

CSI 117 Introductory Object-Oriented Program Analysis and Design

## Unit 4 Functions

#### Text Reference:

- · Gaddis, Chapter 6
- Appendix C (P596 Defining a Function)

## **Objectives**

- Review modules
- Design functions
- Use functions
- Compare similarities and differences between module and function
- Use predefined library functions
- Understand what the black box is

# •

#### **Review Modules**

- A module is a group of statements that exists within a program for the purpose of performing a specific task
- The code for a module is known as a module definition

Module showMessage()
Display "Hello world."
End Module

 To execute the module, write a pseudocode statement that calls it in the main() or other module

Call showMessage()



#### Review Modules (cont.)

- Definition of a module contains two parts:
  - A header
  - The starting point of the module
  - Contains the keyword Module, the name of the module, and ( )
  - A body
    - Comprises one or more statements that are executed when the module is called
- Every executable program MUST contain exactly one module named main(). Program execution always begins with main ()
- A call must be made to the module in order for the statements in the body to execute

1



- In a modularized program, modules use arguments and parameters to communicate with others (share data)
  - An argument on the calling statement and is a value that is sent to a called (invoked) module
  - A parameter is listed in the header of the called module and is a variable that receives the value of an argument passed to it when the module is invoked (called)
  - · Each parameter in the module header must include:
    - a data type and a
  - parameter name

## Review Modules (cont.)

- A parameter list contains multiple parameter declarations separated by commas. Multiple arguments can be passed sequentially into a parameter list.
- Each time a module is called, its parameters are reinitialized to use the new argument values that are passed in when the call takes place
- Arguments passed to a module must match the parameters in the module's header in three ways: number, data type, and order

#### Review Modules (cont.)

- There are two ways to pass arguments to a called module:
  - Pass by Value means that only a copy of the argument's value is passed into the module
  - One-way communication: Calling module can only communicate with the called module
  - Pass by Reference means that the argument is passed into a reference variable
  - Two-way communication: Calling module can communicate with called module; and called module can modify the value of the argument

Gaddis Chapter 3 Part 1 Lecture - Week

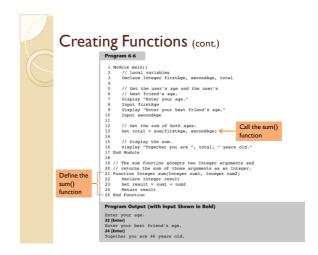


#### **Functions**

- A function is a special type of module. It is like a regular module in the following ways:
  - $^{\circ}$  A group of statements that perform a specific task
  - When you want to execute a function, you call it
- However a function returns a value back to the part of the program that called it
  - The value returned by a function can be used (just like any other value) by the module that calls the function
    - · Can be assigned to a variable
    - · Can be displayed on the screen
    - Can be used in a mathematical expression (if it is a number)

## **Creating Functions**

- The function header specifies
  - The data type of the value that is returned
  - Just like a module header, the function header also specifies the name of the function, and any parameter variables
- The function body comprises one or more statements that are executed when the function is called
  - The Return statement specifies the value that is returned when the function ends
- New keywords:
  - Function
  - Return
  - End Function



## Creating Functions (cont.)

- While you can pass many arguments into a function, you can only return ONE value
- For analysis purposes, an IPO (input, processing, and output) chart, can be used to design a function
  - The input column describes the data that is passed to the function as arguments
  - The processing column describes the process that function performs
  - The output column describes the data that is returned from a function

IPO Chart for the getNewSalary() Function		
Input	Processing	Output (value returned)
	Calculates the new salary for the user	The user's new salary, as a Real

## Creating Functions (cont.)

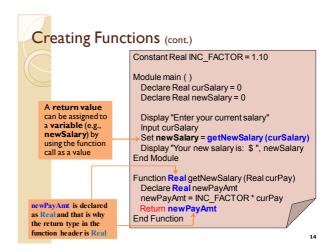
- A function returns exactly one value to the calling module when it returns control to that module. This is done using the Return keyword in pseudocode
- The return type is placed in the header before the immediately after the keyword Function. For example:

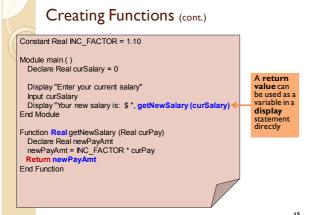
Function Real getNewSalary ( )

The return type is placed immediately after the keyword Function and immediately before the function name.

## Creating Functions (cont.)

- Let's suppose we want to develop a program a main() module and a function named getNewSalary()
  - The main() module input Sthe current salary, call the getNewSalary () function to calculate and return the new salary, and then print the new salary
  - The getNewSalary() function must calculate and return the new salary, which is 10% over the current salary.
     Be sure to use a constant for the amount of the raise





#### Difference between Module and Function

- Both module and function are sets of statements that perform specific tasks
- If a value is needed to be generated in a specific task, and used by other module or function
  - Using module, a pass-by-reference parameter is needed
  - Using function, there is no need to have a pass-byreference parameter, because a function returns a value back to the module or function that called it
    - · As you can see, pass-by-reference technique is more complicated
  - In Java, a variable can not be passed as argument by reference
- Example: Comparison between Module and Function

4



## Library Functions

- Many languages provide libraries of functions which are written by programmers and built into a programming language
- These pre-defined functions usually perform common tasks and save time for the programmer because it allows for code reuse
- Because the internal working of library functions can not been seen, they are also called **black boxes**



## Library Functions (cont.)

- To use a library function
  - You MUST know how to call it in the specific programming language you are using – need to know the function name, type of value returned, and parameter list
  - You DON'T need to know the code written in the function body

18



#### Library Functions (cont.)

- Numbers and math operations:
- random():accepts two arguments which specify in the range of the 1st through the 2nd integer number, and returns a random number:

Display "Rolling the dice..."

Display "The first value is ", random (1, 6)

Display "The second value is ", random (1,6)

In the code above, it displays two random numbers in the range of 1 to 6.

 sqrt(): accepts one argument and returns the square root of the argument.

Set result = sqrt(16)

The result will store the value returned (or generated) by the function sqrt(), which is 4.

For more math functions, see P250



#### Library Functions (cont.)

- Data type conversion functions
  - toInteger():accepts a real number as its argument returns that number converted to an integer. It does not round the fractional part.
  - For more information, see P25 I
- Formatting functions (P254)
- String functions (P254)

20