6.090
Building Programming Experience

Lecture 1
1/10/2007

Outline

• Welcome!
• Goal: Help you prepare for 6.001
• Logistics
• Scheme Introduction

Logistics:

• Course Web Page:
• I’ll be sending class list to Registrar
  – You don’t need to pre-register
• Laptops

Classes

• 9 Classes
• 1-4 pm
• First ~half lecture
• Short Break
• Second ~half lab
• One Quiz in class next Friday

Homework

• 7 assignments
• “In order to pass the class, you must do all of them. Homework will be not be graded; any commentary is to point out things that you did well or could do better. In order to have "done" a homework, you must have put a significant amount of effort into completing it; all the assigned problems need not be working.”

Homework

• 6.001 and 6.090 have been taught before
• Many problems similar
• Please do not use any old solutions you find, they will not help you learn to solve problems
Collaboration

• Write your own answers
• Say who you worked with
• Some non-collaborative problems
  – Discuss these only with course staff

Laptops

• Install DrScheme: [http://www.drscheme.org](http://www.drscheme.org)
• Choose Language:
  Menus: Language->Choose Language
  PLT/ Pretty Big

Goal

• Make a computer do what we want
• Computer only follows very simple and exact instructions
• Need a way of specifying instructions

Scheme

• Programming Language to describe the computational process
• Basics of the language today

Scheme Basics

• We need three things:
  – Way to represent data
  – Basic operations to perform on data
  – Means of combination to do more complicated operations

A simple computation

• We want to compute 3 + 5
• All computations are describe by expressions that are evaluated
• “Evaluate the expression 3 + 5” really means
  “Perform the computation to add 3 and 5, then return the value of the expression”
Our Tool

- DrScheme
- Type Expressions into the lower window
- Scheme will evaluate the expression, then display the value of the expression
- Saving your work
  Save Other -> “Save definitions as text”

Simplest Expression

- Type 3 into the bottom window
- What happens?

Simplest Expression

- Type 3 into the bottom window
- What happens?

- Scheme has
  – Read the expression you typed
  – Evaluated it
  – Printed the result

3 as an expression

- A number is an example of a self-evaluating expression
- The value of the expression is the same as the expression
- Examples:
  – Numbers – 1, 2, 10.5, 100
  – Strings – “Hello, World!”
  – Booleans - #t #f
  – Procedures

Types

- Every value has a type
- Some operations are only defined for certain types
  – Addition defined for numbers
  – #t + #f is not defined in Scheme

Procedures are a type

- A primitive procedure is a built-in operation to manipulate objects
  – Numbers: +, -, *, /, <, =
- Built in procedures have names
- Procedure evaluated by looking up the name in a special table
- Type + into the evaluator window
Back to 3 + 5

- In Scheme, we compute 3 + 5 by applying the primitive addition procedure to the numbers 3 and 5
- Write this as (+ 3 5)
- Try it!

We can apply expressions recursively

- (* (+ 3 2) (- 5 1))

Names

- To solve interesting problems we'll need to use many different types and pieces of data
- Want to control code complexity
- Use abstract expressions by assigning names

Compute area of a circle

- Which is more readable?
  - (* (/ 22 7) 4 4)
  - (define pi (/ 22 7)
    (* pi (square 4))

  We've used names to represent data and computation, which abstracts away the computation

Evaluating Names

- Table inside the interpreter
- To evaluate a name, consult the table

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>pi</td>
<td>3.14159</td>
</tr>
<tr>
<td>+</td>
<td>Procedure</td>
</tr>
<tr>
<td>-</td>
<td>Procedure</td>
</tr>
</tbody>
</table>

Adding Names

- To add names to the table, use (define name value)
- Define is a special form
  - Something that looks like a combination but behaves differently
Evaluation Rule

- If self-evaluating, return value
- If a name, return value associated with the name in the environment
- If a special form, do something special
- If a combination, then
  - Evaluate all of the subexpressions in any order
  - Apply the operator to the values of the operands and return the result

Evaluate an Expression

- (+ 3 5)
- There are four steps that happen
  1.
  2.
  3.
  4.

Exercise

- Choose some simple computation
  - Area of a rectangle with sides of length 1 and 4
  - Length of hypotenuse of triangle with sides of length 1 and 4
  - Perimeter of a circle of radius 1
- Write out the Scheme expression to compute it
- Use define to make it cleaner

One last example

- (+ 3 5) --> 8
- (define fred +) --> undef
- (fred 4 6) --> 10

- How to explain this?
  - + is just a name
  - + is bound to a value which is a procedure
  - Second line binds the name fred to that same value