CS PRINCIPLES

March 3rd - March 31st, 2014

Performance Task: Create — Applications from Ideas

name:	
partner name:	

Project Description

Programming is a creative process that involves individual and collaborative effort to bring ideas to life through designing, developing, testing, and debugging computational artifacts. For this task, you and your partner will individually and collaboratively develop programs related to an area of focus of your choosing. You will be evaluated on your creativity and collaboration, as well as your ability to design algorithms, use abstraction, and program as you develop functioning computational artifacts.

Over the next month you will undertake a major programming and writing project. The project contains both individual and collaborative components. We will work on the project in class for a total of **13 class sessions** or **39 hours of work**. However, successful teams should expect to meet outside of class to ensure that they build all of their features and achieve all writing and programming requirements.

General Requirements

- You must develop and submit both an individual and a collaborative program for this task.
- You must work in pairs on the collaborative portion of this project.
- A group of three is only allowed if there are an odd number of students in the class.
- Your team must identify an area of focus, and both the collaborative and individual programs must share that same area of focus.
- Your team must collaboratively design and develop a program within the selected area of focus, and prepare a summary.
- You must work alone on the individual portions of this task.
- You must write the individual program, the reflection, and the summary by yourself. Both individual and collaborative work must demonstrate problem-solving skills and use complex programming constructs to complete this task.

Prepare and submit the following:

- Collaborative Submission
  For this submission, each group member will submit identical documents.
  - Develop a program along with your partner, in accordance with the Program Requirements (A).
  - Submit a written description of your group's program, as described in the Collaborative Summary section (B), including prompts 5-9.

- Individual Submission
  - Individually develop a program, as described in the Program Requirements (A).
  - Submit a written description of your individual program, as described in the Individual Summary section (C), including prompts 5-9.

- Submit a reflection on the group's programming process for the collaboratively developed program, as described in the Individual Reflection section (D).

Calendar

Day	Objective	Development	Deliverable
Day 1: March 3rd	Task Break Down and Brainstorming	Overview and Ideation	1. Performance Task Quiz 2. Brainstorm
Day 2:	Meet Partners	Ideation and Planning	1. Review Program Requirements 2. Collaborative Summary Prompts 1-3; 3. Peer Feedback Survey; 4. Individual Reflection Journal
Day 3:	Pair programming	Building	1. Submit Program Link; 2. Peer Feedback Survey; 3. Individual Reflection Journal
Day 4: March 10th	Pair programming	Iterate	1. **Significant Progress on your Program**; 2. Peer Feedback Survey; 3. Individual Reflection Journal; 4. Algorithm and Abstraction Survey
Day 5:	Pair programming	Iterate	1. **Significant Progress on your Program**; 2. Peer Feedback Survey 3. Video of Working Program
Day 6:	Collaborative Writing	Reflecting and Documenting	1. Collaborative Summary Prompts 5-9; 2. Peer Feedback Survey
Day 7: March 17th	Individual Ideation	Ideation and Planning	1. Individual Summary Prompts 1-3; 2. Peer Feedback Survey; 3. Individual Reflection Journal;
Day 8:	Individual Programming	Building	1. **Significant Progress on your Program**; 2. Peer Feedback Survey; 3. Individual Reflection Journal;
Day 9:	Individual Programming	Iterate	1. **Significant Progress on your Program**; 2. Peer Feedback Survey; 3. Individual Reflection Journal;
Day 10: March 24th	Individual Programming	Iterate	1. **Significant Progress on your Program**; 2. Peer Feedback Survey; 3. Individual Reflection Journal; 4. Video of Working Program

Day 11:	Individual Writing	Reflecting and Documenting	1. Individual Summary Prompts 5-9
Day 12:	Individual Reflection	Reflecting	1. Reflection Essay
Day 13: March 31st	Final Reflection Document	Packaging	Final Day for All Requirements. Final submission with all files prepared.

A. Program Requirements

All programs must meet the following requirements:

- All program source code must be submitted in two ways: (1) a zip file of original source code from the programming environment and (2) a pdf document of cut-and-pasted code, either as text or a screenshot.
- For each program, submit a video that displays the successful running of a portion of the program. The video must illustrate a portion of the intended purpose of the program and must have a maximum runtime of 1 minute. Submit the video using the following formats: .mov, .wmv, .mp4, or .avi.
- Use the free screen capture software on each operating system platform. For example, on Mac OS you can use free QuickTime screen recording (.mov). On Windows operating systems, you can use Windows Movie Maker with the output format Windows Media Video (.wmv). Other options include screencast-o-matic.com (.mp4, .avi) or camstudio.com (.avi).
- Each program must have an intended purpose that relates to the area of focus.
- Each program should be reasonably complex in demonstrating both the appropriateness of the programming language/environment you use and the significant work needed to create the intended functionality of your program.
    - Each program must include the basic programming elements of the language you use. For example, programs should demonstrate appropriate use of numbers, text, statements, mathematical expressions with arithmetic operators, logical and Boolean operators and expressions, and sets/list/other collections.
    - Each program should demonstrate the use of multiple abstractions in the programming language and creation of abstractions to develop and manage the complexity of the program (e.g., built-in and custom variables, data abstractions, functions/procedures, parameterization, APIs and libraries).
    - Each program should demonstrate the use of algorithms, including sequencing, selection, and/or iteration, as building blocks of the program.
- The program you produce individually must be significantly different from the one you write collaboratively and from your partner's individually produced program.
- The program you write independently can be written in a different language or in the same language as the program that you write collaboratively.

Insert the Link to Your **Collaborative** Snap! Program

Insert the Link to Your **Individual** Snap! Program

Insert the Link to the **Video** of your **Collaborative** Program

Insert the Link to the **Video** of your **Individual** Program

---

B. Collaborative Summary

Respond directly to each of the following prompts.

Collaborative Summary Prompts 1-3

1. State the area of focus your group chose to explore.
2. Describe the purpose of your collaborative program and how it relates to your area of focus. Your answer should be approximately 100 to 200 words.
3. Identify the language, programming environment (and which version you used), and the hardware and operating system on which you developed and tested your program.
4. **Respond to prompts 5–9.**

Collaborative Summary (5-9)

5. Describe how a user runs and interacts with your program. Provide details that allow a novice user to experience the full functionality of your program. This must include sufficient detail for a novice user to perform actions such as clicking on buttons or filling in text boxes. He or she should be able to run your program from beginning to end. Your description should be approximately 300 to 400 words and must include supporting visuals and non-textual representations as appropriate. Please upload non-textual representations as pdf attachments.

6. Demonstrate that your program illustrates abstraction. Your answer should be approximately 200 to 300 words and include the following:
    a. Identify and select a segment (or segments) of code from the program that illustrates the use of abstraction. Upload a pdf document with cut-and-pasted code, either as text or a screenshot. Your answer should be no more than one page of code.
    b. Explain how the selected code illustrates the use of abstraction.
    c. Explain how the code fits into the overall program you wrote.
    d. Include supporting visuals and non-textual representations if needed, and upload them as pdfs.

7. Demonstrate that your program illustrates a complex algorithm. Your answer should be approximately 200 to 300 words and include the following:
    a. Identify and select a segment or segments of code in the program that illustrates a complex algorithm. Upload a pdf document of cut-and-pasted code, either as text or a screenshot. Your

answer should be no more than one page of code.

b. Describe the purpose of the selected algorithm and how fits into the overall program you wrote.
c. Explain how the selected code implements the algorithm.
d. Include supporting visuals and non-textual representations if needed, and upload them as pdfs.

8. Explain how your program demonstrates the appropriateness and effectiveness of your chosen language and programming environment by describing your use of programming elements and how they allowed you to manage the complexity of the program. Your answer should be approximately 200 to 300 words.

9. Identify and discuss one significant runtime error or bug you encountered while writing the program. Your answer should be approximately 200 to 300 words. Include:
    a. What was the error or bug?
    b. What process did you use to discover it?
    c. What modifications did you make to the code to fix it?

C. Individual Summary

Respond directly to each of the following prompts.

Individual Summary Prompts 1-3

1. Describe the purpose of your individually developed program and how it relates to your area of focus. Your answer should be approximately 100 to 200 words.
2. How does your individual program relate to the collaborative program, and how is it different? Your answer should be approximately 100 to 200 words.
3. Identify the language, programming environment (and which version you used), and the hardware and operating system on which you developed and tested your individual program.
4. **Respond to prompts 5–9.**

Individual Prompts (5-9)

5. Describe how a user runs and interacts with your program. Provide details that allow a novice user to experience the full functionality of your program. This must include sufficient detail for a novice user to

perform actions such as clicking on buttons or filling in text boxes. He or she should be able to run your program from beginning to end. Your description should be approximately 300 to 400 words and must include supporting visuals and non-textual representations as appropriate. Please upload non-textual representations as pdf attachments.

6. Demonstrate that your program illustrates abstraction. Your answer should be approximately 200 to 300 words and include the following:
    a. Identify and select a segment (or segments) of code from the program that illustrates the use of abstraction. Upload a pdf document with cut-and-pasted code, either as text or a screenshot. Your answer should be no more than one page of code.
    b. Explain how the selected code illustrates the use of abstraction.
    c. Explain how the code fits into the overall program you wrote.
    d. Include supporting visuals and non-textual representations if needed, and upload them as pdfs.

7. Demonstrate that your program illustrates a complex algorithm. Your answer should be approximately 200 to 300 words and include the following:
    a. Identify and select a segment or segments of code in the program that illustrates a complex algorithm. Upload a pdf document of cut-and-pasted code, either as text or a screenshot. Your answer should be no more than one page of code.
    b. Describe the purpose of the selected algorithm and how fits into the overall program you wrote.
    c. Explain how the selected code implements the algorithm.
    d. Include supporting visuals and non-textual representations if needed, and upload them as pdfs.

8. Explain how your program demonstrates the appropriateness and effectiveness of your chosen language and programming environment by describing your use of programming elements and how they allowed you to manage the complexity of the program. Your answer should be approximately 200 to 300 words.

9. Identify and discuss one significant runtime error or bug you encountered while writing the program. Your answer should be approximately 200 to 300 words. Include:
    a. What was the error or bug?
    b. What process did you use to discover it?
    c. What modifications did you make to the code to fix it?

D. Individual Reflection

Working on your own, write a brief reflection essay in which you describe the collaborative process you used to create your shared program and your collaborative report. Your answer should be no more than 300-400 words. Include answers to the following questions:

A. How did you and your partner share or divide the work?
B. What was the most significant contribution that you, individually, shared with your partner that helped to create the collaborative program or helped to identify and solve a problem that was encountered when developing the program?
C. What was the most significant contribution that your partner, individually, shared with you that helped to create the collaborative program or helped to identify and solve a problem that was encountered when developing the program?

D. What was the most significant question you asked – or significant feedback you provided – that helped your partner review and revise his or her program?
E. What was the most significant question your partner asked – or feedback he or she provided – that helped you review and revise your program?

Learning Objectives

The Create — Applications from Ideas Performance Task addresses the following Computer Science Principles Learning Objectives (LOs):

LO 1.1.1 Use computing tools and techniques to create artifacts. [P2]
LO 1.1.2 Collaborate in the creation of computational artifacts. [P6]
LO 1.1.3 Analyze computational artifacts. [P4]
LO 1.2.1 Use computing tools and techniques for creative expression. [P2]
LO 1.3.1 Use programming as a creative tool. [P2]
LO 2.2.1 Develop an abstraction. [P2]
LO 4.1.1 Develop an algorithm designed to be implemented to run on a computer. [P2]
LO 4.2.1 Express an algorithm in language. [P5]
LO 5.1.1 Explain how programs implement algorithms. [P3]
LO 5.2.1 Use abstraction to manage complexity in programs. [P3]
LO 5.3.1 Evaluate a program for correctness. [P4]
LO 5.3.2 Develop a correct program. [P2]
LO 5.3.3 Collaborate to solve a problem using programming. [P6]

LO 5.4.1 Employ appropriate mathematical and logical concepts in programming. [P1]