Last Lecture

- Functions
  - Built-in
  - User-defined
- Create our own functions
  - Design function interface
  - Syntax
Learning outcomes

At the end of this course, a student is expected to:

• Create (design) small to medium size programs using Python:
  • Decompose a Python program into functions
• Use the core features of Python to design programs to solve problems: variables, expressions, terminal input and output, type conversion, conditionals, iteration, functions, standard library modules
• Design programs requiring approximately 100 lines and 6 functions (of well-designed code)
• Describe the benefits of using functions
• Construct functions such that:
  • Functions have a single purpose (decomposition)
  • Functions are reusable (generalisation)
  • Functions include parameters and local variables
  • Functions return values
• etc...
Today’s Menu

• Create our own functions
  • Docstring
  • Location of functions in program
• Guideline
  • Generalization
• Why creating functions
### Review: Different kinds of functions + syntax

<table>
<thead>
<tr>
<th>Function takes arguments (i.e., function has parameters)</th>
<th>Function does not take arguments (i.e., function does not have parameters)</th>
</tr>
</thead>
</table>
| **Function returns a result (i.e., a returned value)**   | def functionName( <parameter(s)> ) :
|                                                          | < 1 or more statements >
|                                                          | return <expression>          |
|                                                          | def functionName( ) :
|                                                          | < 1 or more statements >
|                                                          | return <expression>          |
| **Function does not return a result (i.e., a returned value)** | def functionName( <parameter(s)> ) :
|                                                          | < 1 or more statements >
|                                                          | return                      |
|                                                          | def functionName( ) :
|                                                          | < 1 or more statements >
|                                                          | return                      |
Creating our own functions

Problem Statement
• Let us create a function that returns the largest number out of 3 numbers

Requirements:
• cannot use the max( ) built-in function
• cannot use the word max to name your function

Solution
See LargestOf3.py on our course web site
Docstring - Demo

- Docstring is a comment
- Can be used to describe the purpose of a function (what it does and returns) and its parameters
- Example:

    """This function returns the largest number out of 3 numbers passed as parameters.
    Parameters: number1, number2, number3
    -> contain the 3 numbers
    """

- What else does it do? -> Demo!!!
Where to put functions in our Python program

```
# Header Comment Block

# Function 1
...

# Function n

# Main part of my program
```

Python program

Header comment block + import statement(s)

Function definitions

The main part of my program
Function call \(\rightarrow\) situation 1

Python program
Header comment block
+ import statement(s)

Function definition

The main part of my program
Function call → situation 2

# Header Comment Block

# Function 1

# Function 2

# Main part of my program

Python program

Header comment block
+ import statement(s)

Function definitions

The main part of my program
Function call → situation 3

# Header Comment Block

# Function 1

# Function 2

# Main part of my program

Python program

Header comment block

+ import statement(s)

Function definitions

The main part of my program
Function call -> Would this situation work?

# Header Comment Block

# Function 1

# Main part of my program

# Function 2

Python program

Header comment block + import statement(s)

Function definition

The main part of my program

Function definition
Function call ->
Would this situation work?

# Header Comment Block

# Function 1

# Function 2

# Main part of my program

Python program
Header comment block
+ import statement(s)
Function definitions
The main part of my program
Where to put functions in our Python program – Bad!

Python program

Header comment block + import statement(s)

The main part of my program

Function definitions

# Header Comment Block

# Main part of my program

# Function 1

# Function 2
Generalization guideline

• We always strive to create functions (and programs) that solve as many similar problems as possible
  • Characteristic of good programming style (GPS)
Revisit Problem Statement - 1

• Let us create a function that returns the largest number out of 3 numbers

• Requirements:
  • cannot use the max( ) built-in function
  • cannot use the word max to name your function

• A more generalised version of the above function, would be:
Solution to Revisited Problem Statement - 1

• See `LargestOfX.py` on our course web site
Problem Statement - 2

• Let us create a function that prints $x$ number of a symbol in a row

Sample Runs:
>>> ! ! ! ! !
>>> ***
>>> ***
Solution to Problem Statement - 2

- See `PrintRow.py` on our course web site
Why creating functions?

1. Incremental development:
   - Dividing a long program into functions allows us to implement, test and debug the parts one at a time and then assemble them into a working whole

2. “Logical” decomposition:
   - Creating a new function gives us an opportunity to name a code fragment, which makes our program easier to read and debug

3. No more repeated code:
   - Functions can make a program smaller by eliminating repeated code
   - Repeated code is also very error-prone

4. Easy modification:
   - Later on, if we make a change to our program, we know “where to go” to find the code fragment we need to change

5. Code reuse:
   - Well-designed functions are often useful in many programs. Once we write, test and debug a function, we can reuse it
Summary

• Create our own functions
  • Docstring
  • Location of functions in program
• Guideline
  • Generalization
• Why creating functions
Next Lecture

• Hand tracing functions – Things to consider:
  • Stack frame
  • Execution flow
  • Immutable versus mutable
  • Scope of local variable and parameters