

# [301] Variables and Expressions

Tyler Caraza-Harter

# Learning Objectives

## Variables:

- Purpose
- Naming

## Assignment:

- Syntax
- Reassignment

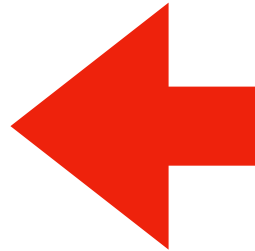
## Types of errors:

- syntax, runtime, semantic

# Today's Outline

## Review

- Operator Precedence



## Expressions, Variables, and Assignments

## Demos

## Bugs



## Demos

## Naming variables

## Demos

## Unordered

What is it?	Python Operator
comparison	==, !=, <, <=, >, >=
signs	+x, -x
AND	and
add/subtract	+, -
exponents	**
NOT	not
OR	or
multiply/divide	*, /, //, %

## Ordered by Precedence

What is it?	Python Operator

**simplify first**

**simplify last**

# Unordered

What is it?	Python Operator
comparison	==, !=, <, <=, >, >=
signs	+x, -x
AND	and
add/subtract	+, -
NOT	not
OR	or
multiply/divide	*, /, //, %

# Ordered by Precedence

What is it?	Python Operator
exponents	**

**simplify first**

**simplify last**

## Unordered

What is it?	Python Operator
comparison	<code>==, !=, &lt;, &lt;=, &gt;, &gt;=</code>
AND	<code>and</code>
add/subtract	<code>+, -</code>
NOT	<code>not</code>
OR	<code>or</code>
multiply/divide	<code>*, /, //, %</code>

## Ordered by Precedence

What is it?	Python Operator
exponents	<code>**</code>
signs	<code>+x, -x</code>

**simplify first**

**simplify last**

## Unordered

What is it?	Python Operator
comparison	==, !=, <, <=, >, >=
AND	and
add/subtract	+, -
NOT	not
OR	or

## Ordered by Precedence

What is it?	Python Operator
exponents	**
signs	+x, -x
multiply/divide	*, /, //, %

**simplify first**

**simplify last**

# Unordered

What is it?	Python Operator
comparison	==, !=, <, <=, >, >=
AND	and
NOT	not
OR	or

# Ordered by Precedence

What is it?	Python Operator
exponents	**
signs	+x, -x
multiply/divide	*, /, //, %
add/subtract	+, -

simplify first

simplify last



# Unordered

What is it?	Python Operator
AND	and
NOT	not
OR	or

# Ordered by Precedence

What is it?	Python Operator
exponents	**
signs	+x, -x
multiply/divide	*, /, //, %
add/subtract	+, -
comparison	==, !=, <, <=, >, >=

simplify first

simplify last

## Unordered

What is it?	Python Operator
AND	and
OR	or

## Ordered by Precedence

What is it?	Python Operator
exponents	**
signs	+x, -x
multiply/divide	*, /, //, %
add/subtract	+, -
comparison	==, !=, <, <=, >, >=
NOT	not

**simplify first**

**simplify last**

## Unordered

What is it?	Python Operator
OR	or

## Ordered by Precedence

What is it?	Python Operator
exponents	**
signs	+x, -x
multiply/divide	*, /, //, %
add/subtract	+, -
comparison	==, !=, <, <=, >, >=
NOT	not
AND	and

**simplify first**

**simplify last**

# Unordered

What is it?	Python Operator

# Ordered by Precedence

What is it?	Python Operator
exponents	**
signs	+x, -x
multiply/divide	*, /, //, %
add/subtract	+, -
comparison	==, !=, <, <=, >, >=
NOT	not
AND	and
OR	or

simplify first

simplify last

# Unordered

What is it?	Python Operator

# Ordered by Precedence

What is it?	Python Operator
exponents	**
signs	+x, -x
multiply/divide	*, /, //, %
add/subtract	+, -
comparison	==, !=, <, <=, >, >=
NOT	not
AND	and
OR	or

simplify first

10 - -2 // 3

simplify last

$$10 - -2 // 3$$

$$10 - -(2 // 3)$$

$$(10 - -2) // 3$$

## Unordered

What is it?	Python Operator

## Ordered by Precedence

What is it?	Python Operator
exponents	**
signs	+x, -x
multiply/divide	*, /, //, %
add/subtract	+, -
comparison	==, !=, <, <=, >, >=
NOT	not
AND	and
OR	or

**simplify first**

**simplify last**

logical operators can "short circuit"

$1+1==2$  **or**  $3 ** 100000000 > 2 ** 200000000$

$3 ** 100000000 > 2 ** 200000000$  **or**  $1+1==2$

$1+1==2$  **or**  $1/0 == 2/0$

$1/0 == 2/0$  **or**  $1+1==2$

False **or** \_\_\_\_\_

$2+3 == 4$  **or** 5

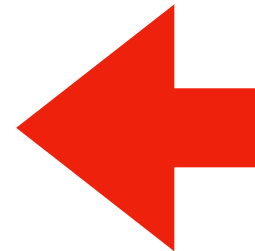




# Today's Outline

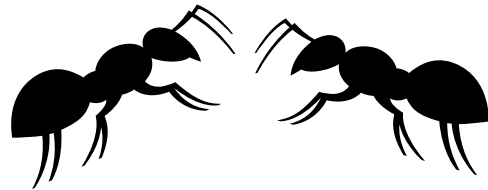
Review

Expressions, Variables, and Assignments



*Demos*

Bugs



*Demos*

Naming variables

*Demos*

# Expressions

Expressions are a mix of operators and operands. For example:

$5 + 5$

$(8/2) ** 2 * 3.14$

$3 * 3 > 4 + 4$

$3 \% 2 == 0$  or  $3 \% 2 == 1$

Each of these operands is an example of a *literal*: a fixed value

# Expressions

Expressions are a mix of operators and operands. For example:

$x + y$

$(\text{diameter}/2) ** 2 * \text{pi}$

$\text{value1} * \text{value1} > \text{value2} + \text{value2}$

$\text{num} \% 2 == 0$  or  $\text{num} \% 2 == 1$

An operand may also be a *variable*: not fixed

# Expressions

Expressions are a mix of operators and operands. For example:

$x + y$

(diameter

value l \*

num % 2

Quick Test! Circle the **literals** (others are **variables**)

1. 0
2. zero
3. num l
4. True
5. hello
6. "goodbye"

An operand may also be a **variable**: not fixed

# Expressions

Expressions are a mix of operators and operands. For example:

$x + y$

(diameter

value l \*

num % 2

Quick Test! Circle the **literals** (others are **variables**)

1. 0

2. zero

3. num l

4. True

5. hello

6. "goodbye"

An operand may also be a **variable**: not fixed

# Expressions

Expressions are a mix of operators and operands. For example:

$x + y$

(diameter

value  $\times$

num  $\%$  2

Quick Test! Circle the **literals** (others are **variables**)

1. 0

2. zero

3. num1

4. True

5. hello

6. "goodbye"

An operand may also be a **variable**: not fixed

# Expressions

Expressions are a mix of operators and operands. For example:

$x + y$

(diameter

value  $\times$

num  $\% 2$

Quick Test! Circle the **literals** (others are **variables**)

1. 0

2. zero

3. num1

4. True

5. hello

6. "goodbye"

An operand may also be a **variable**: not fixed



# Expressions

Expressions are a mix of operators and operands. For example:

$x + y$

(diameter

value l \*

num % 2

Quick Test! Circle the **literals** (others are **variables**)

1. 0

2. zero

3. num l

4. True

5. hello

6. "goodbye"

An operand may also be a **variable**: not fixed

How do we put a value in a variable?

# Assignment

An **assignment** computes an expression (maybe a simple one) and puts the result in a variable:

**x** + **y**

(**diameter**/2) \*\* 2 \* **pi**

**value1** \* **value1** > **value2** + **value2**

**num** % 2 == 0 or **num** % 2 == 1

# Assignment

An **assignment** computes an expression (maybe a simple one) and puts the result in a variable:

$x + y$

$(\text{diameter}/2) ** 2 * \text{pi}$

$\text{value1} * \text{value1} > \text{value2} + \text{value2}$

$\text{num} \% 2 == 0$  or  $\text{num} \% 2 == 1$

# Assignment

An **assignment** computes an expression (maybe a simple one) and puts the result in a variable:

= **x** + **y**

= (**diameter**/2) \*\* 2 \* **pi**

= **value1** \* **value1** > **value2** + **value2**

= **num** % 2 == 0 or **num** % 2 == 1

# Assignment

An **assignment** computes an expression (maybe a simple one) and puts the result in a variable:

```
total = x + y
```

```
area = (diameter/2) ** 2 * pi
```

```
is_bigger = value1 * value1 > value2 + value2
```

```
is_even_or_odd = num % 2 == 0 or num % 2 == 1
```

# Assignment

An **assignment** computes an expression (maybe a simple one) and puts the result in a variable:

**total** = **x** + **y**

**area** = (**diameter**/2) \*\* 2 \* **pi**

**is\_bigger** = **value1** \* **value1** > **value2** + **value2**

**is\_even\_or\_odd** = **num** % 2 == 0 or **num** % 2 == 1

**Expression**

# Assignment

An **assignment** computes an expression (maybe a simple one) and puts the result in a variable:

**total** = **x** + **y**

**area** = (**diameter**/2) \*\* 2 \* **pi**

**is\_bigger** = **value1** \* **value1** > **value2** + **value2**

**is\_even\_or\_odd** = **num** % 2 == 0 or **num** % 2 == 1



**Expression**

**Assignment Operator**

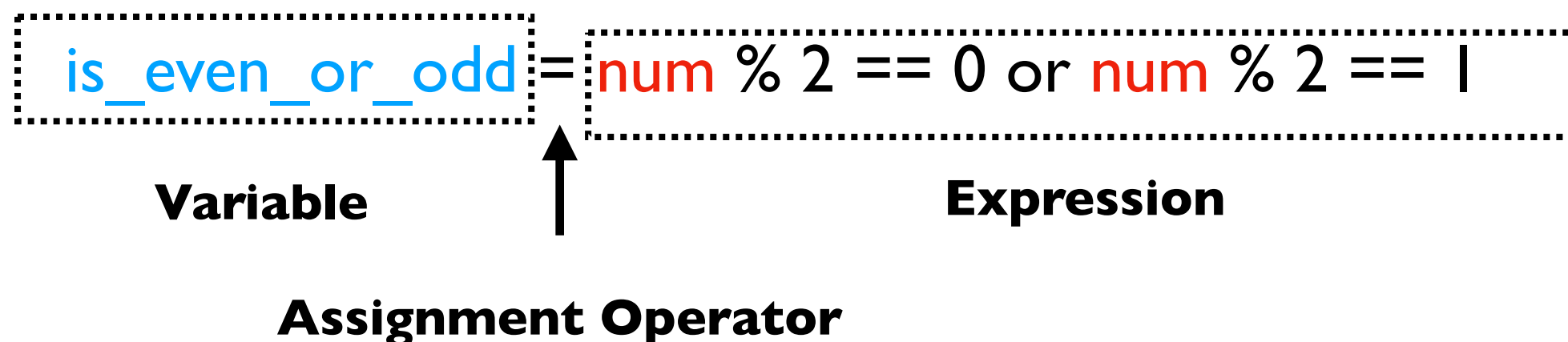
# Assignment

An **assignment** computes an expression (maybe a simple one) and puts the result in a variable:

**total** = **x** + **y**

**area** = (**diameter**/2) \*\* 2 \* **pi**

**is\_bigger** = **value1** \* **value1** > **value2** + **value2**



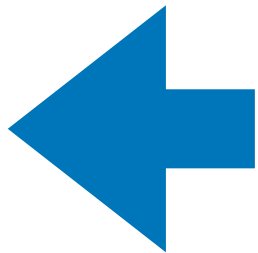


# Today's Outline

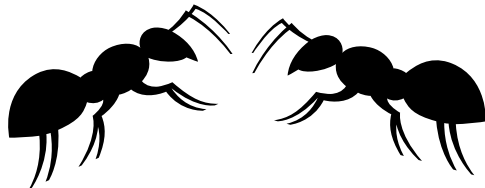
Review

Expressions, Variables, and Assignments

*Demos*



Bugs



*Demos*

Naming variables

*Demos*

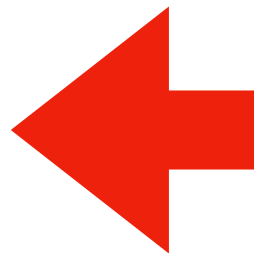
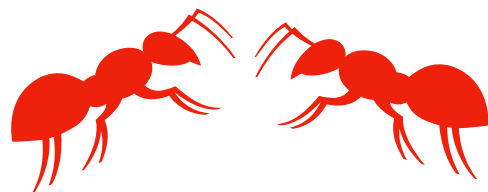
# Today's Outline

Review

Expressions, Variables, and Assignments

*Demos*

Bugs



*Demos*

Naming variables

*Demos*

# Categories of Errors

1

dog cat the of chase any  
[word soup, not grammatically sensible]

2

3

# Categories of Errors

1

## Syntax Error

- It never makes sense in any context; Python doesn't even run
- 5 = x

2

3

# Categories of Errors

1

## Syntax Error

- It never makes sense in any context; Python doesn't even run
- `5 = x`

2

this sentence is false

[grammatical, but my head explodes if I think about it]

3

# Categories of Errors

1

## Syntax Error

- It never makes sense in any context; Python doesn't even run
- `5 = x`

2

## Runtime Error

- Need to run to find out whether it will crash
- Appears with different names (TypeError, ZeroDivisionError, etc)
- `x = 5 / 0`

3

# Categories of Errors

1

## Syntax Error

- It never makes sense in any context; Python doesn't even run
- `5 = x`

2

## Runtime Error

- Need to run to find out whether it will crash
- Appears with different names (TypeError, ZeroDivisionError, etc)
- `x = 5 / 0`

3

one week is 10 days long  
[grammatical, coherent, but incorrect]

# Categories of Errors

1

## Syntax Error

- It never makes sense in any context; Python doesn't even run
- `5 = x`

2

## Runtime Error

- Need to run to find out whether it will crash
- Appears with different names (TypeError, ZeroDivisionError, etc)
- `x = 5 / 0`

3

## Semantic Error

- It runs with no error, but you get the wrong answer
- `square_area = square_side * 2`



# Categories of Errors

1

## Syntax Error

- It never makes sense in any context; Python doesn't even run
- `5 = x`

2

## Runtime Error

- **what kind of error is the worst?**
- `x = 5 / 0`

3

## Semantic Error

- It runs with no error, but you get the wrong answer
- `square_area = square_side * 2`

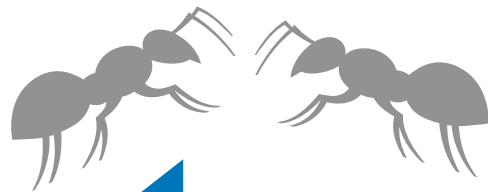
# Today's Outline

Review

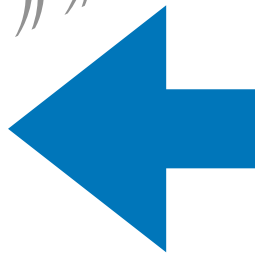
Expressions, Variables, and Assignments

*Demos*

Bugs



*Demos*



Naming variables

*Demos*

# int Demo

```
a = 3  
b = 4  
c = 5  
d = 6
```

**What is the sum of all the odd values?  
(in this case, 3 + 5)**



# float Demo

## Compound growth:

- you start with **\$1000**
- every year it grows by **7%**
- you wait **30 years**
- how much do you have at the end?

**year 0:** \$1000

**year 1:** \$1070

**year 2:** ...



# str Demo

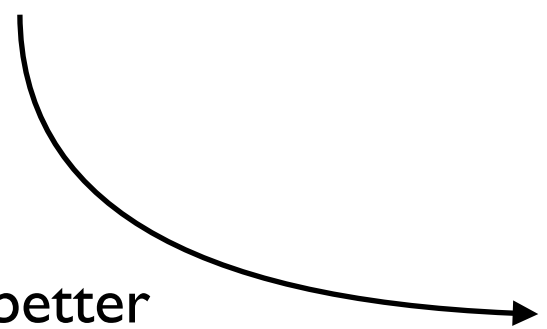
## Visually compare two scores:

- Alice has 10 points
- Bob has 8 points

## Desired output:

```
alice: | | | | | | | | | |
bob:   | | | | | | | |
```

even better



```
alice: | | | | | | | | | |
bob:   | | | | | | | |
```

# bool Demo

**Bounds check:** is the value between 0 and 100?

**YES**

**output is**

you may continue: True

**NO**

**output is**

you may continue: False

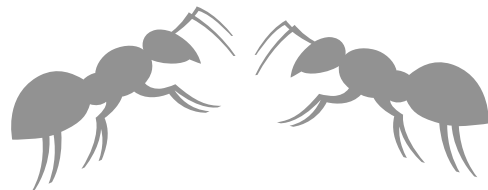
# Today's Outline

Review

Expressions, Variables, and Assignments

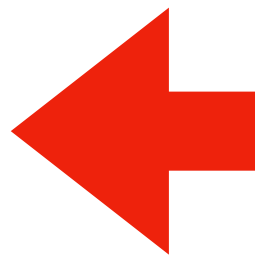
*Demos*

Bugs



*Demos*

Naming variables



*Demos*

# What Variable Names are Allowed?

`1st_score = 100` [bad variable]

`score_1 = 100` [good variable]

current rules are quite complex:

<https://www.python.org/dev/peps/pep-3131>

Python 3 has become friendlier to non-English programmers

`quero_café = ...`

this is allowed, and  
different than "e"





# What Variable Names are Allowed?

`1st_score = 100` [bad variable]

`score_1 = 100` [good variable]

current rules are quite complex:

<https://www.python.org/dev/peps/pep-3131>

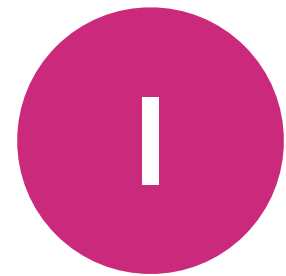
Python 3 has become friendlier to non-English programmers

`quero_café = True`

this is allowed, and  
different than "e"



# Conservative Rules for English Code



Only use letters a-z (upper and lower), numbers, and underscores



Don't start with a number



Don't use Python keywords (e.g., and, False, etc)

for 220, you may use characters from any script and variables in any language you prefer, but we won't cover variable naming rules for any other language

# Conservative Rules for English Code

1

Only use letters a-z (upper and lower), numbers, and underscores

2

Don't start with a number

3

Don't use Python keywords (e.g., and, False, etc)

## **GOOD:**

cs220  
CS220  
cs\_220  
\_cs220

## **BAD:**

220class  
and  
pi3.14  
x!

*what rules are violated?*

# Conservative Rules for English Code

1

Only use letters a-z (upper and lower), numbers, and underscores

2

Don't start with a number

3

Don't use Python keywords (e.g., and, False, etc)

## GOOD:

cs220  
CS220  
cs\_220  
\_cs220

## BAD:

220class  
and  
pi3.14  
x!

2

# Conservative Rules for English Code

1

Only use letters a-z (upper and lower), numbers, and underscores

2

Don't start with a number

3

Don't use Python keywords (e.g., and, False, etc)

## GOOD:

cs220  
CS220  
cs\_220  
\_cs220

## BAD:

220class 2  
and 3  
pi3.14  
x!

# Conservative Rules for English Code

1

Only use letters a-z (upper and lower), numbers, and underscores

2

Don't start with a number

3

Don't use Python keywords (e.g., and, False, etc)

## GOOD:

```
cs220  
CS220  
cs_220  
_cs220
```

## BAD:

```
220class 2  
and 3  
pi3.14 1  
x! 1
```

# Conservative Rules for English Code

1

Only use letters a-z (upper and lower), numbers, and underscores

2

Don't start with a number

3

Don't use Python keywords (e.g., and, False, etc)

## GOOD:

cs220  
CS220  
cs\_220  
\_cs220

## BAD:

220class 2  
and 3  
pi3.14 1  
x! 1

PLEASE never name a variable after a type (e.g., int, str, etc)

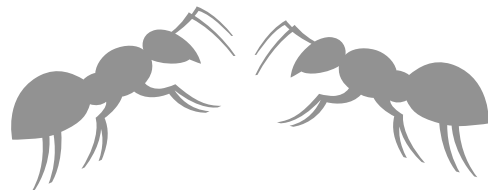
# Today's Outline

Review

Expressions, Variables, and Assignments

*Demos*

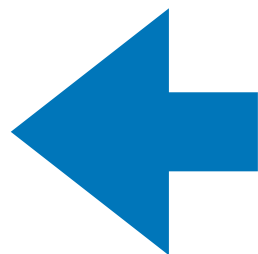
Bugs



*Demos*

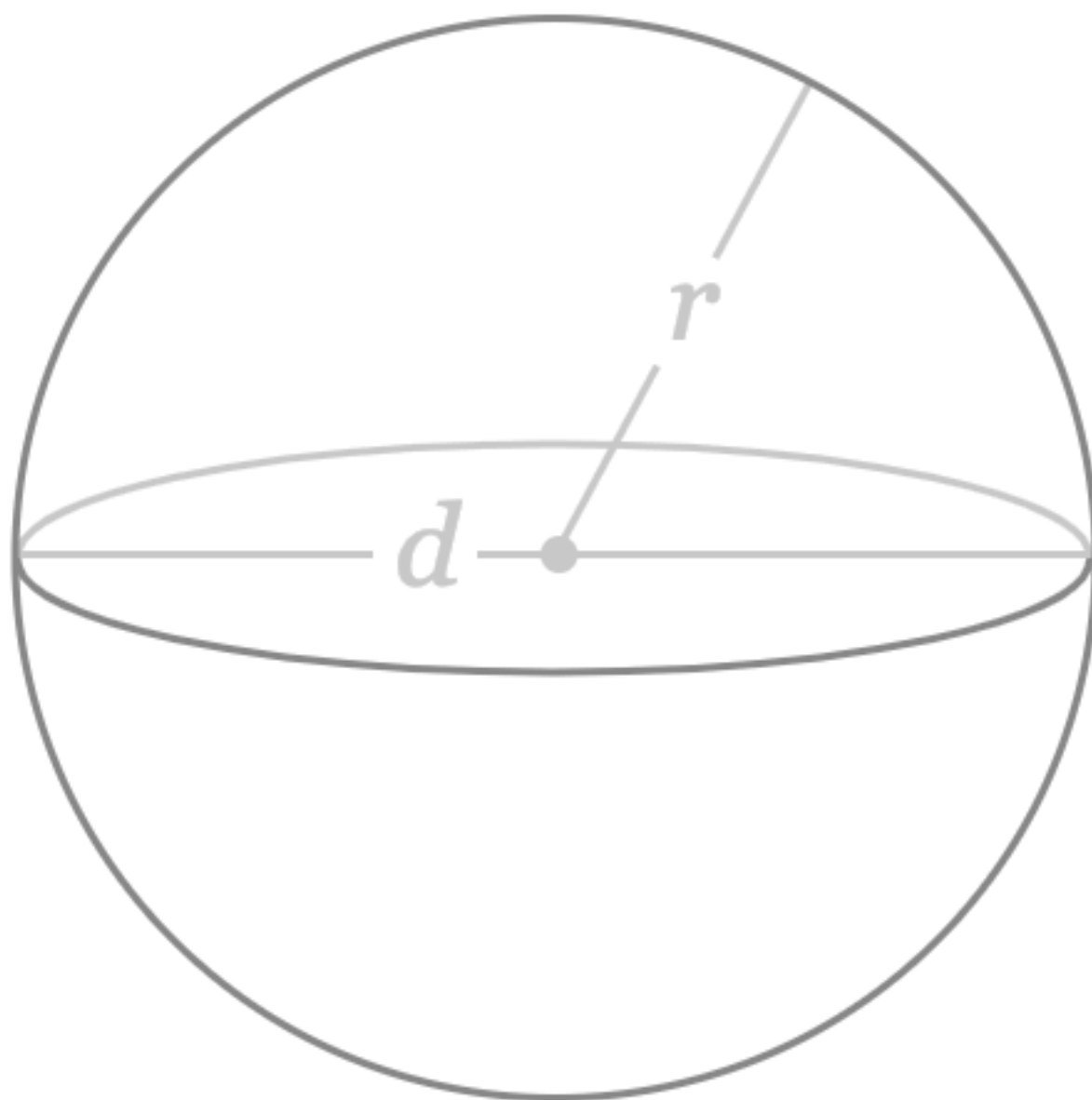
Naming variables

*Demos*





# Sphere Volume Demo



$$V = \frac{4}{3} \pi r^3$$

**bonus:** find radius given a volume

# Quadratic Formula Demo

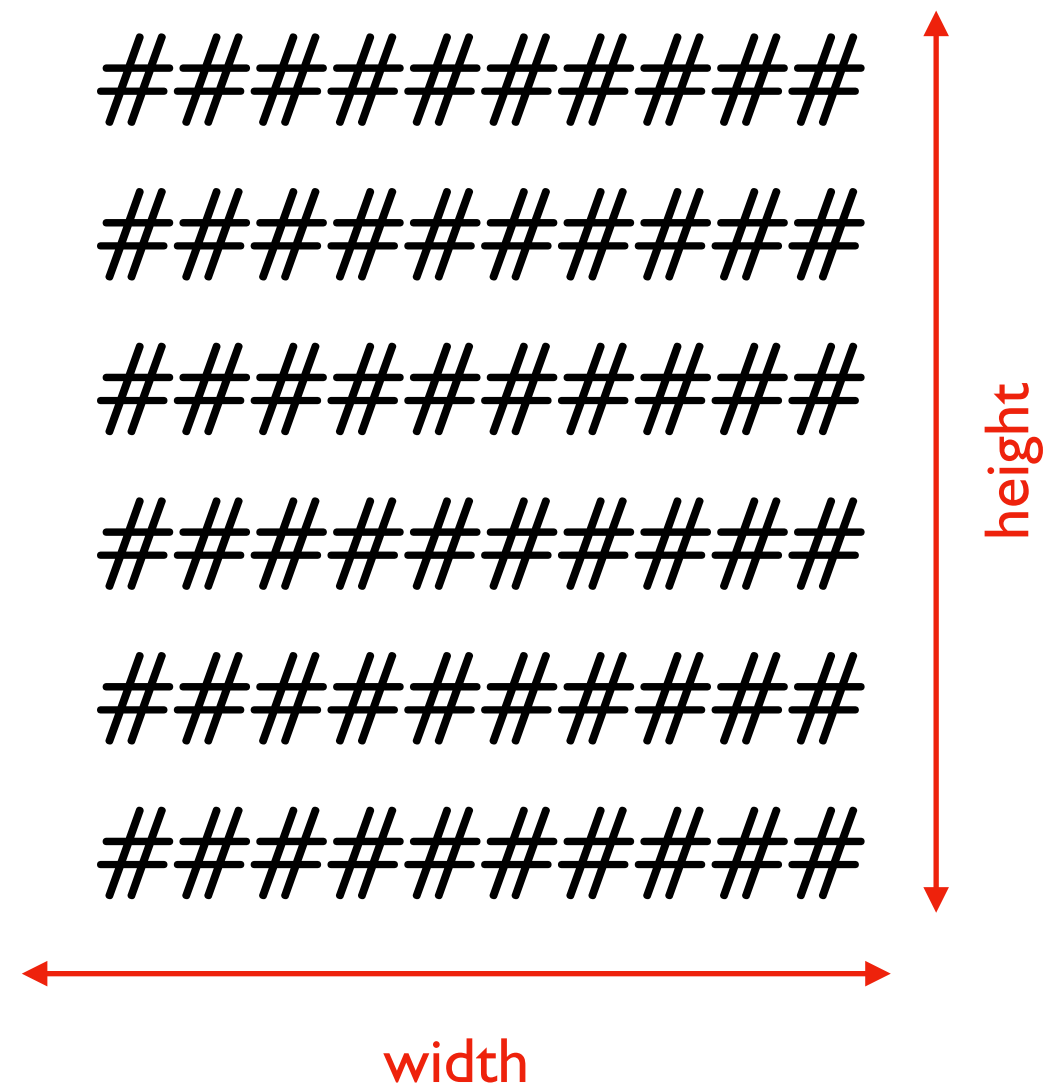
$$ax^2 + bx + c = 0$$

*what values of x satisfy the above?*

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

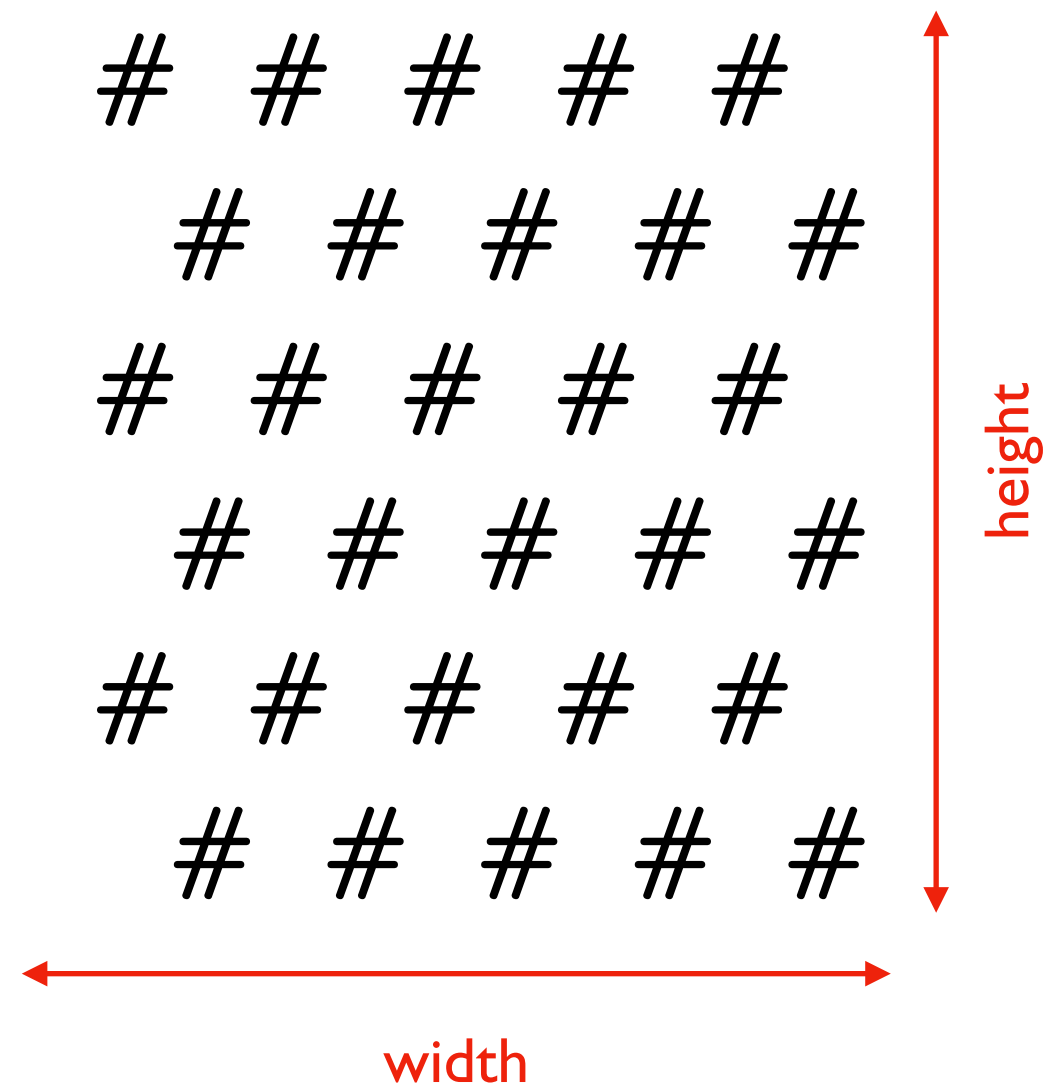
# Character Art: Block

write some code to draw the following:



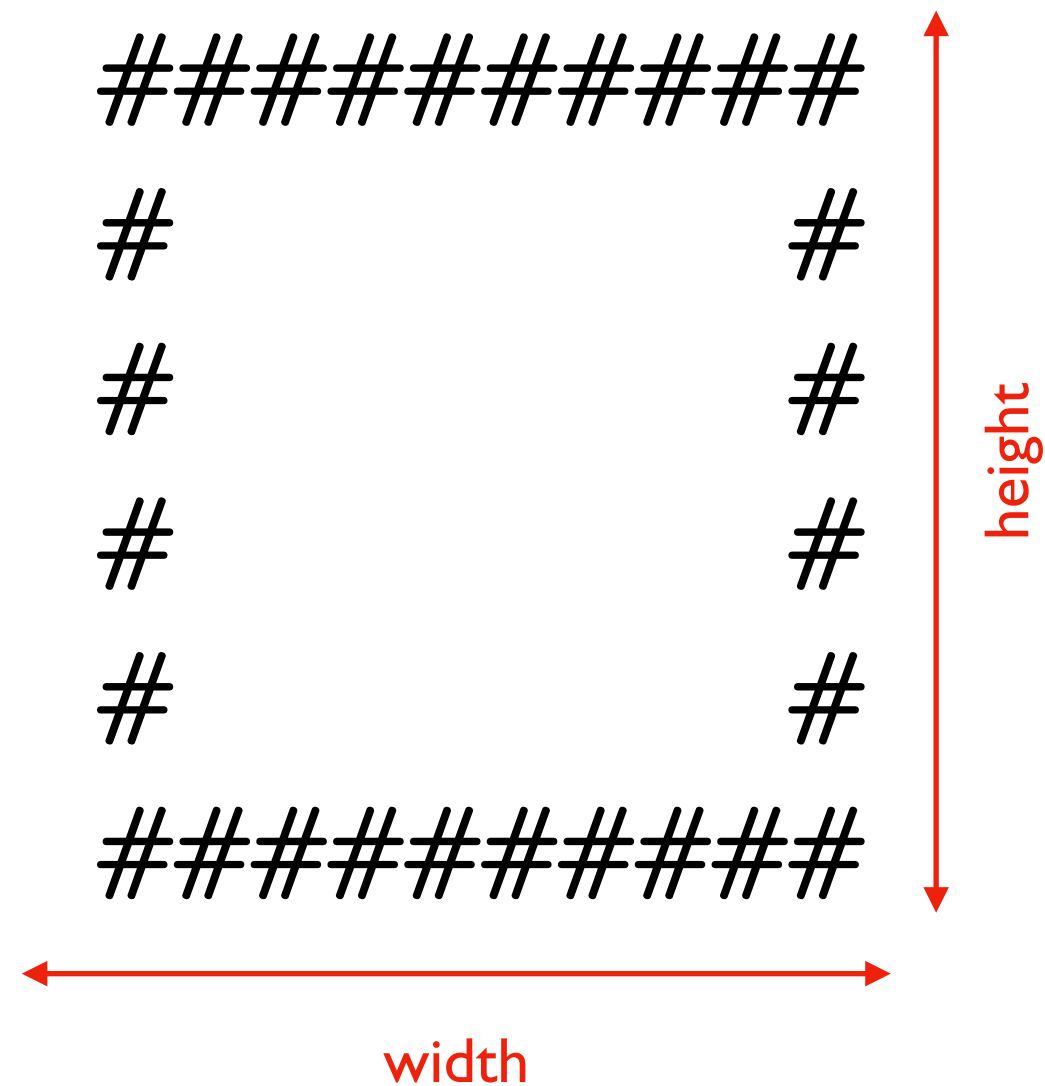
# Character Art: Checkers

write some code to draw the following:



# Character Art: Border

write some code to draw the following:



# Character Art: Snake

write some code to draw the following:

