Day 03

Variables, Statements, and Functions (oh my)
Instructions for replit.com have been posted to Piazza
Recap

Expressions are part of a program that has a value

Expressions are evaluated to produce their value

Simple expressions (cannot be decomposed):

   4, 12.7, True, “Hello”, etc...

Compound expressions (composed of multiple expressions):

   3 + 7, “hello” + “world”, 4 < 12, etc...
Variables

A variable is a **name** that has been assigned a **value**

Because it has a **value**, a variable is another example of an **expression**

But how do we create a variable?

How do we assign it a value?

How do we use it?
Assignment Statements

A **statement**, unlike an expression, does not have a value

A **statement** has an *effect*, and can be *executed*

The first type of statement we’ll look at is the **assignment statement**

\[ \text{<name> = <expression>} \]
Assignment Statements

A **statement**, unlike an expression, does not have a value

A **statement** has an **effect**, and can be **executed**

The first type of statement we’ll look at is the **assignment statement**

\[
<\text{name}> = <\text{expression}>
\]

**assignment operator**

(\text{NOT EQUALS!})
What’s in a name?

A name (in python) must follow a few rules:

1. Begins with an underscore or a letter
2. Contains letters, underscores, or digits

Examples: rose, _romeo, JuLiEt47, shake_Speare, oneTrueLove
What’s in a name?

A name (in python) must follow a few rules:

1. Begins with an underscore or a letter
2. Contains letters, underscores, or digits

Examples: rose, _romeo, JuLiEt47, shake_Speare, oneTrueLove

(please don’t name your variables like this)
Assignment Statements

A **statement**, unlike an expression, does not have a value.

A **statement** has an **effect**, and can be **executed**.

The first type of statement we’ll look at is the **assignment statement**.

\[
\texttt{<name> = <expression>} \]

Assignment Statements

A statement, unlike an expression, does not have a value.

A statement has an effect, and can be executed.

The first type of statement we’ll look at is the assignment statement.

```
myFavoriteNumber = 12
```
Assignment Statements

A **statement**, unlike an expression, does not have a value.

A **statement** has an **effect**, and can be **executed**.

The first type of statement we’ll look at is the **assignment statement**.

```plaintext
myFavoriteNumber = 12
```

The effect of executing this statement is the value 12 getting assigned to the variable named “myFavoriteNumber”.
More Examples

```python
sum = 10 + 12

color = "Blue"

full_name = "Eric " + "Mikida"

average = sum / 2
```
sum = 10 + 12

color = “Blue”

full_name = “Eric” + “Mikida”

average = sum / 2

Remember: Variables have a value! They can be used as simple expressions!
Demo in Replit
Functions

A **function** is a block of code (multiple statements) that has a **name**

A function’s block of code is executed by **calling** the function

A function is called by using its name and a list of **arguments** (inputs)

*Think of a function like a machine that takes some input, does some work, and produces some output.*
Examples

\begin{align*}
\text{pow}(3, 2) \\
\text{round}(467 / 15, 4) \\
\text{print}("Hello " + "world!")
\end{align*}
Examples

Names

- `pow(3, 2)`
- `round(467 / 15, 4)`
- `print("Hello " + "world!")`
Examples

Names

- pow(3, 2)
- round(467 / 15, 4)
- print("Hello " + "world!")

Arguments (values)
Examples

pow(3, 2) computes $3^2$, which is 9

It takes inputs (the base and exponent), does work (computes the answer), and produces output (in this case, 9)
Examples

pow(3, 2) computes $3^2$, which is 9

It takes inputs (the base and exponent), does work (computes the answer), and produces output (in this case, 9)

Since a function call produces a value...a function call is another example of an expression!
Examples

Python has many built-in functions, here are a few more:

abs(x)
help()
min(x,y)
max(x,y)
pow(x,y)
print(x)
round(x) and round(x,y)
Examples

Python has many built-in functions, here are a few more:

- abs(x)
- help()
- min(x, y)
- max(x, y)
- pow(x, y)
- print(x)
- round(x) and round(x, y)

What if the function we want doesn’t exist…cliffhanger for next lecture :)
Demo in Replit