



Practical Programming in Python

Inspired by 'Practical Programming' by Paul Gries, Jennifer Campbell, Jason Montojo

Lecture 2: Summary & Exercises Hello, Python

The Python Interpreter, The Shell, Values, Operators, Expressions, Types, The Memory Model

“Programming is learned by writing programs.”
– Brian Kernighan

Lecture 2: Summary

In this lecture you learned the following:

- An operating system is a program that manages your computer's hardware on behalf of other programs. An interpreter or virtual machine is a program that sits on top of the operating system and runs your programs for you. The Python shell is an interpreter, translating your Python statements into language the operating system understands and translating the results back so you can see and use them.
- Programs are made up of statements, or instructions. These can be simple expressions like `3 + 4` and assignment statements like `celsius = 20` (which create new variables or change the values of existing ones). There are many other kinds of statements in Python, and we'll introduce them throughout the course.
- Every value in Python has a specific type, which determines what operations can be applied to it. The two types used to represent numbers are `int` and `float`. Floating-point numbers are approximations to real numbers.
- Python evaluates an expression by applying higher-precedence operators before lower-precedence operators. You can change that order by putting parentheses around subexpressions.
- Python stores every value in computer memory. A memory location containing a value is called an object.
- Variables are created by executing assignment statements. If a variable already exists because of a previous assignment statement, Python will use that one instead of creating a new one.
- Variables contain memory addresses of values. We say that variables refer to values.
- Variables must be assigned values before they can be used in expressions.

Lecture 2: Exercises

When writing code, only use Python concepts that have been introduced in the lectures already.

Exercise 1: Understanding Operators, Types & Expressions

For each of the following expressions, what value will the expression give? Write it down, and *after that* verify your answers by typing the expressions into Python. Write down the types of the operands and the result.

```
1  9 - 3
2  8 * 2.5
3  9 / 2
4  9 / -2
5  9 // 2
6  9 % 2
7  9.0 % 2
8  9.0 % 2.0
9  9 % -2
10 -9 % 2
11 9 / -2.0
12 4 + 3 * 5
13 (4 + 3) * 5
```

Exercise 2: Unary Operators, Signs of Numbers

Unary minus negates a number. Unary plus exists as well; for example, Python understands `+5`. If `x` has the value `-17`, what do you think `+x` should do? Should it leave the sign of the number alone? Should it act like absolute value, removing any negation? Use the Python shell to find out its behavior.

Exercise 3: Understanding Assignments

Write two assignment statements that do the following.

1. Create a new variable, `temp`, and assign it the value 24.
2. Convert the value in `temp` from Celsius to Fahrenheit by multiplying by 1.8 and adding 32; make `temp` refer to the resulting value.

What is `temp`'s new value? Draw a memory model diagram showing what happens.

Exercise 4: Operator Precedence

For each of the following expressions, in which order are the subexpressions evaluated?

```
1 6 * 3 + 7 * 4
2 5 + 3 / 4
3 5 - 2 * 3 ** 4
```

Exercise 5: Understanding Assignments & Expressions Referring to Variables

1. Create a new variable, `x`, and assign it the value `10.5`.
2. Create a new variable, `y`, and assign it the value `4`.
3. Sum `x` and `y` and make `x` refer to the resulting value.

After the last statement has been executed, what are `x` and `y`'s values? Draw a memory model diagram illustrating what happens.

Exercise 6: Understanding Augmented Assignment Statements

Write a bullet list description of what happens when Python evaluates the statement `x += x - x` when `x` has the value `3`. Draw the corresponding memory model diagram.

Exercise 7: Understanding Errors Related to Undefined Variables

When a variable is used before it has been assigned a value, a `NameError` occurs. In the Python shell, write an expression that results in a `NameError`.

Exercise 8: Understanding Syntax

Which of the following expressions results in `SyntaxErrors`? Think about it and then try it out.

```
1 6 * -----8
2 8 = people
3 (((4 ** 3)))
4 (-(-(-(-5)))
5 4 += 7 / 2
```