

Functions: Part 2 of 3

CMSC 104, Fall 2012
John Y. Park



Functions, Part 2 of 3



Topics

- Functions That Return a Value
- Parameter Passing
- Local Variables
- Header Files

Reading

- Sections 5.1 - 5.7

Functions Can Return Values



```
.....  
** averageTwo - calculates and returns the average of two numbers  
** Inputs: num1 - an integer value  
**         num2 - an integer value  
** Outputs: the floating point average of num1 and num2  
.....  
float AverageTwo (int num1, int num2)  
{  
    float average ; /* average of the two numbers */  
  
    average = (num1 + num2) / 2.0 ;  
    return average ;  
}
```

Using averageTwo



```
#include <stdio.h>

float AverageTwo (int num1, int num2) ;

int main ()
{
    float ave ;
    int value1 = 5, value2 = 8 ;

    ave = AverageTwo (value1, value2) ;
    printf ("The average of %d and %d is %f\n", value1, value2, ave) ;
    return 0 ;
}

float AverageTwo (int num1, int num2)
{
    float average ;

    average = (num1 + num2) / 2.0 ;
    return average ;
}
```

Parameter Passing



- **Actual parameters** are the parameters that appear in the function call.
 average = AverageTwo (**value1**, **value2**) ;
- **Formal parameters** are the parameters that appear in the function header.
 float AverageTwo (int **num1**, int **num2**)
- Actual and formal parameters are matched by position. Each formal parameter receives the value of its corresponding actual parameter.

Parameter Passing (con't)



- Corresponding actual and formal parameters do not have to have the same name, but they may.
- Corresponding actual and formal parameters must be of the same data type, with some exceptions.

Local Variables



- Functions only “see” (have access to) their own **local variables**. This includes `main()`.
- Formal parameters are declarations of local variables. The values passed are assigned to those variables.
- Other local variables can be declared within the function body.

Parameter Passing and Local Variables



```
#include <stdio.h>
float AverageTwo (int num1, int num2);
int main ( )
{
    float ave ;
    int value1 = 5, value2 = 8 ;

    ave = AverageTwo (value1,
                      value2) ;
    printf ("The average of ");
    printf ("%d and %d is %fn",
            value1, value2, ave) ;
    return 0 ;
}
```

value1	value2	ave	num1	num2	average
5	8				
int	int	float	int	int	float

```
float AverageTwo (int num1, int num2)
{
    float average ;

    average = (num1 + num2) / 2.0 ;
    return average ;
}
```

Same Name, Still Different Memory Locations



```
#include <stdio.h>
float AverageTwo (int num1, int num2);
int main ( )
{
    float average ;
    int num1 = 5, num2 = 8 ;

    average = AverageTwo (num1,
                          num2) ;
    printf ("The average of ");
    printf ("%d and %d is %fn",
            num1, num2, average) ;
    return 0 ;
}
```

num1	num2	average	num1	num2	average
5	8				
int	int	float	int	int	float

```
float AverageTwo (int num1, int num2)
{
    float average ;

    average = (num1 + num2) / 2.0 ;
    return average ;
}
```

Changes to Local Variables Do NOT Change Other Variables with the Same Name



```
#include <stdio.h>
void AddOne (int number);

int main ()
{
    int num1 = 5;
    AddOne (num1);
    printf ("In main: ");
    printf ("num1 = %d\n", num1);
    return 0;
}
```

num1
5
int

```
void AddOne (int num1)
{
    num1++;
    printf ("In AddOne: ");
    printf ("num1 = %d\n", num1);
}
```

num1

int

OUTPUT

In AddOne: num1 = 6
In main: num1 = 5

Header Files



- Header files contain function prototypes for all of the functions found in the specified library.
- They also contain definitions of constants and data types used in that library.

Commonly Used Header Files



Header File	Contains Function Prototypes for:
<stdio.h>	standard input/output library functions and information used by them
<math.h>	math library functions
<stdlib.h>	conversion of numbers to text, text to numbers, memory allocation, random numbers, and other utility functions
<time.h>	manipulating the time and date
<ctype.h>	functions that test characters for certain properties and that can convert case
<string.h>	functions that manipulate character strings
others	see Chapter 5 of text

Using Header Files



```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
int main ( )
{
    float side1, side2, hypotenuse ;
    printf("Enter the lengths of the right triangle sides: ");
    scanf("%f%f", &side1, &side2);
    if ( (side1 <= 0) || (side2 <= 0) {
        exit (1);
    }
    hypotenuse = sqrt ( (side1 * side1) + (side2 * side2) );
    printf("The hypotenuse = %f\n", hypotenuse);
    return 0 ;
}
```
