CMSC201
Computer Science I for Majors

Lecture 03 – Variables

Prof. Jeremy Dixon

Based on slides by Shawn Lupoli and Max Morawski at UMBC
Last Class We Covered

• Algorithms
• Program Development
• Control Structures
  – Sequential
  – Decision Making
  – Loops
• Types of Errors
  – Syntax
  – Logic
Any Questions from Last Time?
Today’s Objectives

• To start learning Python
• To learn more 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
“Cowboy Coding”

• Jumping right in to writing code

• Disadvantages
  – No formal management of project
  – No standard way of coding
  – Not planning things out
    • Forgetting to include important things
    • Having to make big changes later
Software Development Process

1. Analyze the problem
   – Determine specifications (requirements)
2. Create a design
3. Implement the design
4. Test and debug the program
5. Maintain the program
Example: Temperature Converter

You have been invited to live in Europe during a semester abroad. You aren’t sure how to dress because the temperature is given in Celsius.

• Problem:
  – Temperature is given in Celsius

• Solution:
  – Write a program to convert Celsius to Fahrenheit
Input/Process/Output

• Input
  – What information do you need for your converter?

• Process
  – What formulas do you need for your converter?

• Output
  – What is the output from your converter?
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:
  
  ```python
  print("Hello World!")
  ```

• In the C++ programming language:
  
  ```cpp
  #include <iostream>
  int main() {
      std::cout << "Hello, world!\n";
  }
  ```
Elements of a Program

• Identifiers
  – Variables
  – Modules (later in the semester)
  – Functions (later in the semester)

• Expressions
  – Code that manipulates or evaluates identifiers
We Start Python Today!

• Two ways to use python
  – You can write a program as a series of instructions in a file and then execute it
  – You can also test simple Python commands in the Python interpreter.
Rules for Naming Variables

• Variables can contain:
  – Uppercase letters (A–Z)
  – Lowercase letters (a–z)
  – Numbers (0–9)
  – Underscores (_)

• Variables can’t contain:
  – Special characters ($, #, &, ^, ), (, @)
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 the reserved words in Python

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>class</td>
<td>finally</td>
<td>is</td>
<td>return</td>
<td></td>
</tr>
<tr>
<td>None</td>
<td>continue</td>
<td>for</td>
<td>lambda</td>
<td>try</td>
<td></td>
</tr>
<tr>
<td>True</td>
<td>def</td>
<td>from</td>
<td>nonlocal</td>
<td>while</td>
<td></td>
</tr>
<tr>
<td>and</td>
<td>del</td>
<td>global</td>
<td>not</td>
<td>with</td>
<td></td>
</tr>
<tr>
<td>as</td>
<td>elif</td>
<td>if</td>
<td>or</td>
<td>yield</td>
<td></td>
</tr>
<tr>
<td>assert</td>
<td>else</td>
<td>import</td>
<td>pass</td>
<td>raise</td>
<td></td>
</tr>
<tr>
<td>break</td>
<td>except</td>
<td>in</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

• Variables cannot be keywords
  – **or** is **not** a valid variable name
  – **orange** is an acceptable variable name
What Is a Variable?

• Something that holds a value
  – Can change (multiple times)

• Similar to variables in math

• In simple terms, a variable is a “box” that you can put stuff in
Exercise: Variables

• Are the following legal or illegal in Python?

  1spam   No – Illegal!
  raise1  Yes – legal!
  Spam_And_Eggs Yes – legal!
Using Variables in Python

• Create a variable by declaring it
• Also need to initialize it
  – Use the 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

numCandy = 10
priceCandy = 0.50
totalCandy = numCandy * priceCandy
Common Mistake

• Many new programmers mix up the left and right hand sides of the assignment operator

• Variable being set is on the left

• Expression is on the right

\[
\text{numCandy} = 10 \quad \checkmark \\
10 = \text{numCandy} \quad \times
\]
Variable Types

• There are many different kinds of variables!
  – Numbers
    • Integers
    • Decimals
  – Booleans (True and False)
  – Strings (collections of characters)
Variables Types: Examples

aString = "Hello class"
decimal_1 = 1.12
myBool = True
wholeNum = 7
dogName = "Mrs. Wuffington"
classCode = 201
Variable Usage

• Variables are designed for storing information

• Any piece of information your program uses or records must be stored in a variable
Introduction to Python
(Input and Output)
Output

• Output is text printed to the screen
  – So the user can see it and respond

• One command for this is `print`
Output Example

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

7
3 4 7

The answer is 7
Output Exercise 1

• What will the following code snippet print?
  ```python
  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?

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

10

There are two possible options for what this could do! Any guesses?
Output Exercise 2 Explanation

• Why does it print out 10?

• When you set one variable equal to another, they don’t become linked!

• After \( b \) is set to 10, it no longer has anything else to do with \( a \)
Input

- Input is text we get from the user

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

- The output will look like this:
  
  Please enter a number: 10
  10
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 two character long string
  – 10 is understood by Python as a number
Converting from String

- To turn an input string into a number, you can do the following:
  
  ```python
  aNum = input("Enter a number: ")
  aNum = int(aNum)
  ```

- `int` stands for integer (a whole number)
Class Exercise: Mad Libs

• Mad Libs is a phrasal template word game where one player prompts others for a list of words to substitute for blanks in a story, before reading the – often comical or nonsensical – story aloud.

• The game is frequently played as a party game or as a pastime
Exercise: Calculating Averages

• Write, on paper or on your computer, a program that asks the user for two numbers and prints out the average.

• Does the order of operations come into play for this exercise?
Exercise: Assignment Weighting

• Pretend you’re writing a program to compute someone’s weight grade. You have so far:

\[
\begin{align*}
\text{hwWeight} & = 0.4 \\
\text{examWeight} & = 0.5 \\
\text{discussionWeight} & = 0.1
\end{align*}
\]

• Write a program that then asks the user for their homework grade, exam grade, and discussion grade and prints out their total grade in the class.
Announcements

• Your Lab 1 is an online lab this week!
  – Due by this Thursday (Sept 3rd) at 8:59:59 PM

• Homework 1 is out
  – Due by next Tuesday (Sept 8th) at 8:59:59 PM

• Both of these assignments are on Blackboard
  – Weekly Agendas are also on Blackboard