General Computer Science for Engineers CISC 106 Final Exam Review

Dr. John Cavazos
Computer and Information Sciences
05/18/2009

Lecture Overview

- Loops and Arrays
- Scripts vs functions
- Recursive functions
- C++

Print rows

```
x = [1 2 3; 4 5 6; 7 8 9];
 for k = 1:3
    for n = 1:3
            x(k, n)
    end
 end
```

Assigning values

```
x = [1 2 3; 4 5 6; 7 8 9];
 for k = 1:3
    for n = 1:3
         x(k, n) = k * n;
    end
 end
```

How to use the : operator

- A(:,j) returns the jth column of A
- For example A(:,2)

A =

1	2	3
4	5	6
7	8	9

How to use the : operator

- A(i,:) returns the ith row of A
- For example: A(3,:)

A =

1	2	3
4	5	6
7	8	9

How to use the : operator

- A(c:end)
- gives all of the elements of A from the c^{th} element to the end

```
>> A = [1 2 3; 4 5 6; 7 8 9]
A =
     1     2     3
     4     5     6
     7     8     9
```

A(5:end) → 5 8 3 6 9

Script versus Functions

- Can a script take in a parameter?
- Can a function change a variable in your current environment?
- Can a script return an output?
- Can you call a script by name?

Recursive Functions

- ***Study this!***
- Be able to code a recursive function from a recursive definition
- For example: review (at least) the following:
  - Lecture 9 and 10
  - Lab 4 and Lab 6

Recursive Functions

expt(base,exponent) = base, if exponent is one

base * expt(base,exponent-1) otherwise

Write the code that implements the “expt” function.

Recursive Functions

```
function num = expt(base,exponent)
   if (exponent == 1)
       num = base;
   else
       exponent = exponent - 1;
       num = base * expt(base,exponent);
   end
end
```

For Loops – An Example

Accessing ten elements in an Array

- In MATLAB

```
for i = 1:10
   x = array(i)
end
```

- In C++

```
for (int i = 0; i < 10; i++){
      int x = array[i];
}
```

For Loops – An Example

Accessing every other element in an Array

- In MATLAB

```
for i = 1:2:10
   x = array(i)
end
```

- In C++

```
int x;
for (int i = 0; i < 10; i+=2){
      x = array[i];
}
```

C++

- Declaring and using variables
- Loops
- Function calling
- Function prototypes
- Arrays
  - Declaring
  - Initializing
  - Passing as arguments