Variables
## Expressions vs Statements

<table>
<thead>
<tr>
<th>Expression</th>
<th>Statement</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Represents</strong> something</td>
<td><strong>Does</strong> something</td>
</tr>
<tr>
<td>- Python <em>evaluates it</em></td>
<td>- Python <em>executes it</em></td>
</tr>
<tr>
<td>- End result is a value</td>
<td>- Need not result in a value</td>
</tr>
<tr>
<td><strong>Examples:</strong></td>
<td><strong>Examples:</strong></td>
</tr>
<tr>
<td>- 2.3</td>
<td>- <code>print('Hello')</code></td>
</tr>
<tr>
<td>- ((3+5)/4)</td>
<td>- <code>import sys</code></td>
</tr>
</tbody>
</table>

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

• A variable is
  ▪ a named memory location (box),
  ▪ a value (in the box)

• Examples

  \[
  \begin{align*}
  x & \quad 5 \\
  \text{area} & \quad 20.1
  \end{align*}
  \]

  Variable \textbf{x}, with value 5 (of type \textbf{int})

  Variable \textbf{area}, w/ value 20.1 (of type \textbf{float})

• Variable names must start with a letter
  ▪ So \textbf{1e2} is a \textbf{float}, but \textbf{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
    
    \[
    \text{x = 3} \quad \text{the value}
    \]
  
  \[
  \text{x} \quad \text{the variable}
  \]

- 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
    
    \[
    \text{x = x + 2} \quad \text{the expression}
    \]
  
  \[
  \text{x} \quad \text{the variable}
  \]
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:

  \[
  \begin{array}{c}
  x \\
  \hline
  5
  \end{array}
  \]

- 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 \( x \)
Which One is Closest to Your Answer?

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A:</td>
<td>B:</td>
<td>C:</td>
<td>D:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| 7 | 5 | 7 | (ツ)_/\
Which One is Closest to Your Answer?

A:

\[ x \times 7 \]

B:

\[ x \times 5 \]
\[ x \times 7 \]

C:

\[ x \times \]
\[ x \times 7 \]

\[ x = x + 2 \]
Assignment Statements

- Make new variables:
  \[ \text{interestRate} = 4 \]

- Change existing variables:
  \[ x = 3.0 \times x + 1.0 \]
But Beware of Misspellings

\[
\text{intrestRate} = x + \text{interestRate}
\]
But Beware of Misspellings

\[ \text{intrestRate} = x + \text{interestRate} \]
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:
  ```python
  >>> x = 1
  >>> x = x / 2.0
  >>> x contains an `int` value
  >>> x = float(x)
  >>> x contains a `float` value
  >>> type(x) == int
  >>> type(x) == float
  ```

• 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 \texttt{int} or \texttt{float} values
- Use expression \texttt{type(<expression>)} to get type
  - \texttt{type(2)} evaluates to \texttt{<type 'int'>}
  - \texttt{type(x)} evaluates to type of contents of \(x\)
- Can use in a boolean expression to test type
  - \texttt{type('abc') == str} evaluates to \texttt{True}