

# CS150 Intro to CS I

Fall 2014

# Chapter 6

## Functions

---

- Reading: pp. 299-323
- Good Problems to Work: pp. 321 [6.5, 6.6, 6.7, 6.8, 6.10]

# Function

---

- “A collection of statements that perform a specific task”
  - Functions can be accessed at any point in the code through a *function call*
  - Functions can optionally *return* a value
  - Built-in functions already exist

```
#include <cmath>
cout << pow (2.0, 3.0); // 2.0 raised to 3.0
```

# Function

---

- Functions
  1. are a way of building *modules* in your program
  2. encapsulate some calculation
  3. result in less repetitive code
  4. have a singular theme

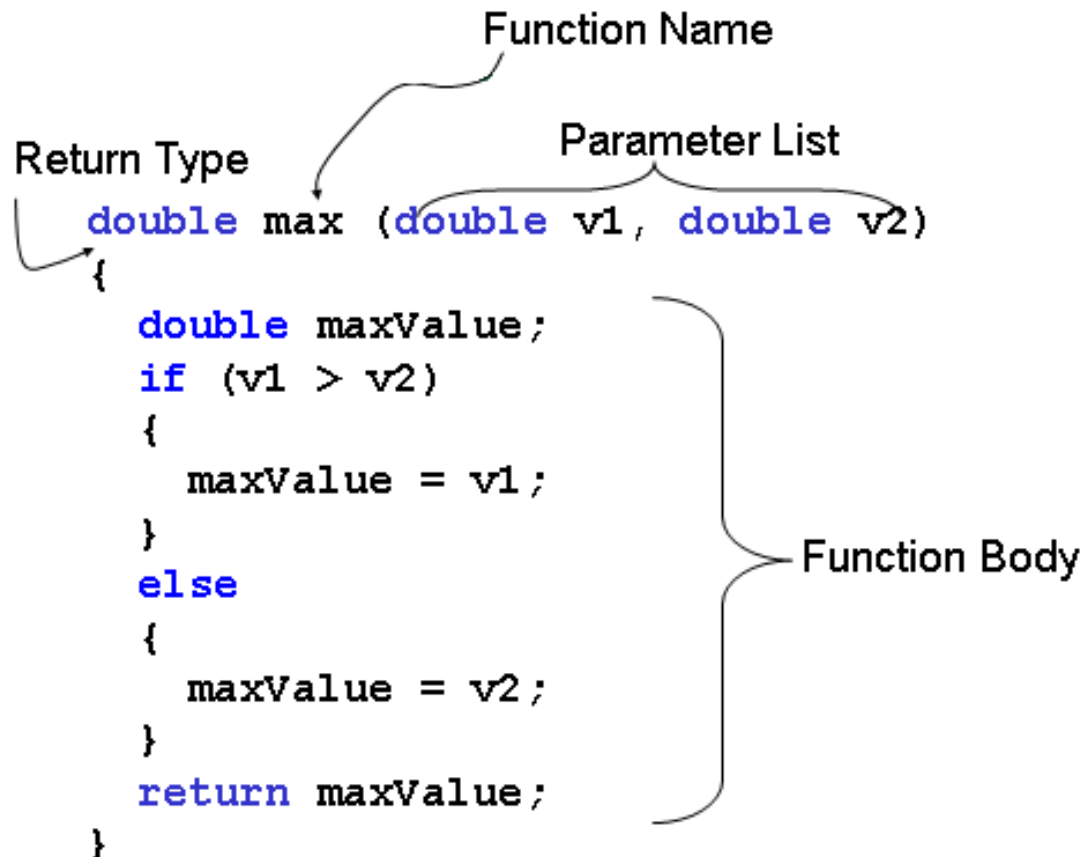
# Writing Functions

---

- Suppose we want to write a function **max** that returns the maximum value of two double values.
- What would a ***call*** to the function look like?

# Max Function Definition

---



# Function Calls

---

```
int main()
{
    double value1, value2, x = 1.5, y = 1.51;

    // must match data types & parameters
    value1 = max(4.2, 2.4);
    value2 = max(x, y);

    cout << value1 << " " << value2
         << max (-1.0, -2.0);

    return EXIT_SUCCESS;
}
```

# Compiling Functions

## Method 1 (preferred method)

---

```
// Function prototype (or function declaration)
double max (double v1, double v2);
```

```
int main ()
{
    cout << max (4.2, 2.4);
    return EXIT_SUCCESS;
}
```

```
// Function definition (slide 6 has complete definition)
double max (double v1, double v2)
{
    . . .
    return maxValue;
}
```



# Compiling Functions

## Method 2

---

```
// Function definition
double max (double v1, double v2)
{
    . . .
    return maxValue;
}

int main ()
{
    cout << max (4.2, 2.4);
    return EXIT_SUCCESS;
}
```

# Functions

---

- You are to use method 1 for your programming assignment solutions
- A function is a group of statements intended to perform a specific task (not specific tasks)
- The function is accessed through a function call
- A function can optionally return a value

# Practice

---

- Write a function **factorial** ( $N! = N * (N-1) * \dots * 2 * 1$ ) to calculate the factorial of a given integer.
  
- Write some C++ statement to use the function to print 4!

# void Functions

---

- Not all functions need to produce a value

```
void printDayOfWeek (int day)
{
    if (SUNDAY == day)
    {
        cout << " Sunday ";
    }
    else if (MONDAY == day)
    {
        cout << " Monday ";
    }
    . . .
    return; // no return value!
}
```

# Practice

---

- Write a function that will calculate the average of three integers and print the result to the screen.
  - What parameters do you need?
  - What should the return type be?
- Write some C++ statements to call this function to determine the average of three integers given by the user.

# Your Favorite!!!!

## Commenting a function definition

---

```
/******  
Function:    maximum  
  
Description: finds the maximum value of two values  
  
Parameters:  value1 - first value of the pair  
             value2 - second value of the pair  
  
Returned:    the maximum value  
*****/  
  
double maximum (double value1, double value2)
```

# Practice

---

- Write a function **charFlip** that flips the case of a letter. When an upper case letter is given, return the lower case version. When a lower case letter is given, return the upper case version.
- If a punctuation or numeric character is given, just return that character.
  - What parameters do you need?
  - What should the return type be?

# Passing Arguments

---

- What is a function argument?
- What is a function parameter?
- A copy of the argument is made in the parameter
- If a parameter is changed in the function, is that reflected in main?



# What will happen?

- What are the arguments? parameters?

```
void swap (int value1, int value2)
{
    int tmp = value1;

    value1 = value2;
    value2 = tmp;
    cout << value1 << " " << value2 << endl;
    return;
}

int main()
{
    int x = 9, y = 10;
    swap(x, y);
    cout << x << " --- " << y << endl;
    return EXIT_SUCCESS;
}
```