# *Activity 1: Introduction*

In this course, you will work in teams of 3--4 students to learn new concepts. This activity will introduce you to the process. We'll also take a first look at variables, assignment, and basic input/output.

**Content Learning Objectives**

*After completing this activity, students should be able to:*

- Describe differences between program and output text.
- Identify and execute Python functions for input/output.
- Write assignment statements and use assigned variables.

**Process Learning Objectives**

*After completing this activity, students should make progress toward:*

- Leveraging prior knowledge and experience of other students. (Teamwork)

## *Model 1 Python Built-In Functions*

You can use built-in Python ***functions*** to perform a specific operation. Sometimes a function will require information (referred to as ***arguments***) to perform its operation. A function will also ***return*** a result after the operation.

To ***call*** (or use) a Python function:

- You must include parentheses after the function's name (e.g., print() prints a blank line).
- If the function takes one or more arguments to perform its operation, you must put that information in the parentheses (e.g., print("Hello, world!") prints a message).

**Do not type anything yet! Read the questions first!**

| Python code | Shell Output |
|---|---|
| `input("enter the mass in grams: ")` | |
| `mass = input("enter another mass in grams: ")` | |
| `mass` | |
| `unit = input("enter the units for mass: ")` | |
| `print(mass, unit)` | |
| `print(mass / 2)` | |
| `ten = 10` | |
| `print(ten / 2)` | |
| `abs(-1)` | |
| `abs(-1 * ten)` | |

## *Questions (15 min)*

start time:

1. List the names of the three functions used in Model 2:

2. What is the argument of the first use of the print function?

3. Type each line of code in a Python Shell, one line at a time, and write the corresponding output (if observed) in the right column of the table. If an error occurs, write what type of error it was (i.e., the first word of the last line of the error message).

   Place an asterisk (*) next to any output for which you were surprised, and note what was unexpected about the output. Don't worry yet about \textit{understanding} any strange output you may see; we will discuss what it all means by the end of class.

4. Which function delayed execution until additional input was entered?

5. Which term, ***user*** or ***programmer***, best defines the role of the person who entered the additional input? Explain.

6. Based on the Shell output, what does the word ***mass*** represent, and how did it get its value?

7. What does the word ***ten*** represent, and how did it get its value?

8. Do the values of ***mass*** and ***ten*** both represent a number? Explain why or why not.

## *Model 2 Variables and Assignment*

In programming, an ***assignment statement*** saves a value to a ***variable***. The variable “is set to'' the value after the ***= operator***. Selecting concise yet descriptive variable names is considered good programming style and will make your programs easier to read.

**Do not type anything yet! Read the questions first!**

| Python code | Shell Output |
|---|---|
| `data = 12` | |
| `data` | |
| `Data` | |
| `Data = 34` | |
| `data` | |
| `Data` | |
| `my data = 56` | |
| `my_data = 78` | |
| `3data = "hello"` | |
| `data3 = "world"` | |
| `hot = 273 + 100` | |
| `273 + 100 = hot` | |
| `hot` | |
| `Hot + 100` | |
| `hot - 100` | |

start
time:

## *Questions (15 min)*

9. Based on the information and Python code in Model 3, give an example representing each of the following:
    a. an assignment statement

    b. the variable being assigned

    c. the assignment operator

    d. the value of the variable immediately after the assignment

10. Similar to \ref{functions.tex}, type each line of code in a Python Shell and write the corresponding output in the space above. If an error occurs, write what type of error. Place an asterisk (*) next to any output for which you were surprised.

11. Circle each *successful* assignment statement in Model 3. How many are there?

12. What is the observed output of a successful assignment statement?

13. After the successful execution of an assignment statement, how can you confirm the value of this variable?

14. For each assignment statement that executed without an error, write the corresponding variable name.

15. Based on the Model 3 output, indicate whether each statement below is true or false.
    a. Variable names in Python can start with a number.

    b. Variable names in Python must start with a lower-case letter.

    c. Variable names in Python may not include spaces.

d. Variable names in Python are case-sensitive.

16. Each of the following assignment statements has an error. Write a valid line of Python code that corrects the assignment statement. Double-check your code using a computer.
    a. 3 + 4 = answer

    b. oh well = 3 + 4

    c. 2x = 7

17. Predict the value of the variable ***hot*** after executing all lines of code in Model 3. Then test your prediction on a computer, and explain the result.

18. Write a line of Python code to assign the current value of ***hot*** to the variable ***cold***. Show output that confirms that you have done this correctly, and explain the code.