CS 271 - August 29, 2017

Topics for today

- Arrays - very quick (ch 6)
- Parameter passing (ch 5)
- Prototypes
- Header files
- gcc -c

To connect to a computer in the CS department.
check out SSH software called Bitvise SSH Client

To find list of host names that you can use with Bitvise or Putty, see the COG Information Site at http://intranet.cs.nmsu.edu

Arrays

To declare an array and allocate space:

```
   int myArray[ 50 ]; // array of 50 elem
```

To use an array, you can refer to only one element at a time.

```
   int i;
   for (i = 0; i < 50; i++)
      printf("%d\n", myArray[ i ]);
```

Passing an array to a function.

```
   float average( int a[ ] , int size ) {
      float sum = 0;
      int count = 0;
      for(count = 0; count < size; count++)
         sum += a[ count ];
      return sum / size;
   }

//  in the main function, if we want to call the average function

int myArray[ ] = { 3, 8, 9, 12, 4 };
               //  ^ initializer list

int avg = average ( myArray, 5 );
```

Parameter passing

All parameters in C are passed by value. Every parameter is a "pass-by-value" parameter.

A copy of the parameter's value is sent to the function.

Average function header:

```
   float average( int a[ ], int size )
```

A call to the average function:

```
   int n = 5;
   float avg = average( myArray, n );
```

The name of an array is the address of the first element. When the function is called, C copies this address and stores the copy of the address in the average function's first parameter.

What do we do if we want to change the parameter?

```
   float average ( int a[ ], int size )
   {
```

C allows us to do that by using pointers.

Pointer = a memory address

```
scanf("%d", &num);
```

& means "the address of"

* means "a pointer to"
or
  means "the value stored at this memory location"

The parameter passing technique of sending an address is called "pass-by-reference".

C does simulated pass-by-reference using pointers.

To repeat: All parameters in C are pass-by-value.

*** assignment: Google the scanf function and find out what its header looks like.

Header file - It's C code but the file name has the extension .h instead of .c
In the header file, we put only the prototypes for the functions (not the main function).

Creating a multifile system

In the next lab, we'll divide our code into 3 files. Here's what goes into each file, using the average function as an example.

File 1: .h contains the prototype for the average function.
File 2: .c contains the definition (implementation) of the average function
File 3: .c that contains the main function

File 1: average.h

```
#ifndef AVERAGE_H
#define AVERAGE_H
#include <stdio.h>

// prototype for the average function
float average( int a[ ], int size );

#endif
```

The three highlighted lines are the "preprocessor wrapper".

File 2: average.c

```
#include <stdio.h>
#include "average.h"
// implementation of the average function
float average (int a[ ], int size) {
   float sum = 0;
   int i;
   for (i = 0; i < size; i++)
      sum += a[ i ];
   return sum / size;
}
```

File 3: test.c

```
#include <stdio.h>
#include "average.h"
int main (void) {
   int x[] = {3, 5, 9, 13, 2};
   float avg = average( x, 5 );
   printf("The average is %f.", avg);
}
```

Compiling the files and creating an executable

```
gcc -c average.c
gcc -c test.c
gcc -o myexecutable average.o test.o
```

Running the executable

```
./myexecutable
```