Study Guide: Functions

This study guide provides a quick-reference summary of what you learned in this lesson and serves as a guide for the upcoming practice quiz.

In the Functions segment, you learned how to define and call functions, utilize a function's parameters, and return data from a function. You also learned how to differentiate and convert between different data types utilizing variables. Plus, you learned a few best practices for writing reusable and readable code.

Terms

- **return value** - the value or variable returned as the end result of a function
- **parameter (argument)** - a value passed into a function for use within the function
- **refactoring code** - a process to restructure code without changing functionality

Knowledge

- The purpose of the **def()** keyword is to define a new function.
- Best practices for writing code that is readable and reusable:
  - **Create a reusable function** - Replace duplicate code with one reusable function to make the code easier to read and repurpose.
  - **Refactor code** - Update code so that it is self-documenting and the intent of the code is clear.
  - **Add comments** - Adding comments is part of creating self-documenting code. Using comments allows you to leave notes to yourself and/or other programmers to make the purpose of the code clear.

Coding skills

Skill Group 1

- Use a function that accepts multiple parameters
- Return a result value

```
# This function calculates the number of days in a variable number of
# years, months, and days. These variables are provided by the user and
# are passed to the function through the function’s parameters.
def find_total_days(years, months, days):
# Assign a variable to hold the calculations for the number of days in
# a year (years*365) plus the number of days in a month (months*30) plus
# the number of days provided through the "days" parameter variable.
    my_days = (years*365) + (months*30) + days
# Use the "return" keyword to send the result of the "my_days"
# calculation to the function call.
    return my_days

# Function call with user provided parameter values.
print(find_total_days(2,5,23))
```

Run

Reset

Skill Group 2

- Use a function to return the result of a measurement conversion
- Use arithmetic operators to perform a calculation
- Combine text with a function call within a print() statement
- Convert the return value from a float data type to a string for the print() function
- Call the function and perform a calculation on the return value within a print() statement

```
# This function converts fluid ounces to milliliters and returns the
# result of the conversion.
def convert_volume(fluid_ounce):
# Calculate value of the "ml" variable using the parameter variable
# "fluid_ounce". There are approximately 29.5 milliliters in 1 fluid
# ounce.
    ml = fluid_ounce * 29.5
# Return the result of the calculation.
    return ml

# Call the conversion from within the print() function using 2 fluid
# ounces. Convert the return value from a float to a string.
print("The volume in millimeters is " + str(convert_volume(2)))

# Call the function again and double the 2 fluid ounces from within
# the print function.
print("The volume in millimeters is " + str(convert_volume(2)*2))
# Alternative calculation:
# print("The volume in millimeters is " + str(convert_volume(4))

```

Run

Reset

Python practice information

For additional Python practice, the following links will take you to several popular online interpreters and codepads:

- Welcome to Python
- Online Python Interpreter
- Create a new Repl
- Online Python-3 Compiler (Interpreter)
- Compile Python 3 Online
- Your Python Trinket

Mark as completed

Like **Dislike** **Report an issue**