General Computer Science for Engineers CISC 106 Lecture 15

Dr. John Cavazos

Computer and Information Sciences

03/16/2009

Lecture Overview

- Unix commands
- if statements
- arrays (or matrices)
- loops
- scripts
- functions

Midterm and Review

- Midterm review session, Gore 318
  - March 17 4:30pm-6:00pm
  - http://www.udel.edu/CIS/106/cavazos/09S/lectures/09S_midterm1review.pdf
- Midterm 1
  - March 18 (Class Time : Wednesday!)

Important Notes on Exam

- Write code from memory
- Study labs
  - Write code for labs
- Study Midterm review
- Attend review session

Unix Commands

- When you log into a UNIX terminal
  - You are in your home directory.
  - To see the files in your directory.
    - ls
  - To make an new folder/directory.
    - mkdir exampledir
  - To change directories.
    - cd exampledir
  - To go back one directory.
    - cd ..
  - To go back to your home directory.
    - cd

Basic if statements

- IF statements allow program to make choices whether a condition is met or not

```
if (expression1)
    statements1;
end

if (expression2)
    statements2;
end
```

IF/Elseif Statements

```
if (expression1)
    statements1;
elseif (expression2)
    statements2;
else
    statements3;
end
```

Major 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

If statements

- print “blue” if N <= 5
- print “red” if N > 5 and N <= 10
- print “green” if N > 10

If statements (cont'd)

```
if (N <= 5)
      fprintf 'blue\n';
end
if (N > 5 & N <= 10)
      fprintf 'red\n';
end
if (N > 10)
      fprintf 'green\n';
end
```

Arrays (aka matrices)

- All variables in matlab are arrays
- An array of one element is called a scalar
- A one dimension array is called a vector

x=3.14; ← scalar

a = [1,2,3,4,5]; ← vector

Arrays (aka matrices)

- x = 1:0.5:5

Now x is an array of numbers;

x = [1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0]

Arrays (aka matrices)

- A = [1, 2; 3, 4; 5, 6]
  - Creates a 3x2 array, 3 rows, 2 columns.
  - semicolon creates a new row.

$$A = \begin{matrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{matrix}$$

For Loops

- Used when you know how many times code is to be executed.
- Syntax

for <variable> = <start>:<increment>:<end>

- Variable is initially the start value
- At end of iteration variable changes by increment
- If value is not greater than end the loop runs again.

Example Problem

- I want to find the average # of widgets sold in 4 days

Day	# of widgets sold
1	15
2	22
3	20
4	18

- Widget(1) = 15
- Widget(2) = 22
- Widget(3) = 20
- Widget(4) = 18

- Avg = (Widget(1) + Widget(2) + Widget(3) + Widget(4)) / 4
  - This is easy for a small number of days.
  - What if we had a 1000 days?
  - We can use a for loop!

Example Problem

```
total = 0;
for i = 1:1:1000
  total = total+widget (i);
end
avg = total / 1000;
```

loop starts at 1

loop increments by 1

loop ends at 1000

A Loop Analogy

- The mail man/woman executes a loop.
- If they know the number of deliveries
- For loop

```
for delivery = start : next_delivery : end
    deliver_mail(delivery)
end
```

Scripts files

- Store commands in
- Variables are global, available after you call script file

Scripts files

```
sumIt=0;
for current=1:finish
    if (mod(current,2)==1)
        sumIt=sumIt+current;
    end
end
```

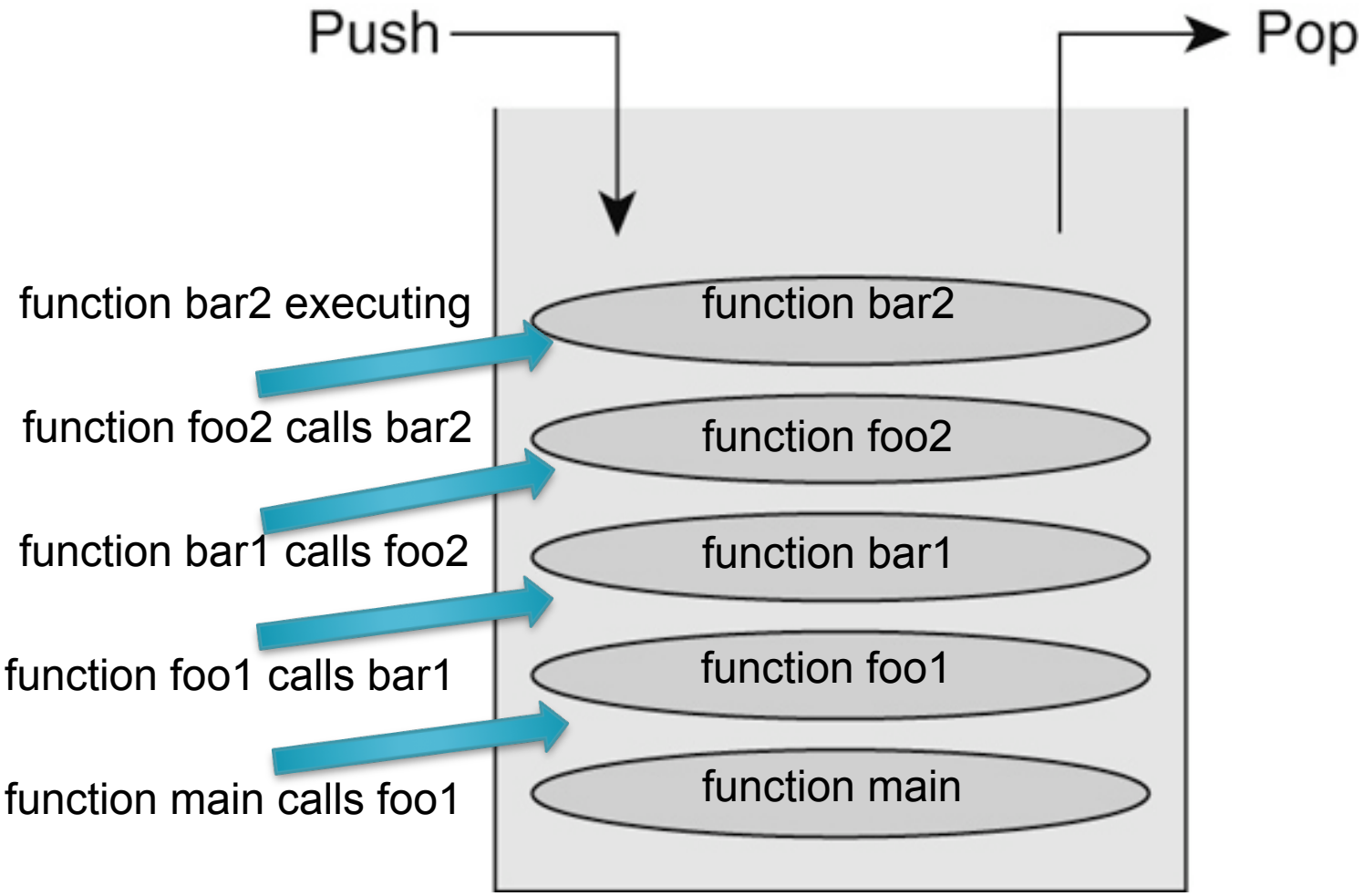
Functions

- Special type of m-file
  - Function name same as file name
- Contains a function name, arguments, output, and "implementation"
- All variables in function are local
  - They are not visible outside call!

Example Function

```
function sumIt=sumOddInt(finish)
   sumIt=0;
   for current=1:finish
       if (mod(current,2)==1)
         sumIt=sumIt+current;
      end
   end
end
% sumIt, current, and finish are local
```

When you call a function…



In recursion these would be same function!!

Another Recursion Example

- Classic Example
  - Function output = numbersSum(input)

```
if (input == 1)
   output = 1;
else
   output = input+numbersSum(input-1)
end
end
```