CMPSC 111
Introduction to Computer Science I
Spring 2016

Final Project: Real-World Applications of Computer Science

Introduction

Throughout the semester, you have learned more about the basics of computer science and Java programming by studying, in a hands-on fashion, topics such as data and expressions, the use and creation of Java classes, conditionals and loops, and arrays. This final project invites you to explore, in greater detail, a real-world application of computer science. You will learn more about how to use, implement, test, and evaluate different types of real-world computer software. Since you will complete the final project with a partner, you will also learn more about how the Git version control system can effectively support collaborative software development.

Your project should result in a detailed report that includes all of your source code, in addition to written materials and technical diagrams that highlight the key contributions of your work. The report should include a description of why the chosen topic is important and discuss the implementation and/or experimentation that you undertook. The written material should be precise, formal, appropriately formatted, grammatically correct, informative, and interesting. The source code that you write must be carefully documented and tested. If you install and use existing computer software (e.g., a Java library for natural language processing), the steps for installation and use should be clearly documented in your report. Also, the report must explain the steps to run your own Java program. Finally, your paper must detail the work completed by each member of your partnership; individual contributions should also be reflected in the Git repository's log.

Description of the Topics

Each partnership is invited to pick one of the following projects. Please note that a partnership selecting the student-designed project must first discuss the idea with the course instructor, during today's laboratory session, and receive feedback and then final approval. Please note that you and your partner are fully responsible for ensuring the feasibility of the project that you propose.

1. **Cryptography and Cryptanalysis**: Explore a topic in the fields that make up the "art and science of sending and decoding secret messages". This project invites you and your partner to implement and test several cryptography and/or cryptanalysis systems. To start, you should investigate, implement, and test ciphers such as the Caesar and Vigenere ciphers. Then, you should use your ciphers to demonstrate that you can successfully send secret messages through, for instance, an email server. In addition to creating and testing these Java programs, your report should include a detailed explanation of how your chosen algorithms work.

2. **Computer Graphics**: Potentially using your textbook's sections on computer graphics as a starting point, implement a complete program that displays computer graphics. Students who select this project should consider ways in which the graphics can, for instance, represent realistic entities, support interactivity, and properly adhere to artistic principles of color, light, and layout. In addition to furnishing the project's source code, your report should include a detailed artistic, technical, and/or mathematical commentary on the final graphics.

3. **Performance Evaluation**: Since it is often important to implement computer software that exhibits acceptable time and space overheads, this project invites students to use and/or extend performance evaluation software like Caliper. After finding, reading, and understanding textbooks and research papers on this topic, students who pick this project should identify a focus area and create a benchmarking framework. The final version of the framework should allow students to measure the performance of computations in Java, report those measurements to the user of the benchmarks, and support informed decision making about design and implementation trade-offs. Along with including the source code of the benchmarking framework, this project invites students to write a performance evaluation report.

4. **Natural Language Processing**: This topic invites students to implement, test, and use Java programs that can analyze natural language text that has been written and saved in a computer-based format. Using and/or extending existing frameworks such as LingPipe, this project asks students to investigate how software can complete real-world tasks such as finding and classifying the names of people mentioned in online news articles, automatically categorizing Twitter search results, and tagging the parts of speech in a full paragraph of English language text. Students who select this project will implement and/or extend a natural language processing system, evaluate its efficiency and effectiveness, and write a report detailing both their implementation experiences and the empirical results.

5. **Software Testing**: In response to the fact that real-world software often contains serious defects, this project encourages students to learn more about techniques that can find program errors before they are delivered to end-users. Students who choose this topic will investigate software testing tools, such as JUnit, and then create a software system that includes a test suite. In addition to implementing Java classes and methods that provide the main functionality, you will, as part of this project, also write tests that assess the correctness of the aforementioned methods. If your team picks this project, you should submit programs and test suites in addition to a report that explains your approach to software testing.

6. **Computer Music**: Potentially leveraging existing computer music libraries, such as JFugue, this project invites students to implement a Java program that can create and play music. After learning about the Java classes and methods provided by a library like JFugue, students selecting this project should write Java programs that can generate musical instrument digital interface (MIDI) files. After creating a MIDI file, you can use Ubuntu's `timidity` program to play the sound on a computer. In addition to implementing the required Java programs, this project requires you to write a report that explains how JFugue works and gives both a technical and a musical commentary on the music your programs generate.

7. **Student-Designed Project**: Students will develop an idea for their own project that focuses on one or more real-world topics in the field of computer science. After receiving the course instructor's approval for your idea, you will complete the project and report on your results.

Project Requirements

To ensure that you and your partner have mastered the concepts discussed in this course, your project's source code should adhere to the following requirements. These requirements may be modified, at the discretion of the course instructor, only if a student receives prior permission

and documents this approval in the final report and the source code. Without prior approval, all submitted programs should contain at least three Java classes that consist of at least five methods in addition to the `main` method. The final project's source code should include examples of the declaration and use of at least three different data types (e.g., `int`, `String`, and `boolean`). The program should also include conditional logic, in the form of `if` or `switch` statements, and one or more iteration constructs, expressed as `while`, `do-while`, and `for` loops. Finally, the programs must also use either an array, an `ArrayList`, or both of these data containers.

The teams, consisting of two students, must create a new version control repository for the final project. The name of the repository must adhere to the following naming convention: "`cs111S2016-<first-user-name>-<second-user-name>`" where both "`<first-user-name>`" and "`<second-user-name>`" correspond to the user name of the Allegheny-approved email address for a student in this course. Please make sure that the course repository, which was initially created by one member of the team, is shared with both the other team member and the course instructor.

Final Project Deadlines

This assignment invites you to submit printed and signed versions of the following deliverables:

1. **Project Assigned and Project Proposal:** Wednesday, April 6, 2016

   After meeting with the course instructor and your partner, pick a topic for your final project. Remember, if your team selects the student-designed project, you must first have your project approved by the course instructor. Next, make sure that you create a Git repository that can be accessed by the instructor. Finally, write and submit a one-page proposal for your project. While you can use the project descriptions on the previous pages as a starting point, your proposal should have an informative title, an abstract, a description of the main idea, an initial listing of the tasks that you must complete, and a plan for completing the work.

2. **Status Update and Project Demonstration**: Wednesday, April 27, 2016

   As you continue working on your project, please submit a one paragraph status update in printed form and through your Git repository. In addition, you should give a demonstration, during the laboratory session, highlighting the most important code that you have finished.

3. **Final Project Due Date**: Monday, May 2, 2016 by 12 noon

   You should submit the final version of your project, in printed form and through the Git repository. This submission should include all of the relevant source code and output, the written report, and any additional materials that will demonstrate the success of your project. While you are encouraged to turn in the final project before the classroom session ends on the due date, students must submit the assignment no later than 2 pm on the due date.

In adherence to the Honor Code, students should complete this assignment while exclusively collaborating with the other member of their team. While it is appropriate for students in this class—who are not in the same team—to have high-level conversations about the assignment, it is necessary to distinguish carefully between the team that discusses the principles underlying a problem with another team and the team that produces an assignment that is identical to, or merely a variation on, the work of another team. Deliverables from one team that are nearly identical to the work of another team will be taken as evidence of violating Allegheny College's Honor Code.