General Computer Science for Engineers CISC 106 Lecture 10

Roger Craig
Computer and Information Sciences
3/06/2009

Lecture Overview

- Lab (due Monday at 11:55pm) / Midterm Review (3/17)
- Sequence, selection, and repetition
- Recursion cont.
- Complex IF statement conditions

Sequence, selection, repetition

- Sequence (statements are ordered, 1, 2, 3…etc.)
- Selection (logical control, IF statement)
- Repetition (iteration and recursion)
- With these three control structures, you can program anything

Recursion cont.

- Simple functions
- f(x) = pi * x^2
- f (b, h) = ½ * b * h

- Functions involving repetition
- f (x) = sum of the squares integers from 1 to x

- Can do the above with a FOR loop. Can we do it recursively?

Recursion cont.

- f (x) = if x == 1, return 1^2
- if x > 1, return x^2 + f(x-1)

- Notice we subtract 1 from x when calling f(x-1). This allows us to get closer to our base case so that we can eventually finish.

- How to write that function in Matlab?

Fibonacci sequence (a famous recursive function)

$$F_n = \begin{cases} 0 & \text{if } n = 0; \\ 1 & \text{if } n = 1; \\ F_{n-1} + F_{n-2} & \text{if } n > 1. \end{cases}$$

F_0	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}
0	1	1	2	3	5	8	13	21	34	55	89

IF statements

- if statements
  anatomy of an if statement
- simple conditions
  complex conditions (using AND, OR and NOT)

IF statements

- If (control expression)
  Code to execute if ‘control expression’ is true
  else
  Code to execute if ‘control expression’ is false
  end

If there is no ‘else’ part, it may be omitted. E.g.

```
If (x > 2)
  fprintf(‘%f is greater than 2’, x);
end
```

Relational Operators

A < B	A is less than B
A > B	A is greater than B
A <= B	A is less than or equal to B
A >= B	A is greater than or equal to B
A == B	A is equal to B
A ~= B	A not equal B

Logic Operators

Operation	Operator	Example
NOT	~	~T = F
OR	| (pipe; shift backslash)	F | T = T
AND	&	F & T = F
XOR	xor	F & T = T

Table 3.3 Truth Tables for Logic Operators

Inputs		and		or		xor	not
l_1	l_2	l_1 & l_2	l_1 && l_2	l_1 | l_2	l_1 || l_2	xor(l_1, l_2)	~l_1
false	false	false	false	false	false	false	true
false	true	false	false	true	true	true	true
true	false	false	false	true	true	true	false
true	true	true	true	true	true	false	false

Control expressions

- Control expressions can be simple

```
If (x > 2)
   fprintf('%f is greater than 2', x);
end
```

- Or more advanced:

```
If ((x > 2) && (x < 4))
   fprintf('%f is greater than 2, but less than 4', x);
end

If ((x == 2) || (x == 4))
   fprintf('%f is either 2 or 4', x);
end
```

Logic Operators

- NOT is unary operator (it only acts on one thing, and flips it)
- AND, OR, and XOR are binary operators. (need to have a pair of things to compare.)

- OR is inclusive OR (both A and B can be TRUE for A OR B to be TRUE)

- XOR is exclusive OR (only one of A and B can be TRUE for A OR B to be TRUE)

Order of operations

1. Arithmetic operators (+, -, *, / etc.)
2. Relational operators (==, <, >, etc.)
3. NOT operator (~)
4. AND operators (& , &&)
5. OR and XOR operators (|, ||, xor)

As with arithmetic expressions, parentheses can be used to change the default order of evaluation.