

## **AC 2007-253: ENCOURAGING CREATIVITY IN INTRODUCTORY COMPUTER SCIENCE PROGRAMMING ASSIGNMENTS**

### **Tammy VanDeGrift, University of Portland**

Tammy VanDeGrift is an Assistant Professor at the University of Portland. She received a B.A. from Gustavus Adolphus College and her M.S. and Ph.D. degrees from the University of Washington (Seattle). Her research interests include computer science education, educational technology, multimedia, software engineering, and CS theory.

# Encouraging Creativity in Introductory Computer Science Programming Assignments

## Abstract

With computer science enrollments declining and the perception that programming is boring, computer science and computer engineering educators are challenged to raise awareness of the discipline. In order to keep students' interest and to provide a means of ownership, creative and open-ended programming assignments are used in an introductory Java course. At the end of the semester, students completed a survey about the programming assignments in the course. Survey results indicate that 64% of students shared at least one program they created with a friend or family member, indicating that students took ownership of their computer programs. Comments that the assignments were "fun", "creative", "could be run later" suggest that students took ownership of their programs. Also, 45% of students added optional features to at least one program over the course of the semester. This paper describes the homework assignments used in the course, examples of students' work, and students' perceptions of the assignments.

## 1. Introduction

With computer science enrollments declining and the perception that programming is boring, computer science and computer engineering educators are challenged to interest more students in the discipline<sup>12</sup>. Having students with and without programming experience in introductory courses and the "ease" of copying code files for submission also pose challenges for computer science educators. In many introductory computer science courses, students complete programming assignments to learn the skills of problem-solving, translating ideas into computer code, debugging programs, and testing programs. Much of the learning takes place while students complete programming assignments. In order to keep students' interest and provide a means of ownership, creative and open-ended programming assignments were used in an introductory Java course.

Most introductory programming courses include a series of programming assignments to ensure students learn programming fundamentals. A typical introductory programming assignment requires all students to complete the same program. Instead of stating project specifications so that all students' projects tackle exactly the same task, assignments for an introductory course were written to allow students to be creative (yet still learn the programming fundamentals). This paper describes a series of such assignments used in an Introduction to Java course. For example, students designed their own madlib story for the string-processing assignment and designed their own custom recipe calculator for basic input/output and mathematical functions. Later during the semester, students completed an adventure game and designed their own characters and their behaviors.

Even though assigning "creative" assignments does not lend itself to automated grading and testing, the author believes that to truly assess students' code, the code itself must be read and graded for style and quality. Therefore, executing each student's program and reading their code takes time, but its advantages in encouraging students to be creative outweigh the cost of

grading. By giving students some latitude in designing their projects, they have the opportunity to explore and learn concepts.

At the end of the semester, students anonymously answered questions about the homework assignments, if they shared their work with friends and/or family, and if they added extra features to their projects. This paper describes a study that answers the following research questions: 1. Did students complete extensions beyond the requirements?, 2. Did students take ownership and pride in their projects?, 3. What assignments did students like best/least and why?, and 4. What is the range of students' projects for a single assignment?

Survey results indicate that 64% of students shared at least one program they created with a friend or family member, indicating that students took ownership of their computer programs. Also, 45% of students added optional features to at least one program over the course of the semester. The favorite and least favorite assignment was the same assignment – a text-based adventure game.

Following the introduction, descriptions of the assignments and grading rubrics are provided in Section 2. The research study including the study participants, research methods, and results are detailed in Sections 3 and 4. Finally, related work and conclusions are described in Sections 5 and 6.

## **2. Background**

### **2.1 Learning Theories**

Several educational theories and instructional practices informed the development of the open-ended programming assignments. Providing models for students is an important part of teaching and learning according to Norman<sup>7</sup>. Students build knowledge by looking at prototype models; therefore, the assignments described in this paper use contexts familiar to students. A second learning theory that served as a basis for assignment construction was the constructivist learning theory<sup>4,10</sup>. Students build knowledge based on existing knowledge. Students must actively construct new ideas and experiment with those new ideas and mold them into existing knowledge. Therefore, students have the opportunity to make several decisions and take ownership of their learning path while constructing solutions to the programming assignments. This relates to problem-based learning<sup>5</sup>, where students are given problems to solve and they seek the necessary material, digest it, and apply it to solve the problem. A third learning theory that informed the creation of the programming assignments is that of metacognition<sup>1,2</sup>. Learners must be aware of their own cognitive system and monitor their own learning process and style. The assignments were devised to have a tandem written summary, where students have an opportunity to reflect upon the learning process while (hopefully) providing a hook for metacognitive activity.

### **2.2 Programming Assignments**

Nine programming projects were assigned in a CS1 semester-long course. The research study results reported in this paper are based on two semesters of data. The assignments were first

piloted in Autumn 2005. Survey responses were gathered from students completing the course in Spring 2006 and Autumn 2006. The CS1 course took place at a small university with a liberal arts core curriculum. The CS1 course typically attracts students majoring in computer science, electrical engineering, math, and the physical sciences. Table 1 describes each assignment, the target concepts, and the assignment characteristics regarding creativity. Some assignments had more room for creativity than others. Note that the optional suggestions did not result in extra credit toward students' grades. The assignments from Autumn 2006 can be found at the following URL: <http://teaching.up.edu/CS203/Archive/AU06/>

**Table 1: Programming Assignment Descriptions and Creative Aspects**

<b>Assignment Name</b>	<b>Assignment Description</b>	<b>Target Concepts</b>	<b>Creative Elements</b>	<b>Optional Suggestions</b>
Recipe	Create a custom recipe of your choice for a user's desired number of servings	Strings Input Output Arithmetic Variables	Each student produces a program for a different recipe	Convert units Add error-checking on user input. Create a "database" of recipes.
Madlib	Create a madlib story and prompt user for words (nouns, verbs, etc.) to fill in blanks of the story	String Processing Input Output	Each student produces a different madlib story	Add error-checking on user input. Use graphics for displaying prompts. Create a repository of stories. Create a choose-your-own-adventure story.
Landscape	Create an outdoor scene with at least 4 distinct objects. The user can select to view the landscape during the day or at night.	Input Output Graphics Conditionals	Each student produces a different outdoor scene	Animate the scene by displaying different frames in sequence.
Tiling	Create floor tile designs and present them in various patterns – checkerboard, stripes, and pattern of students' choice	Input Output Graphics Conditionals Loops	Each student produces different floor tile designs and their own floor pattern	Create 3D-looking tiles. Create tiles and room layouts from user-defined dimensions.

Connect Four	Create a connect four game that supports two players	Input Output Conditionals Loops Arrays Methods		Allow varying game board sizes. Add computer player option and make “intelligent” playing decisions.
Bookstore *	Create classes for an on-line book retailer – a book class and a customer class	Classes Instance Variables Methods Objects	Each student produces a different test plan – creating books and customers of their choice	Add error-detection to book account numbers. Create a BookInventory class and/or AccountInventory class.
Credit Card **	Create a class to represent a credit card for a credit card company	Classes Instance Variables Methods Objects	Each student produces a different test plan – creating credit cards of their choice	Add error-detection to credit card numbers. Add an AccountInventory class.
Image Filters	Create the following image filters: demosaic filter used in digital cameras, flip pictures horizontally/vertically, mirror the image, color to grayscale, and a filter of student’s choice	Loops Interfaces Arrays of Objects Classes Methods	Each student can use their own digital pictures. Each student creates a filter of their choice	Create a watercolor filter to transform a photo into a watercolor painting. Create an edge detector filter to find outlines of shapes in images.
Adventure Game	Create and modify characters (people) in a text-based adventure game. Each character can act automatically and the computer user can interact with the text-based world.	Inheritance Classes Vectors Iterators	Each student creates their own characters, actions for the characters, and locations in the game.	Add a goal to the game. Add more characters, commands, and actions to the game. Add a theme to the game.
Maze Solver	Create a program to determine if a maze has a solution or not	Recursion File I/O Arrays Loops	Each student creates their own maze files for testing	Process non-rectangular mazes. Display the path through the maze if such a path exists. Process 3D mazes.

\* used in Autumn 2006

\*\* used in Spring 2006

Students completed assignments individually throughout the semester. For each assignment, students completed the code and a companion written report. The written reports provided a forum for students to describe the features (sometimes bugs) of their programs, a user manual, the code internals, their testing process, their test cases, and what they learned from completing the assignment.

## **2.3 Grading Rubric**

Each assignment was graded out of 12 points. The following categories were each worth four points: Operation, Quality, and Written Summary. Operation refers to the program execution adhering to the assignment specification. Quality refers to quality of the written code which includes proper commenting, indentation, whitespace, naming, and functional decomposition. Finally, the written summary was graded according to clarity, conciseness, proper use of technical vocabulary, and completeness.

## **3. Methods and Data Collection**

### **3.1 Student Population**

During Spring 2006, 14 students were enrolled in the CS1 course. Of the 14 students, 12 completed the survey at the end of the term. During Autumn 2006, 22 students were enrolled and 19 students completed the end-of-term survey. In total, there were 31 responses to the survey spanning two different semesters.

### **3.2 Data**

Data analyzed for this study came from two sources: students' homework submissions and the survey responses. Both forms of data collection were approved by the author's Institutional Review Board (IRB) for research studies involving human subjects.

The following questions were asked on the end-of-term survey submitted anonymously by students:

- What did you like best about the homework assignments?
- What would you suggest to improve the homework assignments?
- Estimate the number of hours that you spent per week on homework assignments.
- Did you share your programs with your friends?
- Did you share your programs with your family?
- Which assignment did you like best?
- Which assignment did you like least?
- How did writing the summaries benefit or not benefit your learning?
- Did you collaborate with other students?
- For each assignment, circle yes if you completed any additional suggestions and no if you did not.

## 4. Results

The purpose of this study was to determine how assignments with creative aspects and optional extensions affected students' behavior and feelings of ownership. The data collected was analyzed to answer the following research questions:

1. Did students complete extensions beyond the requirements?
2. Did students take ownership and pride in their projects?
3. What assignments did students like best/least and why?
4. What is the range of students' projects for a single assignment?

### 4.1 Completing Extensions

Each assignment had ideas for optional extensions, as presented in Table 1. Completing these extensions was for the students' benefit and they received no extra credit toward their homework grades. The first research question asks if students completed optional extensions to see if providing homework frameworks with this extensibility is worth the instructor's time.

Of the 31 students surveyed at the end of the term, 14 students (45%) stated they completed at least one optional extension during the course of the semester. The homework submissions confirm this number as low (since more students completed the actual assignments than those surveyed at the end of the semester). Table 2 shows the number of students who completed extensions for each homework assignment. For each assignment, at least 2 people completed optional extensions. The assignment with the most optional extensions completed was the landscape graphics assignment.

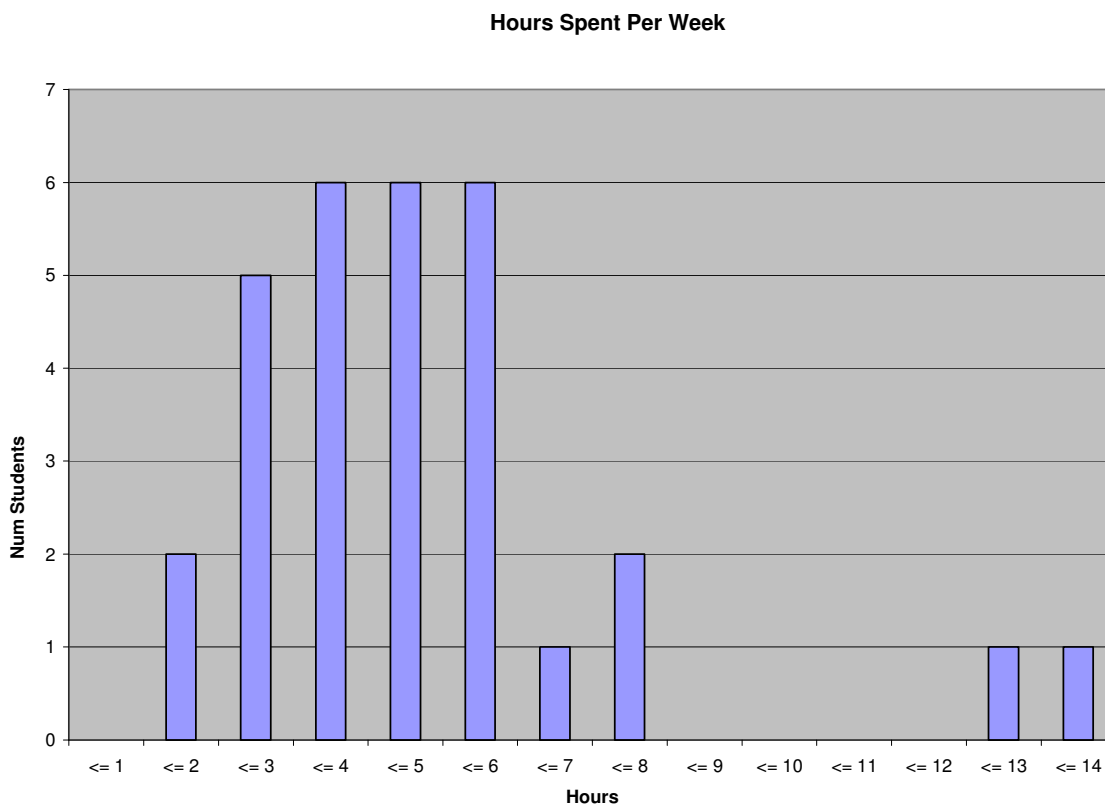
**Table 2: Number of students who completed optional extensions for each assignment**

Homework Assignment	Number of students	Percentage of students
Recipe	8	25.8
Madlib	8	25.8
Landscape	10	32.2
Tiling	6	19.3
Connect Four	4	12.9
Credit Card / Bookstore	4	12.9
Image Filters	3	9.7
Adventure Game	5	16.1
Maze Solver	2	6.5

When looking at the data per student, the maximum number of extensions completed by an individual student was 8. The average number of extensions completed by the 14 who completed at least one extension was 3.57.

## 4.2 Taking Ownership

Taking ownership or pride of the assignments is evidenced by students sharing the assignments with friends and family and the number of hours spent on homework assignments. Of the 31 respondents, 15 (48%) shared at least one assignment with a friend and 12 (39%) shared at least one assignment with a family member. Twenty (64%) of the 31 respondents shared at least one assignment with a friend or a family member. This statistic indicates that these students took enough pride in their work to share it with someone else. Figure 1 shows the average number of hours spent per week by students on programming assignments in the course (one student left the answer blank). All students spent at least 1 hour on the assignments per week. Also, the figure indicates a reasonable workload per week, as students are expected to work 6 – 9 hours outside of lecture per week for a 3-credit-hour course.



**Figure 1: Number of Hours Spent Per Week By Students On Assignments**

Another measure of taking ownership was found in the responses to the question “What did you like best about the programming assignments?” on the survey. Of the 31 responses, 17 (55%) mentioned what they liked best about the assignments had to do with six of these emergent categories: creativity, interesting/fun, rewarding/accomplishment, can use them later, can show friends, and real-world problems. The following are quotes from each of the five categories:

- “being able to be creative, getting to play the game created”
- “The open-endedness .. I loved it! There was always something to make the program that much more fun to code.”

- “The challenge and the feeling of accomplishment.”
- “And we got to have them at the end of the day and use them.”
- “They were fun things to play and show friends.”
- “Very applicable to real world programming problems.”

The quotations by students show that they took ownership and pride in what they created. Five students explicitly mentioned they liked the room for creativity, which was the main design goal of the programming assignments.

### 4.3 Favorite and Least Favorite Assignments

Students reported their favorite and least favorite assignments on the end-of-term survey. Table 3 shows their favorite assignments. Table 4 shows their least favorite assignments. What is interesting is that the most and least favorite assignment is the same one – the text-based adventure game. The Credit Card and Bookstore assignments had the fewest creative extensions, as the assignment was designed to introduce the mechanics of writing classes and methods in Java. Several other assignments appear in both tables, indicating that the same assignment appealed to some students and did not appeal to others.

**Table 3: Favorite Assignment Reported By Students**

<b>Favorite Assignment</b>	<b>Number of Students</b>
Adventure Game	7
Connect Four	6
Maze Solver	4
Image Filters	3
Landscape	3
Madlib	3
Tiling	1
Recipe	1
NO RESPONSE	3

**Table 4: Least Favorite Assignment Reported By Students**

<b>Least Favorite Assignment</b>	<b>Number of Students</b>
Adventure Game	7
Credit Card / Bookstore	6
Image Filters	5
Connect Four	3
Landscape	2
Recipe	1
(Written assignment other than programming)	1
NO RESPONSE	6

Students cited several reasons for why they liked their favorite assignment and disliked their least favorite assignment. The reasons listed were coded into the categories listed in Table 5 and

Table 6. Table 5 shows the reasons and the number of students who cited the reason for liking their favorite assignment. Statements regarding pride and ownership, such as fun, creativity, and rewarding again show up as reasons for the favorite assignment. Table 6 shows the reasons for disliking their least favorite assignment.

**Table 5: Reasons for favorite assignment**

<b>Reason for Favorite</b>	<b>Number of Students</b>
Fun	9
Can be creative	3
Rewarding	2
Can play game	2
Real world context	2
Challenging	1
Useful	1
Siblings play my game	1
Interesting	1
Visual Results	1

**Table 6: Reasons for Least Favorite Assignment**

<b>Reason for Least Favorite</b>	<b>Number of Students</b>
Too much time to complete	4
Too much coding	3
Not interesting	2
The most difficult	2
Confusing	2
No coding	2
No room for creativity	1
Too easy	1
Did not complete it	1
No graphics	1
Tedious	1

#### **4.4 Range of Student Submissions**

Because the assignments included creative aspects, students' submissions had a wide range of contexts for variation. Below are example products of students' programs. Figures 2 and 3 show two different recipe creations and Figures 4 and 5 depict two landscapes. As the reader can see, students chose their own recipe to customize and chose their own landscape to depict.

Ingredients for 4.0 servings of Scrunchy Sweet and Sour Chicken:

4.0 egg yolks

Salt and Ground Black Pepper

4.0 skinless, boneless chicken breast halves, cubed

Vegetable oil

The following is for the Sweet and sour sauce to top the chicken; it is tailored to the 4.0 servings you requested

1.0 Onion(s), sliced

1.0 Small Red Pepper(s), cut into one inch pieces

1.0 Small Orange Pepper(s), cut into one inch pieces

1.0lb of Pacific Friend Pineapple cubes in natural juice

1.0tbsp Cornstarch

2.0tbsp Tomato Ketchup

2.02 tbsp Light Soy Sauce

1.0 tbsp White Wine Vinegar

Handful of fresh Cilantro leaves, to garnish

Preparation - 20 minutes, cooking time - 15 minutes

Directions:

Mix together the egg yolks, with a tbsp of water, the cornstarch, and some salt and pepper.

Heat 2 inches of oil in a wok or deep frying pan. Toss the chicken in the cornstarch mixture and deep-fry in batches for 5 minutes or until crisp and golden.

Drain on paper towels.

Empty the oil from the wok to leave a thin coating in the pan. Stir-fry the onion and peppers over a high heat for 3 minutes.

Drain the pineapple cubes (reserving the juice), add to the pan, and cook for a minute or two.

Mix together the cornstarch and a little of the pineapple juice to form a paste, then stir in the remaining juice, the ketchup, soy sauce, vinegar, and 1/2 cup water.

Pour this into the pan and bring to a boil, stirring until the mixture thickens.

Stir the chicken pieces into the pan and simmer for 5 minutes until cooked through. Check the seasoning then divide between bowls.

Scatter the cilantro on top and serve. Enjoy!

Thank you Joe for choosing the custom recipe center.

**Figure 2: Recipe Example 1**

Welcome to the customized recipe center.

Please enter your name: Susie

What type of marsala wine do you prefer? abcdefg

How many servings of chicken marsala would you like? 5

Ingredients:

10.0 boneless chicken breast halves without skin

0.625 cup flour

1.25 teaspoon salt

0.625 teaspoon pepper

1.25 teaspoon basil, dried

7.5 tablespoons butter or margarine

7.5 tablespoons olive oil

15.0 ounces of fresh mushrooms, sliced

1.25 cup abcdefg Marsala Wine

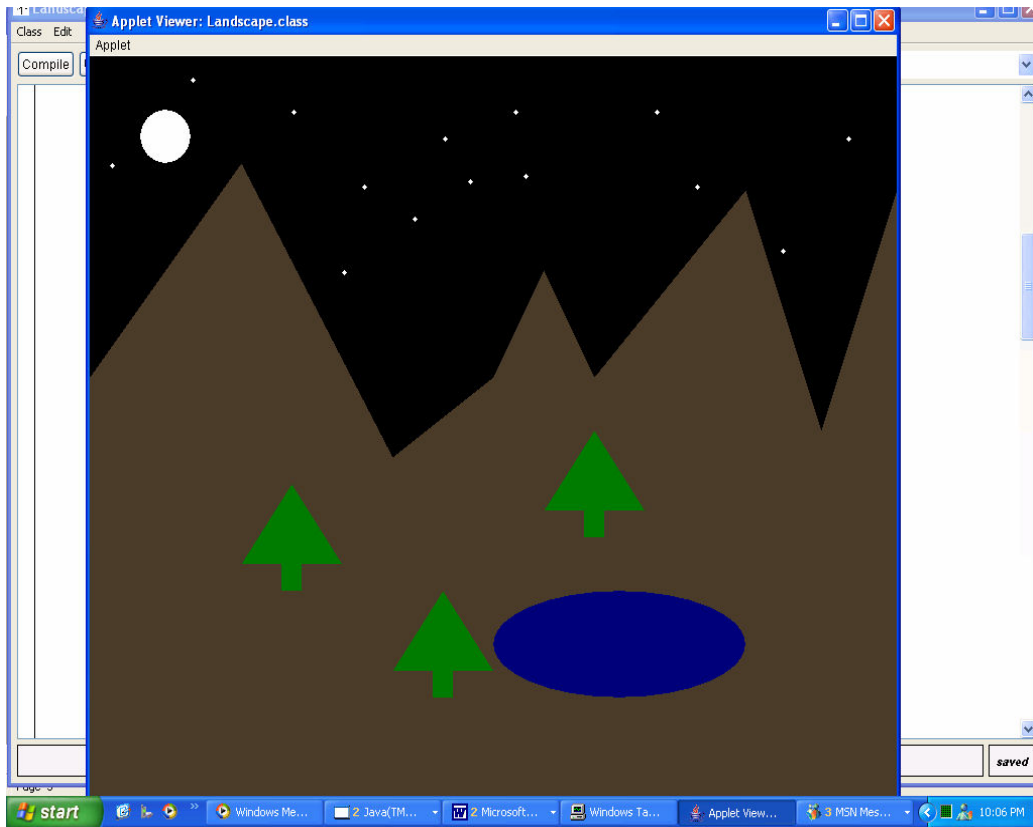
10.0 cups of cooked pasta

Preparation:

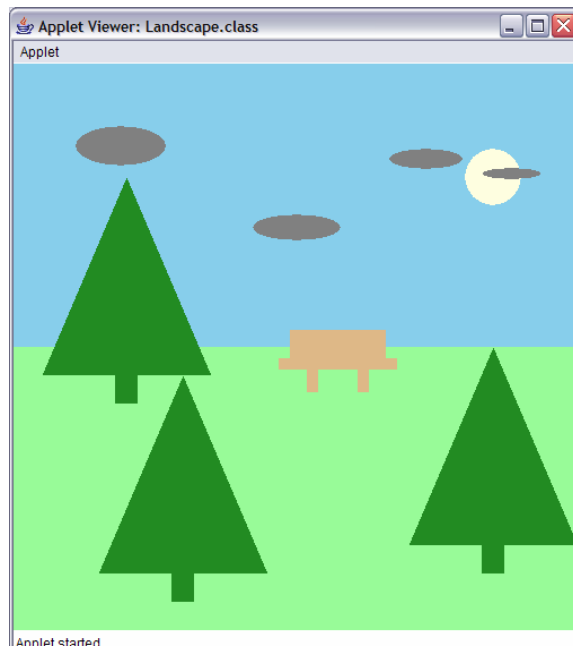
1. Pound chicken to 1/4 inches thickness between 2 sheets of plastic wrap.
2. Combine flour, salt, pepper, and basil; mix well
3. Heat oil and butter in a heavy non-stick skillet over medium high heat.
4. Dredge chicken in seasoned flour mixture
5. Cook chicken until lightly browned on first side (about 2 minutes)
6. Turn chicken and add mushrooms around chicken pieces.
7. Cook about 2 minutes more, until lightly browned; stir
8. Add abcdefg marsala wine to pan. Cover and simmer for 10 minutes.
9. Served over a cooked bed of pasta

Thank you Susie, for using the customized recipe center.

**Figure 3: Recipe Example 2**



**Figure 4: Landscape Example 1**



**Figure 5: Landscape Example 2**

## 5. Related Work

There are numerous papers and websites describing instructional methods and assignments for teaching introductory computer science. We mention just a few of these papers as related work. Turner and Zachary describe a course-long project used in just the second semester of introductory computer science<sup>11</sup>. Through this course-long project, students learned data structures, programming fundamentals, and applied software engineering practices. Our work is different in that the programming assignments described in this paper each span one to two weeks. The annual SIGCSE conference also supports a “Nifty Assignments” panel to share computer science assignments<sup>8</sup>. Each assignment presented by the panel has a “nifty” element, and the assignments described in this paper share the “nifty” aspect of being open-ended and encouraging creativity. Marks and others developed an introductory computer science course to introduce students to concepts and abstract thinking without using any programming<sup>6</sup>. Instead, case studies are used to teach algorithmic concepts and use existing software systems. Yet others use specialized tools and languages as platforms for introductory projects in computer science, such as MiniJava<sup>9</sup> and JKarelRobot<sup>3</sup>.

## 6. Conclusions

We described a set of programming assignments that encouraged creativity and conducted a study to learn of students’ perceptions and feelings of ownership. In particular, the paper explored the questions: 1. Did students complete extensions beyond the requirements?, 2. Did students take ownership and pride in their projects?, 3. What assignments did students like best/least and why?, and 4. What is the range of students’ projects for a single assignment? Students did, in fact, take the opportunity to extend their programs beyond the requirements. Sharing programs with friends and family and their written comments indicate that most took ownership of their work. Students tended to like assignments that were fun, interesting, and creative and tended to not like assignments that were less interesting, confusing, and time-consuming.

Instructors in other disciplines can make use of open-ended assignments that give students some flexibility to be creative. Many disciplines outside science and engineering have already taken this approach, as students are commonly asked to write literary analyses of works of their choosing and painting a scene of their choosing. Using assignments that result in unique solutions adds to the fun, interest, pride, and accomplishment of the students.

## Acknowledgments

The author thanks the students from the Spring 2006 and Autumn 2006 semesters for providing responses to the end-of-term survey. The author appreciates the help in designing some of the homework assignments: thanks to Dr. Steve Vegdahl for his assistance with the Recipe assignment and thanks to David J. Barnes and Michael Kolling (authors of *Objects First With Java*) for the code base “world of zuul” and Max Hailperin, Barbara Kaiser, and Karl Knight (authors of *Concrete Abstractions*) from which the adventure game assignment was adapted.

## References

- [1] Braten, Ivar. Vygotsky as Precursor to Metacognitive Theory: 1. The Concept of Metacognition and Its Roots. *Scandinavian Journal of Educational Research*, 35(3), 1991, 179 – 192.
- [2] Brown, Ann. Metacognition, Executive Control, Self-Regulation, and Other More Mysterious Mechanisms. *Metacognition, Motivation, and Understanding*. Hillsdale, New Jersey: Lawrence Erlbaum Associates, 1987, 65 – 116.
- [3] Buck, Duane and Stucki, David J. JKarelRobot: A Case Study in Supporting Levels of Cognitive Development in the Computer Science Curriculum. In *Proceedings of the ACM Special Interest Group in Computer Science Education Conference (SIGCSE '01)*, 2001, 16 – 20.
- [4] Duit, Reinders. The Constructivist View: A Fashionable and Fruitful Paradigm for Science Education Research and Practice. *Constructivism in Education*. Hillsdale, New Jersey: Lawrence Erlbaum Associates, 1995, 271 – 285.
- [5] Hmelo-Silver, C. E. Problem-based learning: What and how do students learn? *Educational Psychology Review*, 16, 2004, 235 – 266.
- [6] Marks, Joe, Freeman, William, and Leitner, Henry. Teaching Applied Computing Without Programming: A Case-Based Introductory Course for General Education. In *Proceedings of the ACM Special Interest Group in Computer Science Education Conference (SIGCSE '01)*, 2001, 80 – 84.
- [7] Norman, Donald. What Goes on in the Mind of the Learner. *New Directions for Teaching and Learning: Learning, Cognition, and College Teaching*. No. 2. San Francisco: Jossey-Bass, Inc., 1980, 37 – 49.
- [8] Parlante, Nick *et al.* Nifty Assignments. In *Proceedings of the ACM Special Interest Group in Computer Science Education Conference (SIGCSE '06)*, 2006, 562 – 563.
- [9] Roberts, Eric. An Overview of MiniJava. In *Proceedings of the ACM Special Interest Group in Computer Science Education Conference (SIGCSE '01)*, 2001, 1 – 5.
- [10] Southwest Educational Development Laboratory. The Practice Implications of Constructivism. *SEDLetter*, Vol. IX, Issue 3, August, 1996.
- [11] Turner, Joseph A. and Zachary, Joseph L. Using Course-Long Programming Projects in CS2. In *Proceedings of the ACM Special Interest Group in Computer Science Education Conference (SIGCSE '99)*, 1999, 43 – 47.
- [12] Vegso, J. “Drop in CS Bachelor’s Degree Production”, *Computing Research News*, 18(2), March, 2006, <http://www.cra.org/CRN/articles/march06/vegso.html>.