CSCI 141
Scott Wehrwein

Variables
The Assignment Operator
Goals

• Know how to name and store values in variables

• Understand the behavior of the assignment operator

• Know how to use a variable to stand in for the value it represents (stores)

• Know the rules for naming variables and the conventions for deciding on good variable names
Variables

- Variables are a basic component of most programming languages.
- They allow you to store (or remember) values.
- Computers are pretty dumb, but they're really good at a few things, for example:
  - arithmetic
  - remembering things
Variables: Definition

- A variable is a name in your program that refers to a piece of data (value).
Variables: Usage

• A **variable** is a name in your program that *refers* to a piece of data (value).

• How do you use them?
  1. Decide what value you want to store in the variable
  2. Decide on a sensible name
  3. In your program, use the **assignment operator** to assign that variable name to the value:

\[ \text{my\_age} = 32 \]

The assignment operator.
For now, think of \texttt{my\_age} as a named place where we can store any value.

You can replace the current value with a different one:

\begin{verbatim}
my\_age = 33
\end{verbatim}
The Assignment Operator: Not “Equals”

my_age = 32

The assignment operator.

• This is not stating an equality, like in math.

• It is associating a name with a value.

   my_age = 32
   my_age = 33
The Assignment Operator: Not “Equals”

A helpful diagram

```
my_age = 3

my_age = 33

my_age = 32

my_age = 33
```

```
int 32

int 33
```
Using Variables

Assigning a value is not stating an equality, like in math: it’s storing a value.

```python
my_age = 32
my_age = 33
```

A variable’s value can be updated (overwritten) by a new value using the assignment operator.

- ✗ “my_age equals 32”
- ✓ “my_age becomes 32”
- ✓ “the variable my_age takes on the value 32”
What can you do with variables?

Use them anywhere you’d use a value!

```
print(5)  a = 5
print(a)
```

These two programs both print 5.
Variable Names

• How do you use variables?
  1. Decide what value you want to store in the variable
  2. Decide on a sensible name
  3. In your program, use the assignment operator to store that value in the variable

• Great power, great responsibility: variables names can be almost anything!
Variable Names

• Great power, great responsibility: variables names can be almost anything!
• **Valid** variable names:
  • start with a letter or an underscore (_)
  • can contain any letters and digits
  • are case-sensitive (name is not the same as Name)
  • are not the same as any Python language **keywords** (words that already mean something else):
    
    False, None, True, and, as, assert, async, await, break, class, continue, def, del, elif, else, except, finally, for, from, global, if, import, in, is, lambda, nonlocal, not, or, pass, raise, return, try, while, with, yield

True  2p  s2  ✓ a_number  ✓ firstOfThreeValues
Variable Names

- Great power, great responsibility: variables names can be almost anything!
- A **good** variable name:
  - is descriptive - tell a reader what data they refer to
  - is not too long
  - follows a standard naming convention, e.g.:
    - starts with lower case letter
    - words are separated by underscores

These depend on context!