Lecture 42: Wrap-Up & Summary

CS 51G
Spring 2018
Kim Bruce

Announcements

- The final exam is on Wednesday, Dec 14, from 9 to noon.
- Sample Final Exam on exams web page
  - No solutions
- Copies of the Objectdraw, File, and GUI cheat sheets for the exam will be available as needed.
- All Grace but one question on analyzing/ translating Python to Grace.

Final Exam Topics

- Recursion
- Lists & Matrices
- Inheritance
- Using GUI components
- Strings
- Searching and Sorting
- Files & Exceptions

Final Exam Topics

- Explaining Python and/or Python to Grace
- Other materials from earlier in the semester (but most emphasis on above topics)

- Office hours/review session to be announced (perhaps Friday afternoon?)

Final Exam

- Time likely won't be an issue, but will likely find it very challenging. It will require a lot of studying.
- How to study:
  - Understand your own programs
  - Understand my sample code
  - Work problems from back of text -- solutions on line!.
- When done studying, try final on line
- Studying in groups is effective if all participate

What do you Know?

- Good exposure to object-oriented programming, but still lots to learn about design.
- Should find it easy to transition to languages like Python, Java, C#, Swift
- C and C++ much harder and more complicated.
- CS 54 will teach functional language
- CS 62: Java, CS 105: some C,

Principles

- KISS: Keep It Simple Stupid (US Navy)
  - Make everything as simple as possible, but not simpler
    - Einstein
- Test every line of code as it is written
  - Test-driven development
- Keep public interface of classes as small as possible so can make changes w/out affecting others.

Problems Requested

- Exercise 11.1.3
- Exercise 16.7.6. & Question 5 from sample exam

Anonymous Functions In Grace

- Expression of form {n: Number -> ...} is anonymous function.
  - Object with an apply method:
    - {n: Number -> print(n * n)}.apply(7) prints 49
  - {...} is parameterless function {x > 0}.apply gives true/false
  - Use in for loops
    - for(1..10) do {n: Number ->
      print(n*n)}
    - for is a method: method for (e:List⟦T⟧) do (b: Block1D⟦T⟧) where Block1C⟦T⟧ has apply function that takes argument of type T and returns Done.

Add Functionality

- Can add new control structures to Grace
  - Add do{...} while {...} method:
    - method do{blk:Block1D[[T]]} while(cond: Blk0[[Boolean]]) → Done {

      blk.apply
      while(cond) do (blk)

Questions?