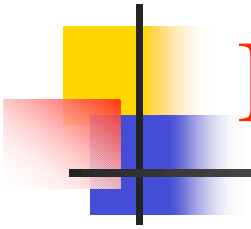
COMP 211
Principles of Program Design
Spring 2009

Prof. Robert "Corky" Cartwright

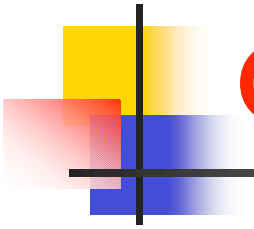
Department of Computer Science

Rice University



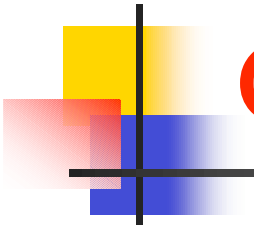
Instructor Information

- Corky Cartwright [cork@rice.edu, cork@oplink.net]
  - Duncan Hall (DH) 3104, 713-348-6042
  - www.cs.rice.edu/~cork
  - Office hours: MWF 1:00-1:55, and by appointment
- Walid Taha [taha@rice.edu]
  - Duncan Hall (DH) 3103, 713-348-5718
  - www.cs.rice.edu/~taha
  - Office hours: TBA
- Zung Nguyen [dxnguyen@rice.edu]
  - Duncan Hall (DH) 3098, 713-348-3835
  - www.cs.rice.edu/~dxnguyen
  - Office hours: TBA
- Teaching Assistants: Cherif Salama + Angela Lu (?) + Jun Inoue (?)



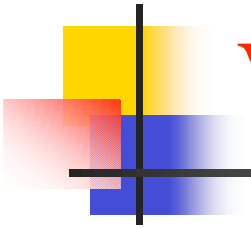
Course Materials

- Course web page:
  www.hope.cs.rice.edu/twiki/bin/view/Teaching/211
- Since Comp 211 is a new course, googling for "comp 211" does not work.
- If you forget the long URL given above, you can simply go to www.cs.rice.edu/~cork and follow the link to Comp 211.
- Course information like TAs, office hours, *etc*. are covered on the course web site. Some of that information is still TBA.



Course Mechanics

- HW0 posted on the course wiki (web site) is *due* on Wednesday!
  - Short entry survey (less than 2 minutes)
  - Sign up for a Twiki account
  - Lab section preference
  - Pick a lab partner
  - Download PLT Scheme 4.1.3
- Optional math enrichment section to be arranged. Time will be chosen Wednesday in class. This section will discuss mathematical foundations of program design. This material is optional and ungraded.

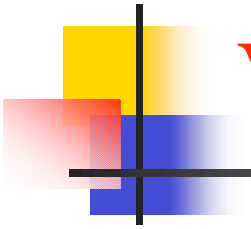


Why Focus on Program Design?

- *Program Design* is the core of Computer Science
- Why not *Algorithms?*
- Software is the dominant artifact of modern civilization
  - "Code is Law" [Lessig]

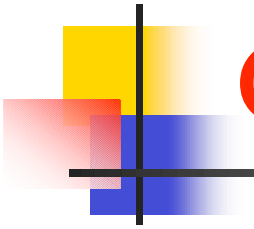
    http://harvardmagazine.com/2000/01/code-is-law.html
    - Code regulates many aspects of our lives

      e-media, e-commerce, e-voting, e-medical records ..
  - Code is emerging medium for expressing knowledge (HTML, PDF)
  - Code is omnipresent in manufactured goods
    - Airplanes, cars, blenders, phones, toys, greeting cards, …
- Program Design is *intellectually challenging*



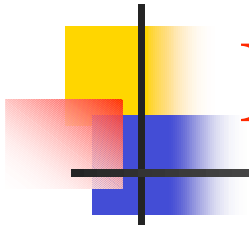
Why COMP 211?

- Repackaging of innovative curriculum for better marketing
  - Comp 210/212 developed at Rice, with major NSF funding
  - DrScheme, DrJava, *How to Design Programs*
- How it is different from other introductory courses?
  - Focus on principles of design, not on details of a particular language or software platform
    - Few programming constructs (in Scheme and Core Java)
    - **Data definitions (types)** drive the design process
      - Note: data definitions are **not** "data structures"
  - Not a typical programming course, and
  - Not a Scheme/Java language course
- Program design is **not** coding (*e.g.*, iterators not loops)



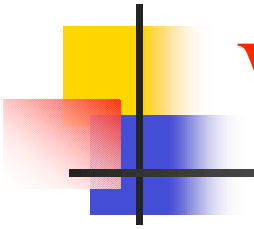
Course Overview

- Functional program design in Scheme (6 weeks)
  - Data-directed (functional) program design 2-12
  - Algorithm design 13-15
  - Applied functional programming 16-18
- Object-oriented (OO) program design in Java (9 weeks)
  - Rudiments of the OO programming model 19-21
  - Data-directed OO program design 22-24
  - OO functional programming 25-27
  - Advanced Java constructs (inner classes, generics) 28-30
  - Fundamental data structures and algorithms 31-39
  - Event-driven programming and GUIs 40-42
  - Concurrency (bonus material- not tested) 43-45



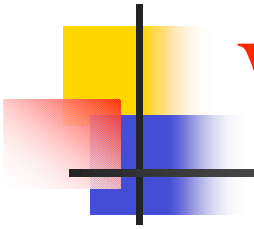
Design Patterns Covered

- union/composite/interpreter
- singleton
- command/strategy
- factory method
- visitor
- model-view-controller
- decorator?
- template method?
- adapter?



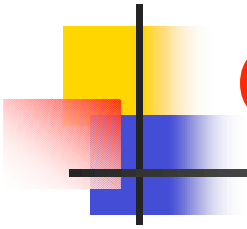
Why Scheme?

- Functional programming is the cornerstone of good programming design.
- Good notation (provided by a functional language derived from mathematics) make functional programming easy.
- Scheme is the simplest functional language and we will use only the core constructs:
  - Function and constant definition
  - Function application
  - Conditionals
  - Structure definitions
  - Local definitions (blocks) and assignment
- Simple formal semantics: rewrite program source text.
- Very good pedagogic IDE: DrScheme.



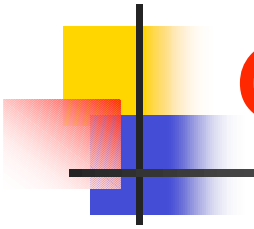
Why Java?

- Object-oriented (OO) programming is a powerful generalizaton of functional programming that decomposes programs into a collection of code units called classes.
- Classes support incremental test-driven development.
- Java/C# now dominate application programming
  - Only Java is almost completely platform independent: "write once; run anywhere."
  - A good (but not great) OO language.
  - Efficiently implemented except for VM startup and memory footprint.
- Very good pedagogic design environment (IDE): DrJava



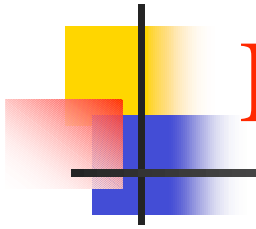
COMP 211 Prepares You to…

- Think about program design without focusing on language features.
- Learn deeper concepts of computing:
  - Programing languages (design and implemenation)
  - Formal methods (program semantics, verification, formal logic)
  - Algorithms (including ideas central to artificial intelligence techniques, data-mining, bioinformatics)
  - Systems (networks, operating systems, compilers)
  - Software engineering (application architecture, test-driven development, unit testing, refactoring)



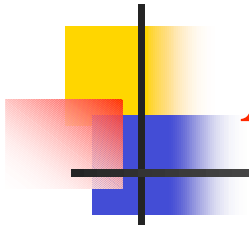
Grading

- Homeworks (50%)
  - Usually once a week, Monday-to-Monday,
  - Work jointly in teams of two. Do **not** divide work up.
  - No late homework will be accepted, except for 5 *slip* days to be used during the term. A fraction of a day counts as a full day. Advice: hoard your slip days until the end.
- Exams (50%)
  - Sample exams will be available online.
  - Take home, pledged, closed book.
  - First exam during week 7 counts 20%
  - Second exam during week 15 counts 30%
- .



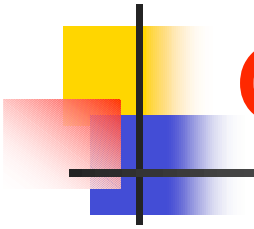
How to Succeed

- Do the reading before class
  - This will help you understand my lectures.
- Attend class
  - Both reading AND lectures are required.
  - Exam questions will emphasize low-attendance classes
- Attend the mandatory labs (once a week)
- Take homework assignments seriously
  - A program that simply "works" is worth little credit.
- Use office hours
  - Having questions is a sign of intelligent life



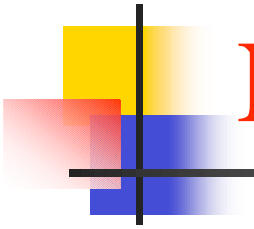
About Your Instructors

- Our research programs are concerned with
  - Improving programming technology including
    - Language design
    - Programming tools: IDEs, “soft” typers, testing frameworks
    - Programming pedagogy
  - Improving programmer productivity, using
    - Automatic program generation,
    - Lightweight formal verification (type systems),
    - Higher-order typed languages (ML, Haskell, Java+, Fortress)
  - Improving productivity of people building:
    - Real-time and embedded systems,
    - Hardware (microprocessors or “chips”),



Copyright

- COMP 211 lecture notes are copyrighted by Profs. R. Cartwright, W. Taha, and Z. Nguyen, Rice University.
- COMP 211 students can copy and modify these materials freely as long as this slide is preserved
- Commercial use requires written permission.
- Publicly available materials for the course are licensed under the Creative Commons (CC) license
  - See creativecommons.org basic idea, and then course Twiki for details



Next Lecture

- Make sure you have done Homework 0
  - Already posted online on web-page
  - Due next class (Wednesday)
  - Help available on Wednesday with installing DrScheme
- Next class
  - More details on how to create and submit homeworks
  - The basics of programming