Python Variables

Fall 2019

Variables

- Variable - a name that represents a value stored in the computer’s memory
- Assignment statement - used to assign a value to a variable (=)
    - variable = value
- Variables can be reassigned -- they aren’t set in stone when assigned once

Variable Naming

- Variables must have a name to be used
- Your names must always start with a letter or underscore, but they can *only* contain these types characters
    - Letters (capital and lowercase)
    - Numbers
    - Underscores
- Valid variable names:
    - firstName, LastName, first_name, last_name, name1, name2, _name1
- Invalid variable names:
    - 1name, 2name, first name, firstName!

Variable Naming

- Python contains some reserved keywords valuable to the functionality of Python
  - Your variable name cannot be a Python keyword
    - It can *contain* a keyword (valid: className), but it cannot be *solely* the keyword (invalid: class)
- Have descriptive variable names (give a hint what it will be used for)

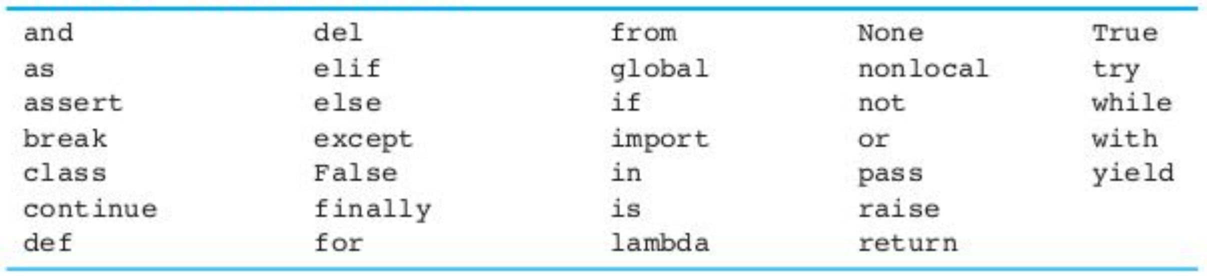


Table 1-2 The Python key words

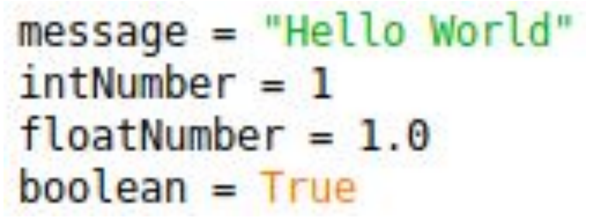
`and`	`del`	`from`	`None`	`True`
`as`	`elif`	`global`	`nonlocal`	`try`
`assert`	`else`	`if`	`not`	`while`
`break`	`except`	`import`	`or`	`with`
`class`	`False`	`in`	`pass`	`yield`
`continue`	`finally`	`is`	`raise`	
`def`	`for`	`lambda`	`return`	

Data Types

- Programming languages have different types they can store data in
- Some of the most basic types are:
    - Integers (int)
        - Whole numbers (1, 2, 3, 4… etc)
    - Floating point (float)
        - Decimals (1.0, 1.2, 1.5… etc)
    - Strings
        - Words, sentences, can contain numbers and special characters also
    - Booleans
        - True/False values

Declaring Variables

- Python doesn't have explicit data typing
  - You don't have to tell Python what type you're storing in the variable
  - It determines this based on how the data is entered into the variable
- Python is case sensitive
  - If you declare a variable "temp" then "Temp", "tEmp", etc. will not work
- Examples:
  - temp = 1 (integer)
  - temp = 1.0 (floating point)
  - temp = "Hello World" (string)
  - temp = True (boolean)



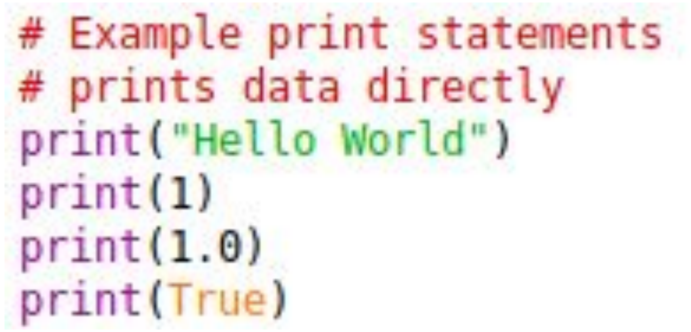
```
message = "Hello World"
intNumber = 1
floatNumber = 1.0
boolean = True
```

Declaring Variables: Notes

- Integers
  - Declared as a number with no quotation marks and *without* a decimal
- Floating points
  - Declared as a number with no quotation marks and *with* a decimal
- Strings
  - Declared as characters encompassed by quotation marks
  - Can be numbers, letters, or special symbols (“2” would be a string, not an integer)
- Booleans
  - Declared as True or False outside of quotation marks (this is *case sensitive* meaning true/false does not read as a boolean and will throw an error)

Printing: Data Types

- Different data types also introduces printing different data types
  - print("Hello World") will print a string
    - As stated previously, print(Hello World) would not work; this is because Hello World is a string and needs quotations to print
  - print(1) will print an integer
    - This does not need quotations to print as it reads the 1 as an integer
  - print(1.0) will print a floating point
  - print(True) will print a boolean



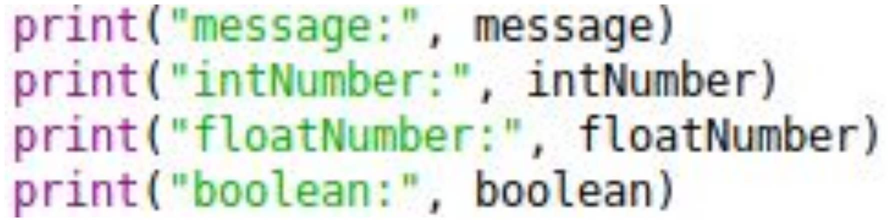
```
# Example print statements
# prints data directly
print("Hello World")
print(1)
print(1.0)
print(True)
```

Printing: Variables

- You can print variables directly to your screen
  - To print a variable you have to assign a value to it
    - Ex. temp = 1
  - Then, you insert the name into a print statement
    - print(temp)
  - If you add quotations it will print what is in the quotations directly
    - print(“temp”) will print “temp”, not 1
  - If you don’t maintain the exact name (down to casing), it will throw an error
    - print(Temp) will cause your program to crash

Printing: Variables

- Variables can be print in a compound statement
  - They can be print with more than one variable per print statement
  - They can also be print with an accompanying string (or two, three, etc.)
- Example:
  - message = “Hello World”
  - print(“Message:”, message) will print “Message: Hello World”
- Example:
  - number = 1
  - print(“My number is:”, number) will print “My number is: 1”

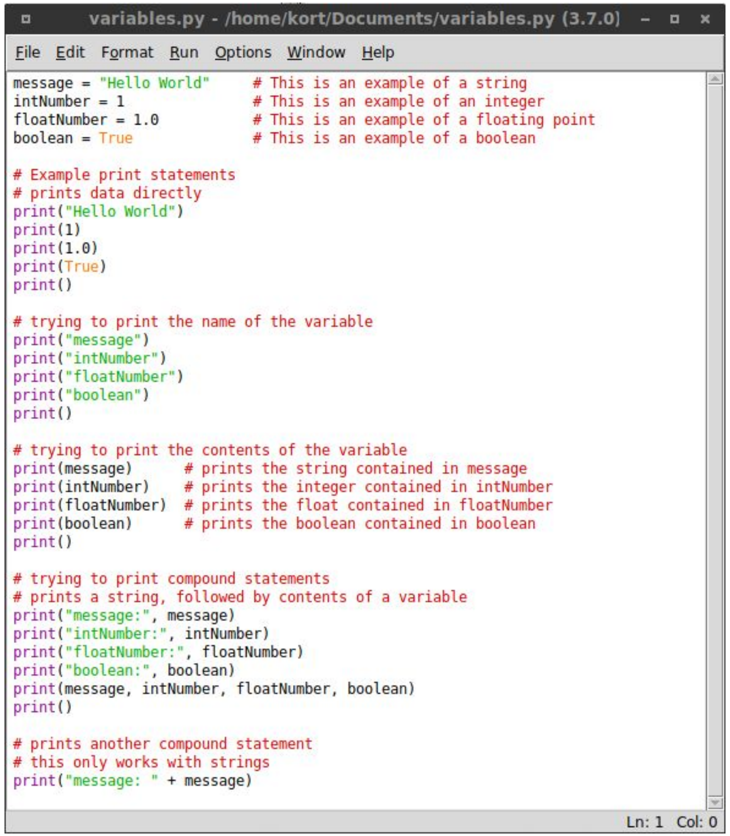


```
print("message:", message)
print("intNumber:", intNumber)
print("floatNumber:", floatNumber)
print("boolean:", boolean)
```

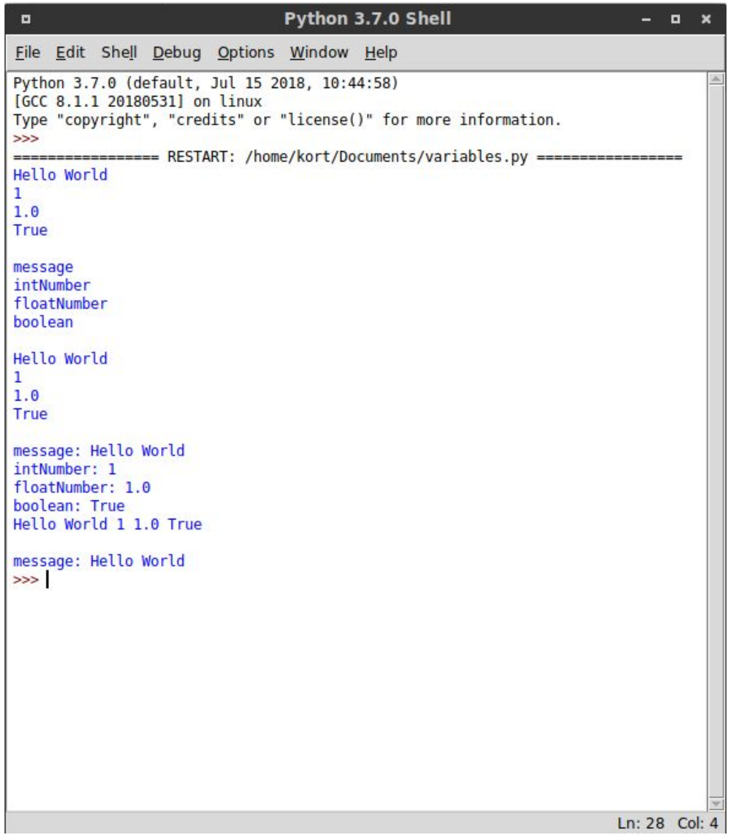
Printing: Notes

- The other way to print a compound statement “+” works only with string data types
    - Example
        - temp = “Kortni”
        - print(“Hello, my name is “ + temp) will work
        - print(“Hello, my name is “ + “Kortni”) will work
    - Example
        - temp = 1
        - print(“My number is “ + temp) will not work
        - print(“My number is “ + 1) will not work
- This will become explained when arithmetic is introduced

Example Code


variables.py - /home/kort/Documents/variables.py (3.7.0)
File Edit Format Run Options Window Help
Ln: 1 Col: 0

```
message = "Hello World"     # This is an example of a string
intNumber = 1               # This is an example of an integer
floatNumber = 1.0           # This is an example of a floating point
boolean = True              # This is an example of a boolean

# Example print statements
# prints data directly
print("Hello World")
print(1)
print(1.0)
print(True)
print()

# trying to print the name of the variable
print("message")
print("intNumber")
print("floatNumber")
print("boolean")
print()

# trying to print the contents of the variable
print(message)      # prints the string contained in message
print(intNumber)    # prints the integer contained in intNumber
print(floatNumber)  # prints the float contained in floatNumber
print(boolean)      # prints the boolean contained in boolean
print()

# trying to print compound statements
# prints a string, followed by contents of a variable
print("message:", message)
print("intNumber:", intNumber)
print("floatNumber:", floatNumber)
print("boolean:", boolean)
print(message, intNumber, floatNumber, boolean)
print()

# prints another compound statement
# this only works with strings
print("message: " + message)
```



```
Python 3.7.0 (default, Jul 15 2018, 10:44:58)
[GCC 8.1.1 20180531] on linux
Type "copyright", "credits" or "license()" for more information.
>>>
================= RESTART: /home/kort/Documents/variables.py =================
Hello World
1
1.0
True

message
intNumber
floatNumber
boolean

Hello World
1
1.0
True

message: Hello World
intNumber: 1
floatNumber: 1.0
boolean: True
Hello World 1 1.0 True

message: Hello World
>>> 
```