Lesson 1.XX — Open-Ended Programming Project (Lesson 1.07 Alternative)

Overview

Objectives — *Students will be able to...*

- **Ideate and construct** a program containing method calls and static methods.

Assessments — *Students will...*

- **Submit** a complete, functional program by the end of class

Homework — *Students will...*

- **Check** class notes for completion, adding daily summaries if needed.
  - Students may use the book to supplement their notes if needed.
  - **All students must turn in notes for each day of class** (even if they were absent).

Emphasize with students...

Big Ideas - Personal design interests require the evaluation and refinement of skills Computers are great at doing repetitive tasks. In computational thinking, we use the words "decomposition", "pattern recognition", "abstraction", and "algorithm design". Think of the problem you are given. Break down the problem into manageable chunks. Are there any parts that show patterns, or similarities? Let's take advantage of the computer to help us do "similar" looking tasks.

In daily life, human beings enjoy repetition and patterns in many areas of life and nature. For the written word, repetition is useful for the learning of songs or poetry. It's a literary technique. Repetition enhances significance and meaning.

Consider the national anthem of USA Did you know that there is an English version, and a Spanish version?

Imagine that you are creating a national anthem medley for your class to perform at a special event. Your challenge is to include a combination of language versions. How can you use static methods to design this musical piece?

Materials & Prep

- **Projector and computer** (if you are able to/opt to use Eclipse with your students)
- **Student self-help system** such as C2B4 ("see two before seeing me") or student pairing

Make sure you are set up to grade student notebooks today while the students work on the project. If possible, you should only collect 3–5 notebooks at a time so students have their notebooks available to reference during programming time.

Pacing Guide

Section	Total Time
Bell-work and attendance	5min
Introduction & classroom procedure review	10min
Programming project	30min
Students trade work, check, and turn in	5min

Procedure

The second week part of this unit will be spent on reinforcing concepts and applying the tools, procedures, and code that were introduced last week. While these classes require little prep before class, you should set up a system that will allow students to help themselves and each other so you aren't running around the computer lab the whole time.

If your computer time requires you to move to another room or to change seating, you should teach and/or review those procedures before introducing the lab material. If you expect students to submit assignments electronically, you should also model and review those procedures before students begin work.

Bell-work and Attendance [5 minutes]

Introduction and Classroom Procedure Review [10 minutes]

1. Introduce the program assignment, taking a moment to talk strategy with your class.

   ASSIGNMENT: Sometimes we write similar letters to different people. For example, you might write to your parents to tell them about your classes and your friends and to ask for money. You might write to a friend about your love life, your classes, and your hobbies, and you might write to your brother about your hobbies and your friends and to ask for money.

   Write a program that prints similar letters such as these to three people of your choice. Each letter should have at least one paragraph in common with each of the other letters. Your main program should have three method calls, one for each of the people to whom you are writing.

   TIPS: Try to isolate repeated tasks into methods. Include comments in with your code so others can easily understand what the code is supposed to do.

2. Ask your class for suggestions as to how to tackle this programming problem. Students should suggest drawing a structural diagram, building the program one method at a time (iterative development), and following the correction steps on their personal algorithms (debugging).

Programming Project [30 minutes]

Reference the exercise above, where you write (ie, output, or print) similar letters to different people. The main program has common parts. In the main method, make calls to static methods that do other variations.

Examples of text-based output that have this characteristic: - A standardized form, or letter - Lyrics of a song or rhyme that has: * Lines that are repeated at each verse, e.g. "There Was an Old Lady" (Reference text book Chapter 1, Project #2) * New lines are added onto existing old lines, e.g. "Twelve Days of Christmas", "The House That Jack Built". (Reference text book Chapter 1, Project #3, #5.)
- Lyrics of a song that includes words (or sections) in different languages - A cheer or chant for a school team - A medley of different songs (verse and chorus) put together Static methods can simplify output of the above task. For example, you can use methods for each verse, and for repeated text.

What other types of output have repeated patterns? What are other types of printed output that can benefit from static methods? Choose one to implement.

Students trade work, check, evaluate and turn in [5 minutes]

At the end of class, have students briefly look at each other's projects and review their work before they submit.

Evaluation Question: How many lines of code did this program "save" by calling static methods, instead of calling print statements to output every single line of text?

Accommodation and Differentiation

If you have students who are speeding through this lesson, you should encourage them to tackle programming project #4 in the text book.

Forum discussion

Lesson 1.XX Programming Project (TEALS Discourse account required)