Lecture 2

Variables & Assignment
Announcements for Today

If Not Done Already

- Enroll in Piazza
- Sign into CMS
  - Fill out the Survey
  - Complete AI Quiz
- Read the textbook
  - Chapter 1 (browse)
  - Chapter 2 (in detail)

Lab 1

- Please stay in your section
  - If you drop, you are stuck
  - E-mail conflicts to Amy
  - ahf42@cornell.edu
  - Will review by next week
- Have one week to complete
  - Fill out questions on handout
  - Show to TA before next lab
  - Show in consulting hours
Helping You Succeed in this Class

• **Consultants.** ACCEL Lab Green Room
  - Daily office hours (see website) with consultants
  - Very useful when working on assignments

• **AEW Workshops.** Additional discussion course
  - Runs parallel to this class – completely optional
  - See website; talk to advisors in Olin 167.

• **Piazza.** Online forum to ask and answer questions
  - Go here first *before* sending question in e-mail

• **Office Hours.** Talk to the professor!
  - Available in Carpenter Hall Atrium between lectures
# Labs vs. Assignments

<table>
<thead>
<tr>
<th>Labs</th>
<th>Assignments</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Held every week</td>
<td>• Every two weeks</td>
</tr>
<tr>
<td>• Graded on <strong>completeness</strong></td>
<td></td>
</tr>
<tr>
<td>▪ Always S/U</td>
<td>▪ First one due Sep. 18</td>
</tr>
<tr>
<td>▪ Try again if not finished</td>
<td>▪ Graded on <strong>correctness</strong></td>
</tr>
<tr>
<td>• Indirect affect on grade</td>
<td></td>
</tr>
<tr>
<td>▪ Can miss up to 2 labs</td>
<td>▪ Assign points out of 100</td>
</tr>
<tr>
<td>▪ After that, grade reduced</td>
<td>▪ But <strong>first</strong> one is for <em>mastery</em></td>
</tr>
<tr>
<td>• Similar to language drills</td>
<td></td>
</tr>
<tr>
<td>▪ Simple, but take time</td>
<td>▪ Resubmit until perfect grade</td>
</tr>
</tbody>
</table>

- 40% of your final grade
- Designed to be more fun
  - Graphics, game design
iClickers

• Have you registered your iclicker?
• If not, visit
  ▪ atcsupport.cit.cornell.edu/pollsrvc/
• Instructions on iClickers can be found here:
  ▪ www.it.cornell.edu/services/polling/howto-students.cfm
• Find these links on the course webpage
  ▪ Click “Texts/iClickers”
  ▪ Look under “iClickers”
Warm-Up: Using Python

• How do you plan to use Python?

A. I want to work mainly in the ACCEL lab
B. I want to use my own Windows computer
C. I want to use my own Macintosh computer
D. I want to use my own Linux computer
E. I will use whatever I can get my hands on
Type: Set of values and the operations on them

- **Type int:**
  - **Values:** integers
  - **Ops:** +, −, *, /, %, **

- **Type float:**
  - **Values:** real numbers
  - **Ops:** +, −, *, /, **

- **Type bool:**
  - **Values:** True and False
  - **Ops:** not, and, or

- **Type str:**
  - **Values:** string literals
    - Double quotes: "abc"
    - Single quotes: 'abc'
  - **Ops:** + (concatenation)

Will see more types in a few weeks
Converting Values Between Types

• Basic form: \textit{type(value)}
  \begin{itemize}
  \item \texttt{float(2)} converts value 2 to type \texttt{float} (value now 2.0)
  \item \texttt{int(2.6)} converts value 2.6 to type \texttt{int} (value now 2)
  \item Explicit conversion is also called “casting”
  \end{itemize}

• Narrow to wide: \texttt{bool \Rightarrow int \Rightarrow float}

  \begin{itemize}
  \item \textit{Widening}. Python does automatically if needed
    \begin{itemize}
    \item \textbf{Example}: 1/2.0 evaluates to 0.5 (casts 1 to \texttt{float})
    \end{itemize}
  \item \textit{Narrowing}. Python never does this automatically
    \begin{itemize}
    \item Narrowing conversions cause information to be lost
    \item \textbf{Example}: \texttt{float(int(2.6))} evaluates to 2.0
    \end{itemize}
  \end{itemize}
Operator Precedence

• What is the difference between the following?
  ▪ 2*(1+3)
  ▪ 2*1 + 3

• Operations are performed in a set order
  ▪ Parentheses make the order explicit
  ▪ What happens when there are no parentheses?

• Operator Precedence: The fixed order Python processes operators in absence of parentheses
Operator Precedence

• What is the difference between the following?
  ▫ 2*(1+3)  \text{add, then multiply}
  ▫ 2*1 + 3  \text{multiply, then add}

• Operations are performed in a set order
  ▫ Parentheses make the order explicit
  ▫ What happens when there are no parentheses?

• \textbf{Operator Precedence}: The \textit{fixed} order Python processes operators in \textit{absence} of parentheses
Precedence of Python Operators

- **Exponentiation**: `**`
- **Unary operators**: `+`  `-`
- **Binary arithmetic**: `*`  `/`  `%`
- **Binary arithmetic**: `+`  `-`
- **Comparisons**: `<`  `>`  `<=`  `>=`
- **Equality relations**: `==`  `!=`
- **Logical not**
- **Logical and**
- **Logical or**

- Precedence goes downwards
  - Parentheses highest
  - Logical ops lowest
- Same line = same precedence
  - Read “ties” left to right
  - Example: `1/2*3` is `(1/2)*3`

- Section 2.7 in your text
- See website for more info
- Was major portion of Lab 1

8/25/16

Variables & Assignments
Expressions vs Statements

Expression

• **Represents** something
  ▪ Python *evaluates it*
  ▪ End result is a value

• **Examples:**
  ▪ 2.3
  ▪ \((3+5)/4\)

Statement

• **Does** something
  ▪ Python *executes it*
  ▪ Need not result in a value

• **Examples:**
  ▪ `print "Hello"`
  ▪ `import sys`

Will see later this is not a clear cut separation
Variables (Section 2.1)

- A variable
  - is a named memory location (box)
  - contains a value (in the box)
  - can be used in expressions

- Examples:

  - Variable \( x \), with value 5 (of type int)
  - Variable \( \text{area} \), w/ value 20.1 (of type float)
• A variable
  - is a **named** memory location (**box**)
  - contains a **value** (in the box)
  - can be used in expressions

• Examples:

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>5</td>
<td>int</td>
</tr>
<tr>
<td>area</td>
<td>20.1</td>
<td>float</td>
</tr>
</tbody>
</table>

Variable names must start with a letter (or _). The type belongs to the value, not to the variable.
Variables (Section 2.1)

• A variable
  ▪ is a named memory location (box)
  ▪ contains a value (in the box)
  ▪ can be used in expressions

• Examples:

  Variable names must start with a letter (or _).

  | x  | 5   | Variable x, with value 5 (of type int) |
  | area | 20.1 | Variable area, w/ value 20.1 (of type float) |

The value in the box is then used in evaluating the expression.

The type belongs to the value, not to the variable.
Variables (Section 2.1)

• A variable
  ▪ is a **named** memory location (**box**)  
  ▪ contains a **value** (in the box)  
  ▪ can be used in expressions

• Examples:

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>5</td>
<td>int</td>
</tr>
<tr>
<td>area</td>
<td>20.1</td>
<td>float</td>
</tr>
</tbody>
</table>

Variable names must start with a letter (or _).

The value in the box is then used in evaluating the expression.

The type belongs to the value, not to the variable.

1e2 is a float, but e2 is a variable name
Variables and Assignment Statements

• Variables are created by assignment statements
  ▪ Create a new variable name and give it a value
    \[ x = 5 \]

• This is a statement, not an expression
  ▪ Tells the computer to DO something (not give a value)
  ▪ Typing it into `>>>` gets no response (but it is working)

• Assignment statements can have expressions in them
  ▪ These expressions can even have variables in them
    \[ x = x + 2 \]

Two steps to execute an assignment:
1. evaluate the expression on the right
2. store the result in the variable on the left
Variables and Assignment Statements

- Variables are created by **assignment statements**
  - Create a new variable name and give it a value
    
    \[ x = 5 \]
    
    **the value**
    
    **the variable**

- This is a **statement**, not an **expression**
  - Tells the computer to DO something (not give a value)
  - Typing it into `>>>` gets no response (but it is working)

- Assignment statements can have expressions in them
  - These expressions can even have variables in them
    
    \[ x = x + 2 \]

Two steps to execute an assignment:
1. evaluate the expression on the right
2. store the result in the variable on the left
Variables and Assignment Statements

- Variables are created by **assignment statements**
  - Create a new variable name and give it a value
    - \( x = 5 \)
  - This is a **statement**, not an **expression**
    - Tells the computer to DO something (not give a value)
    - Typing it into `>>>` gets no response (but it is working)

- Assignment statements can have expressions in them
  - These expressions can even have variables in them
    - \( x = x + 2 \)

Two steps to execute an assignment:
1. evaluate the expression on the right
2. store the result in the variable on the left
Variables and Assignment Statements

- Variables are created by **assignment statements**
  - Create a new variable name and give it a value
    
    \[
    x = 5
    \]

- This is a **statement**, not an **expression**
  - Tells the computer to DO something (not give a value)
  - Typing it into `>>>` gets no response (but it is working)

- Assignment statements can have expressions in them
  - These expressions can even have variables in them
    
    \[
    x = x + 2
    \]

Two steps to execute an assignment:
1. evaluate the expression on the right
2. store the result in the variable on the left
Variables and Assignment Statements

- Variables are created by assignment statements
  - Create a new variable name and give it a value
    \[ x = 5 \]
  - This is a statement, not an expression
    - Tells the computer to DO something (not give a value)
    - Typing it into `>>=` gets no response (but it is working)
- Assignment statements can have expressions in them
  - These expressions can even have variables in them
    \[ x = x + 2 \]

Two steps to execute an assignment:
1. evaluate the expression on the right
2. store the result in the variable on the left
Variables and Assignment Statements

- Variables are created by **assignment statements**
  - Create a new variable name and give it a value
    - \( x = 5 \)
    - This is a **statement**, not an **expression**
      - Tells the computer to DO something (not give a value)
      - Typing it into >>> gets no response (but it is working)
    - Assignment statements can have expressions in them
      - These expressions can even have variables in them
        - \( x = x + 2 \)
  - Two steps to execute an assignment:
    1. evaluate the expression on the right
    2. store the result in the variable on the left
Execute the Statement: $x = x + 2$

- Draw variable $x$ on piece of paper:

  $x$ 5
Execute the Statement: $x = x + 2$

• Draw variable $x$ on piece of paper:

  \[
  x \quad 5
  \]

• Step 1: evaluate the expression $x + 2$
  - For $x$, use the value in variable $x$
  - Write the expression somewhere on your paper
Execute the Statement: \( x = x + 2 \)

- Draw variable \( x \) on piece of paper:
  
  \[
  x \quad 5
  \]

- Step 1: evaluate the expression \( x + 2 \)
  
  - For \( x \), use the value in variable \( x \)
  - Write the expression somewhere on your paper

- Step 2: Store the value of the expression in \( x \)
  
  - Cross off the old value in the box
  - Write the new value in the box for \( x \)
Execute the Statement: \( x = x + 2 \)

- Draw variable \( x \) on piece of paper:
  
  \[
  x \quad 5
  \]

- Step 1: evaluate the expression \( x + 2 \)
  - For \( x \), use the value in variable \( x \)
  - Write the expression somewhere on your paper

- Step 2: Store the value of the expression in \( x \)
  - Cross off the old value in the box
  - Write the new value in the box for \( x \)

- Check to see whether you did the same thing as your neighbor, discuss it if you did something different.
Which One is Closest to Your Answer?

A:  
\[
\begin{array}{c}
\times \red{7} \\
\times 7 \\
\end{array}
\]

B:  
\[
\begin{array}{c}
\times \red{5} \\
\times 7 \\
\end{array}
\]

C:  
\[
\begin{array}{c}
\times \red{7} \\
\times 7 \\
\end{array}
\]

D:  
\[
\\_\left(\text{T^u}_n\right)\_\/\ /\]

8/25/16  Variables & Assignments  27
Which One is Closest to Your Answer?

A:

\[
\begin{align*}
x & \times 7 \\
x & \times 5 \\
x & \times 7
\end{align*}
\]

B:

\[
\begin{align*}
x & \times 5 \\
x & \times 7
\end{align*}
\]

C:

\[
\begin{align*}
x & \times 5 \\
x & \times 7
\end{align*}
\]

\[
x = x + 2
\]
Execute the Statement: \( x = 3.0 \times x + 1.0 \)

- You have this:

  \[
  \begin{array}{c}
  x \quad \boxed{7} \\
  \end{array}
  \]
Execute the Statement: \( x = 3.0 \times x + 1.0 \)

• You have this:

\[
\begin{array}{c}
\text{x} \\
7
\end{array}
\]

• Execute this command:

- Step 1: **Evaluate** the expression \( 3.0 \times x + 1.0 \)
- Step 2: **Store** its value in \( x \)
Execute the Statement: $x = 3.0 \times x + 1.0$

- You have this:
  
  \[
  x = 7
  \]

- Execute this command:
  
  - Step 1: **Evaluate** the expression $3.0 \times x + 1.0$
  - Step 2: **Store** its value in $x$

- Check to see whether you did the same thing as your neighbor, discuss it if you did something different.
Which One is Closest to Your Answer?

A:  
\[ x \times 22.0 \]

B:  
\[ x \times 7 \times 22.0 \]

C:  
\[ x \times 22.0 \]

D:  
\[ \_\_(ツ)_\_/\_ \]
Which One is Closest to Your Answer?

A: 
\[ x \times 22.0 \]  
✓

B: 
\[ x \times 7 \]
\[ x \times 22.0 \]

C: 
\[ x \times 22.0 \]

\[ x = 3.0 \times x + 1.0 \]
Execute the Statement: \( x = 3.0 \times x + 1.0 \)

- You now have this:
  \[
  x \quad 22.0
  \]

- The command:
  - Step 1: **Evaluate** the expression \( 3.0 \times x + 1.0 \)
  - Step 2: **Store** its value in \( x \)

- This is how you execute an assignment statement
  - Performing it is called **executing the command**
  - Command requires both **evaluate** AND **store** to be correct
  - Important **mental model** for understanding Python
Exercise: Understanding Assignment

• Add another variable, interestRate, to get this:

  x  22.0   interestRate  4

• Execute this assignment:

  \[
  \text{interestRate} = \frac{x}{\text{interestRate}}
  \]

• Check to see whether you did the same thing as your neighbor, discuss it if you did something different.
Which One is Closest to Your Answer?

A:

\[
\text{interestRate} = 5.5
\]

\[
x \times 22.0 \times 5.5
\]

B:

\[
\text{interestRate} = 5.5
\]

\[
x \times 22.0
\]

C:

\[
\text{interestRate} = 5.5
\]

\[
x \times 22.0
\]

D:

\[
\text{interestRate} \times 5
\]

\[
x \times 22.0
\]
Which One is Closest to Your Answer?

A:

\[ x \times 22.0 \times 5.5 \]

interestRate

B:

\[ x \times 22.0 \]

interestRate

C:

\[ x \times 22.0 \]

interestRate

interestRate

E:

\[ \_{(ツ)_/\_} \]
interestRate = x/interestRate

B:

x 22.0

interestRate 5.5

D:

x 22.0

interestRate 5

Which One is Closest to Your Answer?
Exercise: Understanding Assignment

• You now have this:
  \[
  x \times \times 22.0 \quad \text{interestRate} \times 5.5
  \]

• Execute this assignment:
  \[
  \text{interestRate} = x + \text{interestRate}
  \]

• Check to see whether you did the same thing as your neighbor, discuss it if you did something different.
Which One is Closest to Your Answer?

A:

\[
\begin{array}{c}
\times x \times 22.0 \\
\text{interestRate} \times 5.5 27.5
\end{array}
\]

B:

\[
\begin{array}{c}
\times x \times 22.0 \\
\text{interestRate} \times 5.5 \\
\text{intrestRate} 27.5
\end{array}
\]

C:

\[
\begin{array}{c}
\times x \times 22.0 27.5 \\
\text{interestRate} \times 5.5
\end{array}
\]

D:

\[
\begin{array}{c}
\times x \times 22.0 \\
\text{interestRate} \times 5.5 \\
\text{intrestRate} 27.5
\end{array}
\]

8/25/16 Variables & Assignments 40
Which One is Closest to Your Answer?

A:

\[ x \times 22.0 \]
interestRate \[ \times 5.5 \]

B:

\[ x \times 22.0 \]

intrestRate \[ 27.5 \]

C:

\[ x \times 22.0 \]
interestRate \[ 5.5 \]

E: \( (ツ)_/\)
Which One is Closest to Your Answer?

A:

\[ x \times 22.0 \]
\[ \text{interestRate} \times 5.5 \times 27.5 \]

B:

\[ x \times 22.0 \]
\[ \text{interestRate} \times 5.5 \]
\[ \text{intrestRate} = x + \text{interestRate} \]

\[ \wedge \theta \]
Which One is Closest to Your Answer?

<table>
<thead>
<tr>
<th>A:</th>
<th>B:</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>22.0</td>
<td>22.0</td>
</tr>
<tr>
<td>interestRate</td>
<td>interestRate</td>
</tr>
<tr>
<td>5.5</td>
<td>5.5</td>
</tr>
<tr>
<td>27.5</td>
<td>27.5</td>
</tr>
</tbody>
</table>

\[ \text{intrestRate} = x + \text{interestRate} \]

Spelling mistakes in Python are bad!!
Dynamic Typing

• Python is a **dynamically typed language**
  - Variables can hold values of any type
  - Variables can hold different types at different times
  - Use `type(x)` to find out the type of the value in `x`
  - Use names of types for conversion, comparison

• The following is acceptable in Python:
  ```
  >>> x = 1
  >>> x = x / 2.0
  >>> x
  0.5
  ```

• Alternative is a **statically typed language** (e.g. Java)
  - Each variable restricted to values of just one type
Dynamic Typing

• Python is a **dynamically typed language**
  ▪ Variables can hold values of any type
  ▪ Variables can hold different types at different times
  ▪ Use `type(x)` to find out the type of the value in `x`
  ▪ Use names of types for conversion, comparison

• The following is acceptable in Python:

```
>>> x = 1
x contains an **int** value
>>> x = x / 2.0
x now contains a **float** value
```

• Alternative is a **statically typed language** (e.g. Java)
  ▪ Each variable restricted to values of just one type
Dynamic Typing

• Often want to track the type in a variable
  ▪ What is the result of evaluating x / y?
  ▪ Depends on whether x, y are int or float values

• Use expression type(<expression>) to get type
  ▪ type(2) evaluates to <type 'int'>
  ▪ type(x) evaluates to type of contents of x

• Can use in a boolean expression to test type
  ▪ type('abc') == str evaluates to True