[301] Variables and Expressions

Tyler Caraza-Harter

Learning Objectives

Variables:

- Purpose
- Naming

Assignment:

- Syntax
- Reassignment

Types of errors:

- syntax, runtime, semantic

Documentation:

- comments

Please read Chapter 2

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	==, !=, <, <=, >, >=
AND	and
add/subtract	+, -
NOT	not
OR	or
multiply/divide	*, /, //, %

Ordered by Precedence

simplify first

What is it?	Python Operator
exponents	**
signs	+x, -x

simplify last

Unordered

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

Ordered by Precedence

simplify first

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

simplify last

Unordered

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

Ordered by Precedence

simplify first

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

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

simplify first

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

simplify last

Unordered

What is it?	Python Operator
OR	or

Ordered by Precedence

simplify first

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

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

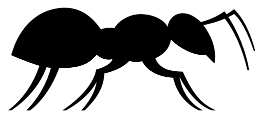
simplify last

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
```

Expressions

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

```
x + y

(diameter/2) ** 2 * pi

value1 * value1 > value2 + value2

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

An operand could be a “variable” instead of value

Expressions

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

```
x + y

(diameter/2) ** 2 * pi

value1 * value1 > value2 + value2

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

An operand could be a “variable” instead of value

How do variables get associated with a value?

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
```

Variable **Expression**

Assignment Operator

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

Syntax Error

- What you typed never makes sense in any context, so Python doesn't run
- 5 = x

Runtime Error

- What you typed looks like it could make sense, but when you get to running it, it actually doesn't work
- Appears with different names (TypeError, ZeroDivisionError, etc)
- x = 5 / 0

Semantic Error

- What you did actually runs, but produces the wrong answer
- square_area = square_side * 2

Categories of Errors

Syntax Error

- What you typed never makes sense in any context, so Python doesn't run
- 5 = x

Which kind of bugs do you think scare programmers the most?

Runtime Error

- What you typed looks like it could make sense, but when you get to running it, it actually doesn't work
- Appears with different names (TypeError, ZeroDivisionError, etc)
- x = 5 / 0

Semantic Error

- What you did actually runs, but produces the wrong answer
- square_area = square_side * 2

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

Python Naming Rules

Variable naming rules have actually become fairly complex:

- https://www.python.org/dev/peps/pep-3131
- Motivation: why should everybody be forced to program in English?

Python Naming Rules

Variable naming rules have actually become fairly complex:

- https://www.python.org/dev/peps/pep-3131
- Motivation: why should everybody be forced to program in English?

Conservative rules that will work nearly everywhere:

- Only use letters (upper and lower), numbers, and underscores
- Don’t start with a number
- Don’t use Python keywords (e.g., “and”)

Python Naming Rules

Variable naming rules have actually become fairly complex:

- https://www.python.org/dev/peps/pep-3131
- Motivation: why should everybody be forced to program in English?

Conservative rules that will work nearly everywhere:

- Only use letters (upper and lower), numbers, and underscores
- Don't start with a number
- Don't use Python keywords (e.g., "and")

GOOD:	**BAD:**
cs301	**301class**
CS301	**and**
cs_301	**pi3.14**
_cs301	**x!**

Today's Outline

Review

Expressions, Variables, and Assignments

Demos

Bugs

Demos

Naming variables

Demos