UMBC

AN HONORS UNIVERSITY IN MARYLAND

CMSC201 Computer Science I for Majors

Lecture 02 – Intro to Python

Last Class We Covered

- Syllabus
  - Grading scheme
  - Academic Integrity Policy
    - (Collaboration Policy)
- Getting Help
  - Office hours
- Programming Mindset
  - “Failure” (isn’t really failure)

Any Questions from Last Time?

Today's Objectives

- To start learning Python
- To learn about variables
  - How to use them
  - Different types
- To learn how to use input and output
  - To do interesting things with our program
- To play a party game

Introduction to Python (Variables)

Python

- Python is a widely used language
  - General purpose
  - High-level language
- Emphasizes code readability
  - More streamlined than some other languages

“Hello World!”

- In Python:

```
print("Hello World!")
```

- In the C++ programming language:

```
#include <iostream>
int main() {
    std::cout << "Hello World!\n";
}
```

Elements of a Program

- Identifiers
  - Variables
  - Functions (later in the semester)
- Expressions
  - Code that manipulates or evaluates identifiers
- Literals
- Operators

We Start Python Today!

- Two ways to use Python

  We will write programs for assignments

  - You can write a program as a series of instructions in a file and then execute it

  Use the interpreter to help you test things

  - You can also test simple Python commands in the Python interpreter

What Is a Variable?

- Something that holds a value
  - Can change (unlimited number of times)
- Similar to variables in math
- In simple terms, a variable is a "box" that you can put stuff in

Rules for Naming Variables

- Variable names can contain:
  - Uppercase letters (**`A-Z`**)
  - Lowercase letters (**`a-z`**)
  - Numbers (**`0-9`**)
  - Underscores (**`_`**)
- Variables can't contain:
  - Special characters like **`$`**, **`#`**, **`&`**, **`^`**, **`)`**, **`(`**, **`@`**

More Rules for Naming Variables

- Variables can be any length
  - `x`
  - `IsKanyeRunningForPresidentIn2020`
  - `myName`

- Variables cannot **start** with a digit
  - `2cool4school` is not a valid variable
  - `cool4school` is a valid variable

Variables and Keywords

- Keywords are “reserved” words in Python

```
False      class      finally    is         return
None       continue   for        lambda     try
True       def        from       nonlocal   while
and        del        global     not        with
as         elif       if         or         yield
assert     else       import     pass
break      except     in         raise
```

- Variables cannot be keywords
  - **`or`** is <u>not</u> a valid variable name
  - **`orange`** is an acceptable variable name

Exercise: Variables

- Are the following legal or illegal in Python?

```
1spam

raise1

Spam_and_Eggs

EXIT_CODE
```

Exercise: Variables

- Are the following legal or illegal in Python?

`1spam`	**No – Illegal!**
`raise1`	**Yes – legal!**
`Spam_and_Eggs`	**Yes – legal!**
`EXIT_CODE`	**Yes – legal!**

Exercise: Variables

- Are the following legal or illegal in Python?

`Spam_and_Eggs`

Yes – legal!

But it doesn't follow our coding standards!

`spamAndEggs` **or** `spam_and_eggs`

Using Variables in Python

- You create a variable when you declare it
- You also need to initialize it before using
  - Use the assignment operator (equal sign)

assignment operator

```
richFiddy   = 50000000
poorFiddy   = 0.50
brokeFiddy = 0
```

Introduction to Python (Expressions)

Expressions

- Programs manipulate data
  - Allows us to do interesting things
- Expressions calculate new data values
- Use assignment operator to set new value

Expressions Example

variable being set

assignment operator

value (a "literal")

```
numCandy    = 4
priceCandy = 0.58
totalCost   = numCandy * priceCandy
```

expression

Common Mistake

- Many new programmers mix up the left and right hand sides of the assignment operator
  - Variable being set must be on the ***left***
  - Expression is on the ***right***
  - Evaluate the expression first, then assign the value

```
numCandy = 4 + 1 ✓

4 + 1 = numCandy ✗
```

Variable Types

- There are many different kinds of variables!
  - Numbers
    - Whole numbers (Integers)
    - Decimals (Floats)
  - Booleans (**`True`** and **`False`**)
  - Strings (collections of characters)

Variables Types: Examples

```
aString    = "Hello class"
float_1    = 1.12
myBool     = True
anInteger = 7

dogName    = "Ms. Wuffington"
classCode = 201
```

Variable Usage

- Variables are designed for storing information
- Any piece of information your program uses or records <u>must</u> be stored in a variable
  - Python doesn’t have a “short term memory,” so everything needs to be written down for it

Introduction to Python (Literals and Operators)

Literals

- Literals in Python are values you use “literally”
  - Can be assigned to a variable or not
- For example:
  - 2 is an integer literal
  - “Hello” is a string literal
  - 4.0 is a float literal
  - False is a Boolean literal

Using Literals

- The expression below assigns the string literal “CMSC” to a variable called major

```
major = "CMSC"
```

- The expression below prints the integer literal 50 without assigning it to a variable

```
print(50)
```

Operators

- Operators are special symbols that allow Python to perform different operations

- There are many types of operators
  - Mathematical
  - Comparison
  - Assignment
  - Logical

Operator Types

- We won’t cover all the types in detail, but here are some simple examples
- Mathematical

```
+     -     *     /     %
```

- Comparison

```
<     <=    !=    >=    ==
```

- Assignment

```
=     +=    *=
```

we’ll cover the “weird” ones later

Practice Exercises

- Print the value of the variable myDog, but assign a value to myDog first
- Set a value for a variable called bill, and calculate and print the 15% tip for that bill
- Set a value for a variable called numClasses and a variable called name, and print out that NAME has NUMCLASSES classes

Introduction to Python (Input and Output)

Output

- Output is text that is printed to the screen
  - So the user can see it (and respond)

- The command for this is **`print`**
  - Use the keyword “**`print`**” and put what you want to be displayed in parentheses after it

Output Example

```
print (3 + 4)
print (3, 4, 3 + 4)
print()
print("The answer is", 3 + 4)
```

What does this output to the screen?

```
7
3 4 7

The answer is 7
```

Output Exercise 1

- What will the following code snippet print?

```
a = 10
b = a * 5
c = "Your result is:"
print(c, b)
```

```
Your result is: 50
```

Output Exercise 2

- What will the following code snippet print?

```
a = 10
b = a
a = 3
print(b)
```

There are two possible options for what this could do! Any guesses?

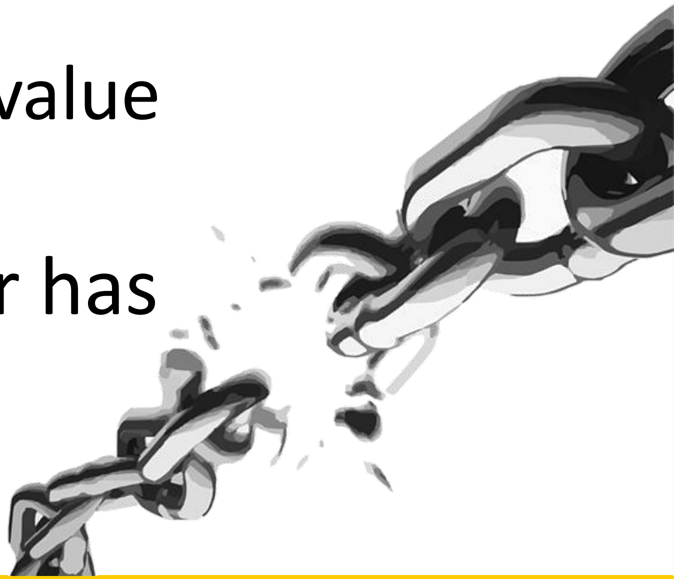
```
10
```

Output Exercise 2 Explanation

- Why does it print out 10?

- When you set one variable equal to another, they <u>don't</u> become linked!
  - They are separate <u>copies</u> of a value

- After `b` is set to 10, it no longer has anything else to do with `a`

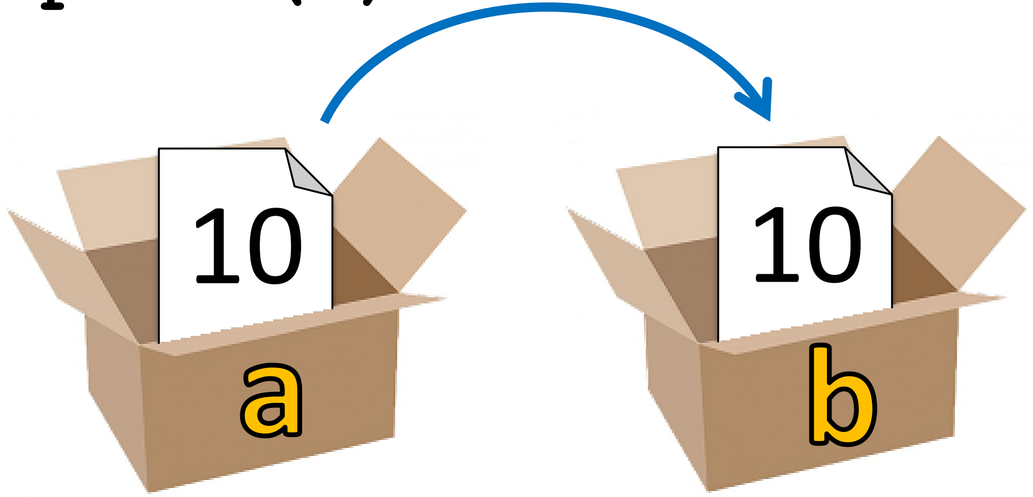


Output Exercise 2 Explanation

```
a = 10
b = a
a = 3
print(b)
```

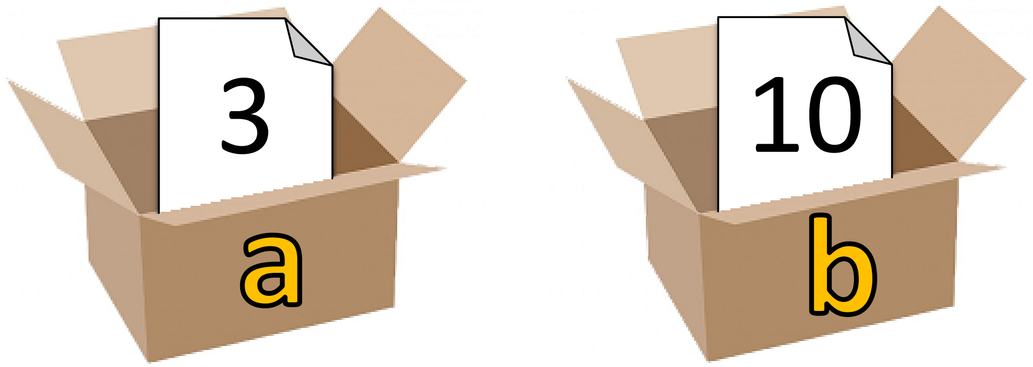
Output Exercise 2 Explanation

```
a = 10
b = a
a = 3
print(b)
```



Output Exercise 2 Explanation

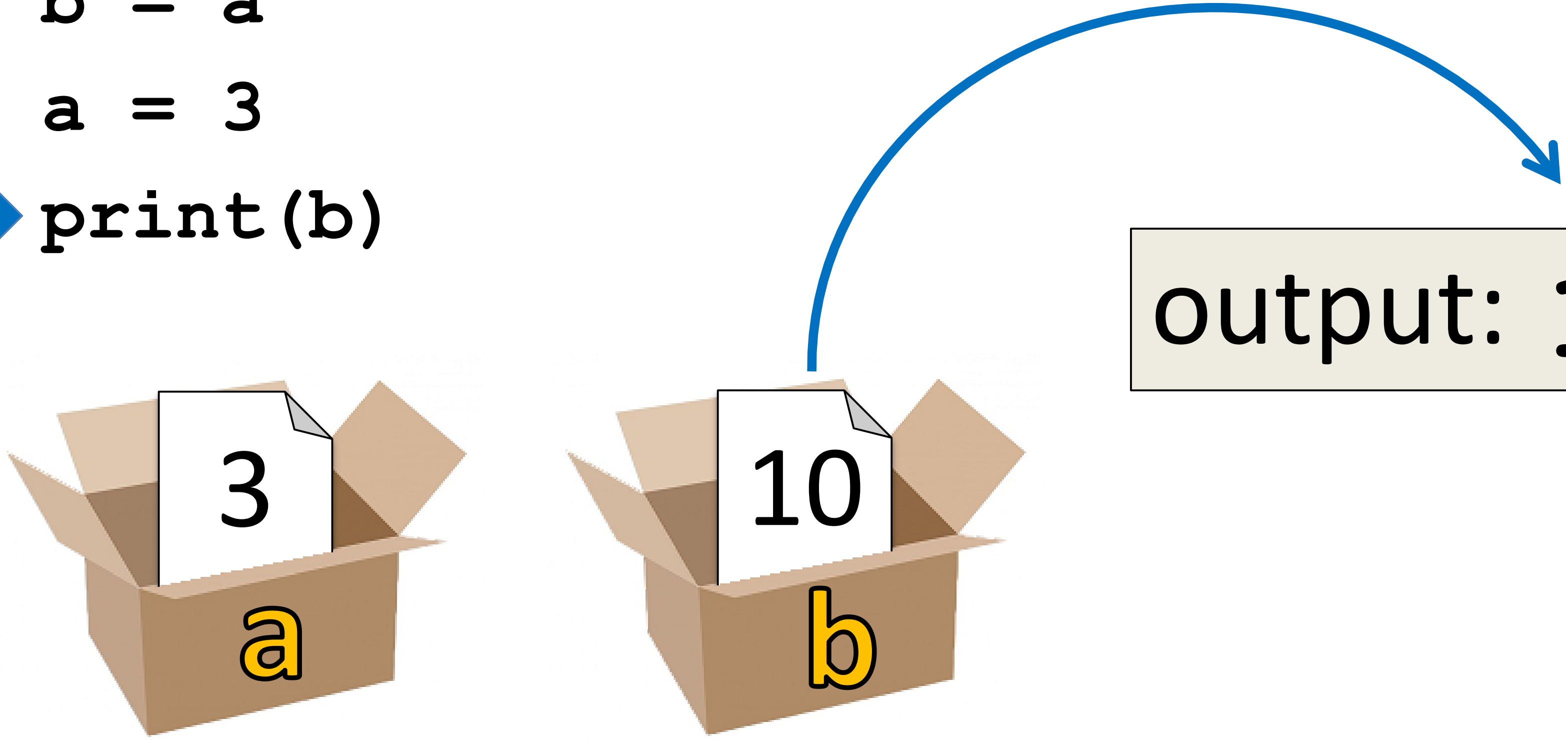
```
  a = 10
  b = a
➡ a = 3
  print(b)
```



Output Exercise 2 Explanation

```
a = 10
b = a
a = 3
print(b)
```

output: 10



Input

- Input is text we get from the user
  - We must tell them what we want first

```
userNum = input("Please enter a number: ")
print(userNum)
```

- The output and input will look like this:

```
Please enter a number: 22
22
```

How Input Works

```
userNum = input("Please enter a number: ")
```

- Takes the text the user entered and stores it
  - In the variable named **userNum**

- You can do this as many times as you like!

```
userNum  = input("Enter another number: ")
userNum2 = input("Enter a new number: ")
userAge  = input("Please enter your age: ")
```

Input as a String

- Everything that comes through **`input`**`()` will come in the form of a string
- There is a difference between `"10"` and `10`
  - `"10"` is a string containing two characters
  - `10` is understood by Python as a number

Converting from String

- To turn an input string into a number, you can do the following:

```
aNum = input("Enter a number: ")
aNum = int(aNum)
```

- `int` stands for "integer" (a whole number)

- You can also do it in one line:

```
aNum = int(input("Enter a number: "))
```

Converting from String

- Do you think the string `"1,024"` will work if we try to cast it as an integer? Why?
- It won't work
  - The comma character isn't a number
- We can cast to other data types as well

```
flt = float(input("Enter float: "))
```

Class Exercise: Mad Libs

- Mad Libs is a word game where one player prompts the others for different types of words, using them to fill the blank in a story.

- The result is often hilarious, and almost always nonsensical.

Announcements

- Your discussions (Labs) start next week!
  - Go to your scheduled location and time
- HW 0 will be out on the course website soon
  - Due by ***Wednesday*** (Feb 8th) at 8:59:59 PM
- HW 1 will be out (on Blackboard) Saturday
  - You must first complete the Syllabus/Course Website Quiz to see it (also released by Saturday)