatically when new missions are generated.**

rewards usually do not translate to real-world gains and instead provide intrinsic benefits.

Instead, other types of points-independent rewards can be used. Irwin and Edwards [22] have experimented with *submission energy* inspired by commercially successful game design techniques that condition frequent gameplay. Players have a limited but regenerative resource required for play that they must manage. Doing so encourages small, periodic play sessions spread over a longer period of time [22]. To achieve this goal in an automated grading system, Irwin gave each student an *energy bar* displayed on-screen that holds a maximum of three units of submission energy. One unit of energy is used each time a student makes a submission, and a student can only submit if they have energy. A partially filled energy bar regenerates at the rate of 1 unit per hour until full. Results indicate that this approach does encourage students to start working earlier, to work more frequently, and to think more carefully about the changes they make each time they submit to get additional feedback.

The daily missions feature adds another layer of incentive to this submission energy concept. In addition time-based regeneration, submission energy is awarded to students for completing daily missions. For each newly completed mission, the student has their submission energy bar completely refilled, earning 1-3 units of energy. Other ideas for non-points-based rewards are discussed in [23].

## 3.2. Visual Display and Presentation

The section details the visual elements used to represent the Daily Missions and why these choices were made over others. Like the design of the missions themselves, the visual design was driven by the following goals.

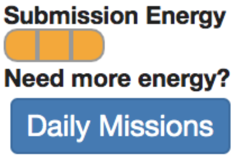


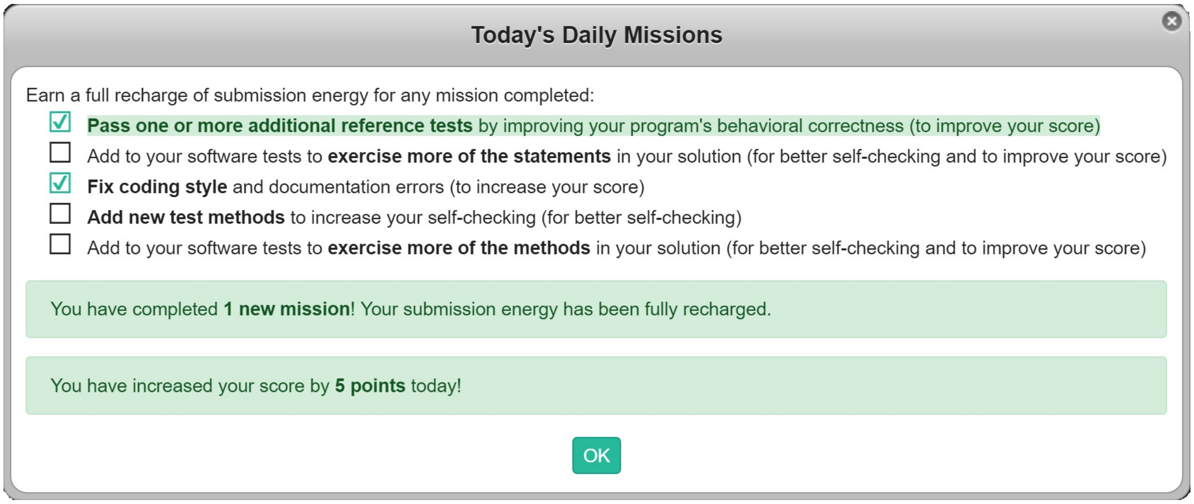
**Figure 2. The "Daily Missions" button is placed adjacent to the submission energy bar.**

**The display should alert the user when new missions have become available.** Since new missions are generated each day, the student needs to be aware when the set of active missions changes over to a new set. Calling attention to the new set of missions will help keep the availability of the missions in the student's mind, while also making them aware of the change in missions that are available.

**The display should indicate which missions have been completed and which missions have yet to be achieved.** Completed missions should stay visible to students in order to recognize their accomplishments made so far, but the display should also differentiate between missions completed and missions not completed so that students do not get confused.

**The display should recognize when a student completes a new mission in a way that is distinguishable from older missions completed.** When a new mission is completed, it is important that students be made aware of the measured progress they just made and attain a feeling of accomplishment. Therefore, newly completed missions should be given visual attributes that are different from those completed in previous submissions.

Since the feedback page produced by the automated grading system already contains plenty of static information on the screen, we chose not to display


Today's Daily Missions

Earn a full recharge of submission energy for any mission completed:

☑ **Pass one or more additional reference tests** by improving your program's behavioral correctness (to improve your score)

□ Add to your software tests to **exercise more of the statements** in your solution (for better self-checking and to improve your score)

☑ **Fix coding style** and documentation errors (to increase your score)

□ **Add new test methods** to increase your self-checking (for better self-checking)

□ Add to your software tests to **exercise more of the methods** in your solution (for better self-checking and to improve your score)

You have completed **1 new mission**! Your submission energy has been fully recharged.

You have increased your score by **5 points** today!

**Figure 3. Displaying an additional mission just completed, the state of a previous mission, the reward earned, and the impact on the current score.**

the mission information inline as part of the feedback page itself. Based on prior experience with automated feedback design, we believed doing so would make it more likely that students would view the daily missions as visual noise and would skip over the content in favor of scanning for score information. Instead, we designed the daily mission display as a pop-up modal dialog that would only be visible sometimes, but would provide a mechanism for calling the student's attention to mission information when necessary [24]. While pop-up dialogs have potential usability issues, we elected to have the dialog normally hidden in all situations where no new information was available. Students can call up the dialog to view their mission progress at any time through a dedicated button on the feedback page.

Figure 1 shows the content of the daily mission dialog for a sample submission. Here, the set of 5 missions available on the current day is shown, and none of the missions have been completed so far. When new missions are generated on the first submission of each day, the daily mission dialog is *automatically popped open* when the student loads the feedback page so that the new missions are immediately visible.

Throughout the day, any time the student wishes to check the status of their missions or see which missions are available, they can open the dialog by clicking on a button labeled "Daily Missions", as shown in Figure 2. To make the button prominent when needed, the button was placed directly underneath the submission energy bar showing the student's currently available submission energy. In addition, the key phrase "Need more energy?" serves as an informative label inviting the student to open the dialog by hinting at the rewards earned for completing missions. These choices balance the need to remind students of the availability of the missions in a task-oriented way, while also avoiding the problem of students becoming "blind" to less-frequently used content on information-heavy web pages.

While the student can open the daily missions dialog at any time, the dialog also automatically pops up on when the student has just completed one or more of the missions. This actively alerts the student to the change in mission status, as well as highlights the reward earned. Figure 3 shows the same set of missions after the student has been through several submissions and where the current submission has completed a new mission. As shown, one new mission has been completed and is highlighted in the list. A second mission that was completed earlier has also been checked off, so the student can clearly see the status of all of today's missions at once. A message at the bottom reminds the student that their submission energy has been fully recharged because of the new achievement. Finally, if the student's score has increased today, the dialog also indicates this relative gain to remind the student of the forward progress that has been achieved. While there is no guarantee that these points were earned because of any of the missions, the message provides additional reinforcement while also recognizing the small-scale achievements the student has made.

## 4. Experiences in the Classroom

After designing the daily missions strategy, we implemented it and conducted a pilot deployment in the Spring 2019 offering of CS1 at Virginia Tech. 322 students enrolled in the course all experienced the daily missions, 218 of whom consented to allow their data to be used in this study. The daily missions were introduced on later assignments in the course, after students had already gained experience using the automated grading system. Initial use of the feature helped uncover various implementation issues, and also allowed us to refine and improve the visual presentation and reward mechanisms. Data was then collected to assess mission effects on the final assignment.

## 4.1. Mission Performance

The 218 students who consented to participate made a total of 1,706 submissions to the assignment included in the study, for a mean of 7.8 submissions per student (s.d. 4.9). Also of interest is the number of distinct sets of missions students saw during one assignment. Since one new set of missions is offered each day, a measure of how many mission sets a student saw is also a measure of how many days early a student made their first submission. Among all students who submitted the assignment, 67.4% only saw one set of missions, 23.3% saw two sets of missions, 5.1% saw three sets of missions, and 4% saw four sets of missions. The average number of mission sets a student saw was 1.5.

Figure 4 illustrates the number of students who saw each mission at least once and the percentage of students who completed each mission among the pool who saw them. On the y-axis of the chart, the names of each mission are displayed. On the x-axis, the numerical value colored in blue represents the number of students who saw the corresponding mission, and the length of the green bar maps to the percentage of those students who completed the mission.

Among the missions assigned to students, missions were completed 72.2% of the time (s.d. 38.7%). However, progress on all missions was tracked, including those that were not assigned to students. For missions that were not assigned to students, they were only completed 61.5% of the time (s.d. 31.7%). This difference was statistically significant according to an analysis of variance (F(1, 756) = 17.2, p < 0.0001). This suggests students were more likely to complete missions that were assigned than those that were not, and it is plausible that this was caused by the presence of the daily missions feature.

## 4.2. Survey Results

We used an only survey to collect student perceptions of the daily missions strategy. For each of the statements listed on the survey, students were able to respond with one of seven choices from "strongly agree" to "strongly disagree". To avoid as much as possible any built-in bias incurred by the phrasing of the statements, some statements were centered on positive outcomes pertaining to the daily missions and others were centered on negative outcomes.

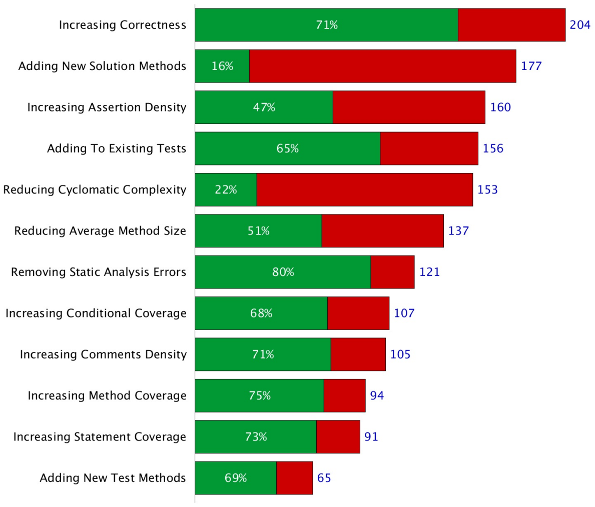


**Figure 4. Number of students seeing and completing each mission.**

Table 1 summarizes the survey results. Responses were mapped to a 1–7 scale where 1 represents a "strongly disagree" response, 7 represents "strongly agree" and 4 represents neutral. Students reflected mildly positive responses to all questions across the board. Among the strongest responses, 54% of students agreed that "reinforced the things I did well", with only 17% disagreeing. Similarly, 50% of students agreed that the daily missions gave them "a better idea of ways to improve" their code, with only 19% disagreeing. Student responses indicate that missions were around the right level of difficulty, with only 11% of students believing believing they were too difficult while the majority (54%) did not. At the same time, only 41% of students believed the missions were too easy. 57% of students agreed that one day was enough time for completing a mission, with only 11% disagreeing.

We performed a stepwise linear regression to explore potential relationships between survey answers on all survey questions, and overall course performance (final course grades, as determined by cumulative numeric course scores at the end of the course). In this analysis, only one survey question was significantly associated with course outcomes: agreeing with the statement that daily missions "gave me a better idea of ways to improve my code" was *negatively* related to course outcome (F = 5.54, p = 0.0232). It is plausible that stronger students, or students with more experience, both found that the subgoals used as daily missions were not new

**Table 1. Survey response summary (4 is neutral).**

| Question | Score |
|---|---|
| *Positively phrased questions* | |
| Motivated me to start on the assignment early | 4.3 |
| Motivated me to work on the assignment more frequently | 4.5 |
| Conditioned me to be a better programmer | 4.4 |
| Reinforced the things I did well on the assignment | 4.7 |
| Gave me a better idea of ways to improve my code | 4.7 |
| Made me more confident in my ability to complete the assignment | 4.2 |
| Made the assignment seem more manageable | 4.3 |
| Accomplishing them was a satisfying feeling | 4.5 |
| *Negatively phrased questions* | |
| Made it seem like I had extra work to do for the assignment | 3.3 |
| Were difficult to understand | 3.2 |
| Did not help improve my assignment score | 4.3 |
| Were too difficult or frustrating to achieve | 3.1 |
| Were too easy to achieve | 4.3 |
| One day was not an adequate amount of time | 3.0 |

ideas to them, and also performed better overall (or that students with less experience found missions more useful, even though they also performed less well). In either case, however, this relationship seems likely due to a separate factor related to individual student ability or experience that was not controlled in this study, so specific conclusions related to course performance are not indicated based on this evidence.

Among the comments students entered on the survey, the most common theme was that of mission rewards. Multiple comments indicated that students valued the ability to gain additional submission energy through completing missions. However, there were two concerns about the reward mechanism. First, some students noted that if they completed multiple missions in one submission, they earned a full recharge of their submission energy—the same reward as if they only completed a single mission. Students valued the reward, but felt cheated or treated unfairly that they did not earn the full value in this situation. Second, other students pointed out that a full recharge of their submission energy does not mean as much if their energy bar is already full, and that they would rather use that reward when energy is out. While these comments convey student concerns about the rewards, they also indicate that students did find value in them and wished to be able to fully utilize that value, while also feeling unfairly treated if they did not receive the full value they believed they earned. This suggests both the missions and the reward scheme were recognized by the students and were significant enough to affect student decisions, although they also indicate additional work is needed.

## 5. Conclusion

This paper describes the design of a daily missions strategy that can be added to automated grading systems. Daily missions give students a frequent sense of accomplishment and forward progress by providing small, achievable tasks. They allow students to self-monitor, invite them to rise to challenges, and help develop persistence. This pilot study shows the viability of the approach and indicates a mildly positive reception by students. Further, students were significantly more likely to complete missions they were assigned. Daily missions offer a new technique for gamification that fits nicely into project-based assignments. At the same time, this field-test only establishes feasibility. A longer-term study over a full academic term is necessary to determine whether daily missions can encourage students to start assignments earlier, as well as to assess changes in programming habits or student achievement. Further work is necessary on the reward scheme to provide the smoothest experience.

Within the limited scope of this pilot study, survey responses indicate that students agreed with statements about daily missions' motivational impact and identification of ways for students to improve their code. Responses also suggest on average that 24 hours was an appropriate amount of time to complete the missions and that the missions were not overly difficult or overly easy. The daily missions strategy was based primarily on literature research and common commercial practices, making it difficult to form reliable predictions about the effect of various attributes. However, from this experiment, a number of future improvements are possible to refine the approach. With regard to rewards, one of the issues raised was that the reward for completing one mission was the same as for completing multiple at once. To more appropriately reflect the level of accomplishment achieved, a future improvement would be to change rewards so that completing two missions earns twice the amount of submission energy as completing one. This may entail allowing the energy bar to be "overcharged", holding more than three units when necessary.

An idea considered during this design is the concept of *starter missions* consisting of basic tasks providing direction on how to begin work on the assignment. These would be assigned in place of random missions when a student first begins an assignment. These missions would ideally give students a greater sense of early accomplishment and better guide students through the beginning phases of an assignment. A future version of daily missions in which this type of initialization missions are offered would be helpful to explore.

Finally, the current design offers opportunities for students to "game" the system. For example, if a student figures out that the "Adding New Solution Methods" mission can be completed simply by adding one (new) empty method, the student can intentionally keep this mission incomplete until they need an energy recharge. They can then complete the mission, receive the reward, and then undo the change. Strategies for assessing and preventing such negative behaviors need exploration.

## Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. DUE-1625425. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

## References

[1] K. M. Kapp, *The gamification of learning and instruction: Game-based methods and strategies for training and education*. San Francisco, CA, USA: Pfeiffer, 2012.

[2] M. Leaning, "A study of the use of games and gamification to enhance student engagement, experience and achievement on a theory-based course of an undergraduate media degree," *Journal of Media Practice*, vol. 16, no. 1, pp. 1–16, 2015.

[3] A. Santos, "The role of emotion in ux (what "happy users" really means)," *Envanto*, 2017.

[4] A. Lieberoth, "Shallow gamification: Testing psychological effects of framing an activity as a game," *Games and Culture*, vol. 10, no. 3, pp. 229–248, 2015.

[5] C. Poondej and T. Lerdpornkulrat, "The development of gamified learning activities to increase student engagement in learning," *Australian Educational Computing*, vol. 31, no. 2, 2016.

[6] E. Kaila, M.-J. Laakso, T. Rajala, and E. Kurvinen, "A model for gamifying programming education: University-level programming course quantified," in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pp. 689–694, 2018.

[7] B. Brown, "The psychology of gamification in 2016: Why it works (& how to do it!)," *Bitcatcha*, 2019.

[8] M. Glotzbach, "Gamification can help education—here's how," *VentureBeat*, 2018.

[9] J. Gressick and J. B. Langston, "The guilded classroom: Using gamification to engage and motivate undergraduates," *Journal of the Scholarship of Teaching and Learning*, vol. 17, no. 3, pp. 109–123, 2017.

[10] B. J. Zimmerman, "Self-regulated learning and academic achievement: An overview," *Educational Psychologist*, vol. 25, no. 1, pp. 3–17, 1990.

[11] A. Damsa and R. Fromann, "Gamification and gameful approaches in education, business, and IT," *Informatika*, vol. 43, no. 1, pp. 28–33, 2017.

[12] E. L. Deci, R. Koestner, and R. M. Ryan, "A meta-analytic review of experiments examining the effects of extrinsic rewards on intrinsic motivation," *Psychological Bulletin*, vol. 125, no. 6, pp. 627 – 668, 1999.

[13] W. Scott, "The effects of extrinsic rewards on "intrinsic motivation": A critique," *Organizational Behavior and Human Performance*, vol. 15, no. 1, pp. 117–129, 1976.

[14] D. L, "Game reward systems," *Learning Theories*, 2016.

[15] H. Wang and C.-T. Sun, "Game reward systems: Gaming experiences and social meanings," in *Proceedings of DiGRA 2011 Conference: Think Design Play*, 2011.

[16] H. O'Neil and M. Drillings, eds., *Motivation: Theory and Research*. New York: Routledge, 1994.

[17] D. Thompson, T. Baranowski, R. Buday, J. Baranowski, M. Juliano, M. Frazior, J. Wilsdon, and R. Jago, "In pursuit of change: Youth response to intensive goal setting embedded in a serious video game," *Journal of Diabetes Science and Technology*, vol. 1, no. 6, pp. 907–917, 2007.

[18] A. Bandura, *Social Learning Theory*. Englewood Cliffs, N.J.: Prentice Hall, 1977.

[19] P. Ihantola, T. Ahoniemi, V. Karavirta, and O. Seppälä, "Review of recent systems for automatic assessment of programming assignments," in *Proceedings of the 10th Koli Calling Int'l Conf. Computing Education Research*, (New York, NY, USA), pp. 86–93, ACM, 2010.

[20] S. Edwards and Z. Li, "Towards progress indicators for measuring student programming effort during solution development," in *Proceedings of the 16th Koli Calling Int'l Conf. Computing Education Research*, (New York, NY, USA), pp. 31–40, ACM, 2016.

[21] J. C. Norcross, L. J. Horrocks, and J. F. Stevenson, "Of barfights and gadflies: Attitudes and practices concerning extra credit in college courses," *Teaching of Psychology*, vol. 16, no. 4, pp. 199–203, 1989.

[22] M. S. Irwin and S. H. Edwards, "Can mobile gaming psychology be used to improve time management on programming assignments?," in *CompEd '19: Proceedings of the ACM Conference on Global Computing Education*, (Chengdu, China), pp. 208–214, ACM, 2019.

[23] S. H. Edwards and Z. Li, "Designing boosters and recognition to promote a growth mindset in programming activities," in *Proceedings of the 2019 ASEE Annual Conference & Exposition*, (Tampa, Florida, USA), ASEE, 2019.

[24] T. Fessenden, "Modal & nonmodal dialogs: When (& when not) to use them," *Nielsen Norman Group*, 2017.