

Programming Assignment Ideas from CS 1705

Stephen Edwards
Virginia Tech, Dept. of Computer Science
All materials at:
http://web-cat.cs.vt.edu/CsEdWiki/Virginia_Tech_CS1

Recurring Themes

Although these assignments aren't all nifty, there are some common trends that we've come to use to our advantage:

- Testing:** We require our students to write test cases for all of their code. BlueJ helps by allowing them to interactively record test cases even before they program.
- I/O:** We never write programs that use `System.out` or `System.in` directly. Instead, all I/O is through `BufferedReader`s or `PrintWriter`s, and we provide simple wrappers for one-call creation of such objects. This is **critical for writing convenient test cases**, but also means it is easy to operate on a wide variety of I/O sources, including URLs.
- Connection to the real world:** We try to relate each assignment to real-world experiences or real-world programming behaviors, even if only in a small way—reading information from the web, processing real-life data, mirroring aspects of programs students have used themselves, etc.
- Identifiable learning objectives:** We try to spell out for our students exactly what the learning objectives are for each assignment. Even if students don't enjoy an assignment, they know why we are asking them to do it.

These ideas do not guarantee nifty or compelling assignments, but they have helped us write better assignments.

Counting Beepers

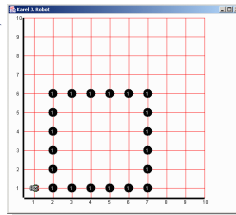
Lab assignment (2 hrs)
Summary: Program Karel J Robot to count an unknown arrangement of beepers
Objectives: If statements, recursion, while loops
Nifty: The world is initialized from dynamically generated PHP output via a URL

Beeper Squares

Lab assignment (2 hrs)
Summary: Program Karel J Robot to pick up an unknown number of beepers and lay them out in a square of an appropriate size
Objectives: Method parameters, local variables
Nifty: The world is initialized from dynamically generated PHP output via a URL

Rot-13 Decryption

Lab assignment (2 hrs)
Summary: Read/write one character at a time and decrypt Rot-13 text
Objectives: Text streams and basic text I/O
Nifty: Uses `BufferedReader` & `PrintWriter`, so input can come from many sources, including the web



Collecting Beepers

Lab assignment (2 hrs)
Summary: Direct Karel J Robot to pick up piles of beepers
Objectives: Introduces objects, classes, and method calls
Nifty: students must interactively record their first test cases

Planting Beepers

Lab assignment (2 hrs)
Summary: Program Karel J Robot to pick up beepers and plant them in a row
Objectives: If statements, methods, recursion, while loops
Nifty: The world is initialized from dynamically generated PHP output via a URL

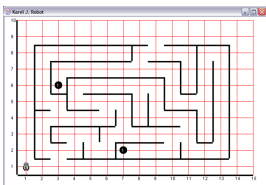


Rot-13 Decryption

Lab assignment (2 hrs)
Summary: Read/write one character at a time and decrypt Rot-13 text
Objectives: Text streams and basic text I/O
Nifty: Uses `BufferedReader` & `PrintWriter`, so input can come from many sources, including the web

Walking a Maze

Lab assignment (2 hrs)
Summary: Program Karel J Robot to solve a fixed maze
Objectives: Introduces subclasses, defining methods
Nifty: The world is initialized from a URL via a PHP script; the maze is fixed, but beeper locations vary

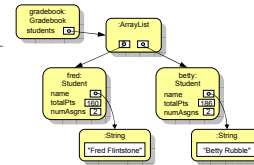


Maze Solver

Program assignment (2 wks)
Summary: Program Karel J Robot to solve a dynamically generated maze
Objectives: Subclasses, methods, if statements, recursion
Nifty: The world is initialized from a URL, and a PHP script dynamically generates a new maze for every run

Line Counter

Lab assignment (2 hrs)
Summary: Implement a simple version of the Unix `wc` command
Objectives: Text streams, character and line input
Nifty: Uses a `BufferedReader` and can be targeted at the course's web home page, simplicity is deceptive



Grade Book

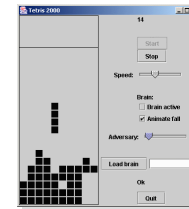
Lab assignment (2 hrs)
Summary: Build a simple set of classes to implement a grade book
Objectives: constructors, containers, `ArrayList`s
Nifty: Inspired by textbook, natural lead-in for iterators

Spelling Checker

Lab assignment (2 hrs)
Summary: Write a `BufferedReader`-based spelling checker that reads lines, splits them into words, and checks them against a list loaded from a file
Objectives: text I/O, string splitting, container classes
Nifty: Can replace the "starter" word dictionary with something much bigger, inspires maps, can be directly applied against live URLs like the course's web pages

Time Table

Lab assignment (2 hrs)
Summary: Write a simple weekly calendar that can store events scheduled on the hour
Objectives: 2D arrays, writing interacting classes
Nifty: Potentially very open-ended, Some students like to play with printing scheduled in HTML



DUML Translator

Programming assignment (2 wks)
Summary: Write a translator that reads simple text markup and translates it into HTML
Objectives: character I/O, state machines, complex logic and state
Nifty: Use a simple utility function to pop up the result in your web browser so you can see the formatting

Karel Interpreter

Lab assignment (2 hrs)
Summary: Implement an interactive command line interpreter to control Karel, using a map of command objects that implement a common interface
Objectives: Interfaces, maps, line-oriented input, simple interpreters
Nifty: Students write their own interpreter in 2 hrs, and see how extensible the map + interfaces combo is

Scriptable Calculator

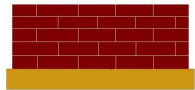
Lab assignment (2 hrs)
Summary: Students are given a buggy class (with weak tests) and a partially complete subclass; they must complete and debug the code
Objectives: Debugging skills, code reading
Nifty: End result is an extensible interpreter for textual calculator commands

Web Log Analyzer

Program assignment (2 wks)
Summary: Inspired by text, but modified to use real Apache logs from our dept server
Objectives: Interfaces, 2D arrays, using code libraries
Nifty: Works with real-world data that the students can relate to

Debugging Bricks

Lab assignment (2 hrs)
Summary: Read, test and debug existing code (no code writing)
Objectives: Testing and debugging skills
Nifty: Inspired by textbook, some bugs are hard to find, students race to see who can find them all



Mine Sweeper

Lab assignment (2 hrs)
Summary: Implement the "board" for this game (other classes provided), and play the result
Objectives: 2D arrays, working with libraries
Nifty: Students create a playable game in a short time



URL Harvester

Program assignment (2 wks)
Summary: Write a program that crawls an interconnected set of web pages and prints out the links they contain
Objectives: Text streams, character and line input
Nifty: Uses a `BufferedReader` and can be targeted at the course's web home page, simplicity is deceptive

Tetris Brain

Program assignment (2 wks)
Summary: Write an "AI" computer player for tetris, based on one of the Stanford "nifty" assignments
Objectives: Interfaces, nested loops, working with existing code, reading docs
Nifty: Playable game, plus graphical animation of computer player; students informally compete for best tetris scores

Adventure

Program assignment (2 hrs)
Summary: Like Zuul, but using abstract based classes: students extend a framework, rather than modify existing code
Objectives: Abstract classes, simple frameworks, etc.
Nifty: Very open-ended, students vote on the most creative adventure game submitted in a contest

