Functions

Remix CS 2019-20
Review: Variables & Arithmetic

- What value does `num1` have after line 4 is executed? What value does `num2` have after line 5?
- What value does `num3` have after line 6?
- What value does `num1` have after line 7?
- What value does `num3` have after line 8?

```python
1  test1_grade = 85.5
2  test2_grade = 91
3
4  num1 = (test1_grade + test2_grade) / 2
5  num2 = test1_grade + test2_grade // 2
6  num3 = str(average2)
7  num1 *= 2
8  num3 *= 2
9```
What we will cover today

- What are functions
- Why do we use them
- How to write functions
- CodeSkulptor link for today:
What are functions

- Math Perspective: Functions can represent a transformation from input to output
  
  \[ f(x) = 5x \] : transforms \( x \) by multiplying it with 5

  **Input** = 3

  **Output** = \( 3 \times 5 = 15 \)

- Computer science uses functions to transform input data into output data in code too!
Why do we use functions

- Code becomes **reusable**: a function can reuse code instructions for different inputs to get different outputs.

- Code becomes **human-readable**: complex set of instructions within a program can be written as a single named function and **reused** to make your code shorter and easy to understand.
How to write a Python function

- Functions have three components
  - Declaration
  - Function Body
  - Return Statement (optional)
How to write a Python function (pt. 2)

Component 1: Declaration

- Syntax: `def function_name(input1, input2):`
  - `def` signifies start of a Python function
  - `function_name` is the name of the function
  - `(input1, input2, input3):` are comma separated variables that refer to each input

- Note: function does not always need an input
  - `def function_name():`
How to write a Python function (pt. 3)

Component 2: Function Body

● Underneath the function body is an indented set of instructions
● These instructions can use the input variables or create new ones to perform operations (ex. perform arithmetic)
How to write a Python function (pt. 4)

Component 3: Return Statement

- This is optional, but last instruction line can have a return statement
- Syntax: `return output`
- Output data that is returned goes back to the caller to store
Composition of Functions in Python

**Input:**
This is where the inputs of the function are initialized. Notice that the inputs are then referred to by the name that is given (like a variable).

```python
def add_x_y(x, y):
    """
    This function takes as input an integer x and integer y. It returns the sum of these two values.
    """
    total = x + y
    return total
```

**Docstring:**
Every function should have a docstring. Python ignores whatever is inside of the quotation marks. A way to briefly describe the function, and its input and output.

**Return statement:**
This line indicates the output of the function. It will return the output to wherever the function is called.

Include whitespace to let Python know which code is apart of the function body (tab).
Using a function

- Functions can be called using their name, followed by parentheses containing their inputs.
- Inputs can be variables or values.
  - `my_function (a, b)` and `my_function (1, 2)` are the same if `a = 1` and `b = 2`.
- Use a variable to store the function’s output.
  - `result = my_function (a, b)`
Example

```python
def add(a, b):
    sum = a + b
    return sum

result1 = add(1, 2)
print("Result1: ", result1)
x = 1
y = 2
result2 = add(x, y)
print("Result2: ", result2)
```

Output:

```
Result1: 3
Result2: 3
```
Using Python Functions

Call the function using the function’s name

Variable z stores output

```
def add_x_y(x, y):
    sum = x + y
    return sum
```

```
a = 2
b = 7
z = add_x_y(a, b)
print(z)
```
Example 1: Write a function to compute $x^2 - x$ for a given integer $x$
Step 1: Identify the inputs and write the function declaration

What are the inputs?

- An integer x

What should the function declaration look like?
Step 1: Identify the inputs and write the function declaration

What are the inputs?

- An integer x

What should the function declaration look like?

```
def my_function(x):
```
Step 2: Figure out the intermediate steps for the function (use variables to store intermediate values)

What are the intermediate steps?

- Compute $x^2$
- Then compute $x^2 - x$

What does this look like?
Step 2: Figure out the intermediate steps for the function (use variables to store intermediate values)

What are the intermediate steps?

- Compute $x^2$
- Then compute $x^2 - x$

What does this look like?

```python
def my_function(x):
    x2 = x ** 2
    x2_minus_x = x2 - x
```
Step 3: Decide what the function should return

What value should the function return

- The value computed for $x^2 - x$

What does this look like?
Step 3: Decide what the function should return

What value should the function return

- The value computed for $x^2 - x$

What does this look like?

```
Code
1 def compute(x):
2     x2 = x ** 2
3     x2_minus_x = x2 - x
4     return x2_minus_x
5
```
Step 4: Test if your function outputs what you expect

What value should the function return for \(x = 7, x = -3\)

- \(7^2 - 7 = 42\)
- \((-3)^2 - (-3) = 12\)

Does it work?

```python
def compute(x):
    x2 = x ** 2
    x2_minus_x = x2 - x
    return x2_minus_x

val_1 = compute(7)
print("val_1 is ", val_1)

val_2 = compute(-3)
print("val_2 is ", val_2)
```

('val_1 is ', 42)
('val_2 is ', 12)
Example 2: Write a function that for a given temperature in fahrenheit computes the temperature in celsius
Step 1: Identify the inputs and write the function declaration

What are the inputs?

- A float fahrenheit

What should the function declaration look like?
Step 1: Identify the inputs and write the function declaration

What are the inputs?

- A float fahrenheit

What should the function declaration look like?

```
1 def f_to_c(fahrenheit):
```
Step 2: Figure out the intermediate steps for the function (use variables to store intermediate values)

\[ \text{Celsius} = \frac{5}{9} (\text{Fahrenheit} - 32) \]

What are the intermediate steps?

- Compute \((\text{fahrenheit} - 32)\)
- Compute \(\frac{5}{9} (\text{fahrenheit} - 32)\)

What does this look like?
Step 2: Figure out the intermediate steps for the function (use variables to store intermediate values)

What are the intermediate steps?

- Compute \((fahrenheit - 32)\)
- Compute \(\frac{5}{9}(fahrenheit - 32)\)

What does this look like?

```
def f_to_c(fahrenheit):
    adjusted_temp = fahrenheit - 32
    celsius = (5/9) * adjusted_temp
```
Step 3: Decide what the function should return

What value should the function return

● The value computed for celsius

What does this look like?
Step 3: Decide what the function should return

What value should the function return

- The value computed for celsius

What does this look like?

```python
def f_to_c(fahrenheit):
    adjusted_temp = fahrenheit - 32
    celsius = (5/9) * adjusted_temp
    return celsius
```
Step 4: Test if your function outputs what you expect

What value should the function return for

- **temp = 64**
  - 17.778
- **temp = -7**
  - -21.667

Does it work?

```python
def f_to_c(fahrenheit):
    adjusted_temp = fahrenheit - 32
    celsius = (5/9) * adjusted_temp
    return celsius

celsius1 = f_to_c(64)
print("Celsius1: ", celsius1)
celsius2 = f_to_c(-7)
print("Celsius2: ", celsius2)
```
More Practice Problems

1. Write a function that computes the average of 3 test grades
2. Write a function that concatenates three strings together
3. Write a function that takes in 2 numbers, \( x \) and \( y \), and computes \( xy - \frac{x^2}{y} + \frac{1}{x+y} \)