

# CSE 503

## Introduction to Computer Science for Non-Majors

Dr. Eric Mikida

epmikida@buffalo.edu

208 Capen Hall

**Day 03**

**Variables, Statements, and Functions (oh my)**

# Announcements

Instructions for [replit.com](https://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**

$$\langle name \rangle = \langle expression \rangle$$

# 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**

*<name> = <expression>*

assignment operator  
**(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**

$$\langle name \rangle = \langle expression \rangle$$

# 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**

```
myFavoriteNumber = 12
```

The effect of executing this statement is the value 12 getting assigned to the variable named "myFavoriteNumber"

# More Examples

```
sum = 10 + 12
```

```
color = "Blue"
```

```
full_name = "Eric " + "Mikida"
```

```
average = sum / 2
```

# More Examples

```
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

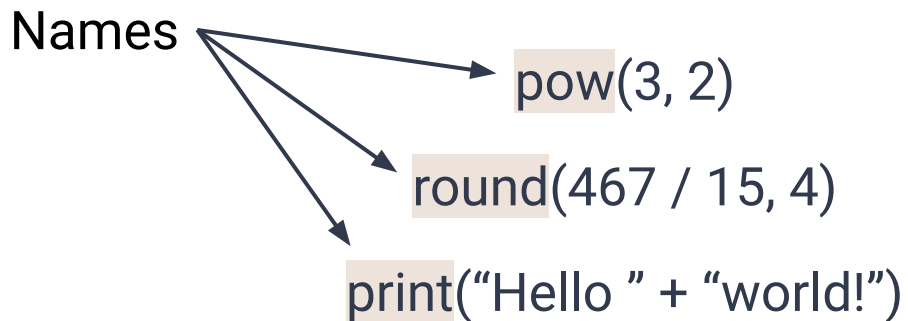
```
pow(3, 2)
```

```
round(467 / 15, 4)
```

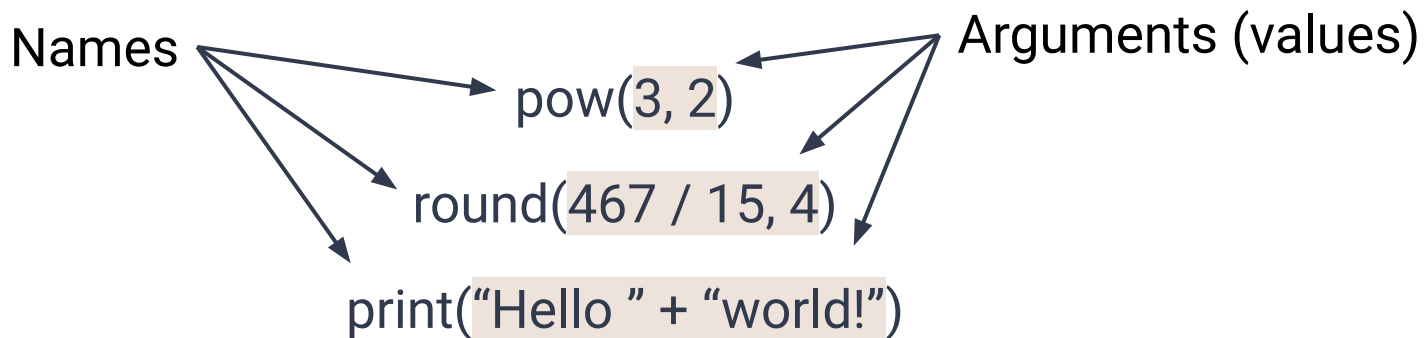
```
print("Hello " + "world!")
```



# Examples



# Examples



# 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:

**What if the function we want doesn't exist...cliffhanger for next lecture :)**

`abs(x)`

`help()`

`min(x,y)`

`max(x,y)`

`pow(x,y)`

`print(x)`

`round(x)` and `round(x,y)`

**Demo in Replit**