Introduction to Programming

Functions

Functions

- **The Function of Functions**
- **Functions informally**
- Functions and Parameters: The Executing Details
- Functions That Return Values
- Functions that Modify Parameters
- Functions and Program Structure

Functions

- Functions are used to break down a program into smaller modules.
  - We want each function to be responsible for one task.
  - Rule of thumb
    - Function must be less than 22 lines!
- Example

Functions

- IPO Model
  - Input
  - Processing
  - Output

The Function of Functions

- So far, we've seen four different types of functions:
  - Our programs comprise a single function called main().
  - Built-in Python functions (print, abs)
  - Functions from the standard libraries (math.sqrt)
  - Functions from the graphics module (p.getX(), p.draw())

The Function of Functions

- Our program (main function) uses other functions to complete a task

The Function of Functions

- Example:

The Function of Functions

- Having similar or identical code in more than one place has some drawbacks.
  - Issue one: writing the same code twice or more.
  - Issue two: This same code must be maintained in two separate places.
- Functions can be used to reduce code duplication and make programs more easily understood and maintained

Functions, Informally

- A function is like a subprogram (sub contractor), a small program inside of a program
- The basic idea – we write a sequence of statements and then give that sequence a name. We can then execute this sequence at any time by referring to the name.
  - Just like we can execute our main function by using the name main()
- Example

Functions, Informally

- The part of the program that creates a function is called a function definition.
- When the function is used in a program, we say the definition is called or invoked.
- Creating functions saved us a lot of typing!
- We can customize our function using parameters
  - We can build house using different title