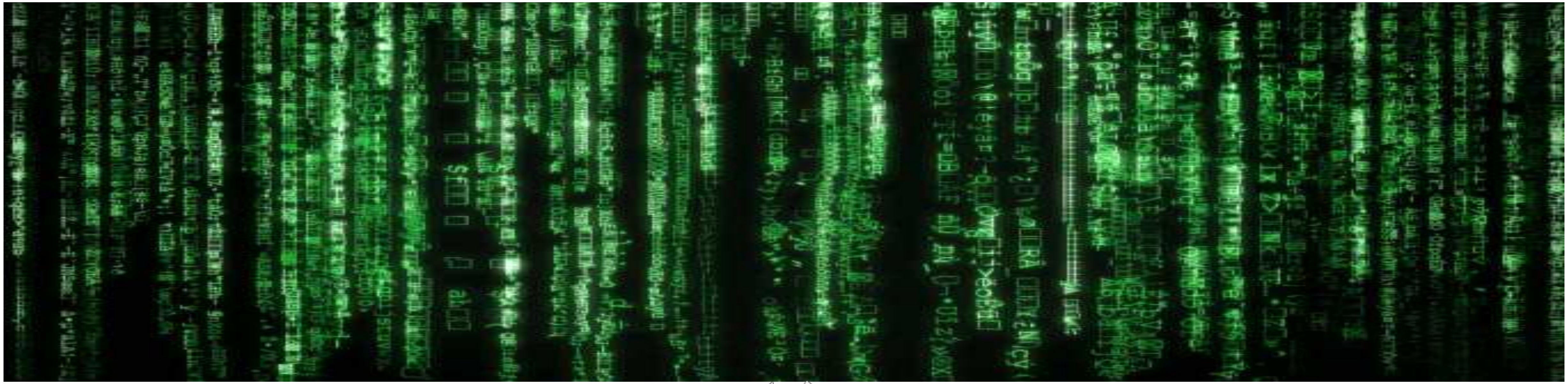


INF 110 Discovering Informatics

More Expressions

NORTHERN ARIZONA UNIVERSITY

Logical Operations

- `x and y`
- `x or y`
- `not x`

A	B	A and B	A or B
False	False	False	False
False	True	False	True
True	False	False	True
False	False	True	True

- `True and True -> True`
- `True or True -> True`

If statements

An if statement uses logical expressions to conditionally evaluate statements

```
if <expr>:
   statement1
   statement2
   …
elif <expr>:
   statement1
   statement2
   …
else:
   statement1
   statement2
   …
```

Live Code How’s the Weather?

Task: Write an if statement that provides a statement about the weather based on the temperature

Learning Outcomes

- Designing if statements
- Solving problems with comparison operators
- Solving problems with logical operators

Function Composition

```python
# First we associate the string with a variable
x = "Plum"

# Second we find the length of x
x_length = len(x)

# Third we convert it to binary
bin_string = bin(x_length)

# Fourth, we print the result
print(bin_string)
```

Function Composition

By using function composition, we can combine all of those steps into a single line:

```
print(bin(len("Hello")))
```

Function Composition

```
number = input("Enter␣a␣number:␣")
base = input("Enter␣its␣base:␣")

print(int(number, int(base)))
```

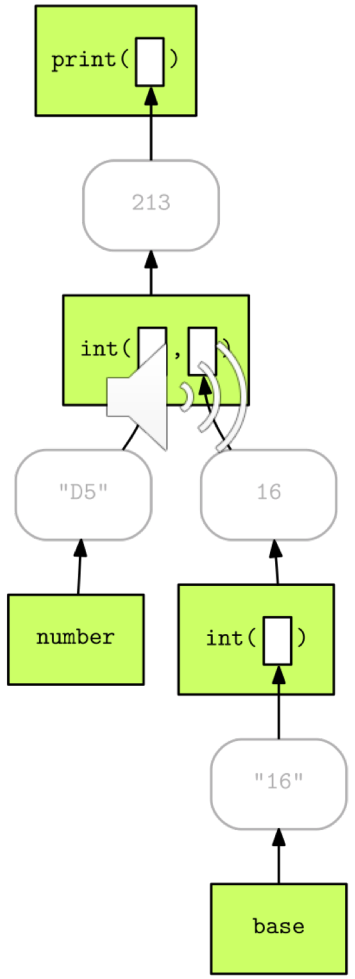
Subexpressions

A logically complete expression that is part of a compound expression.

```
print(bin(len(x) + 3))
```

- What are the subexpressions?
- What wouldn't be a subexpression?

Expression Tree Evaluation



Method Chaining

```
x = "␣␣␣flagstaff␣␣␣␣"

# Remove the whitespace
x = x.strip()

# Capitalize
x = x.capitalize()

print(x)
```

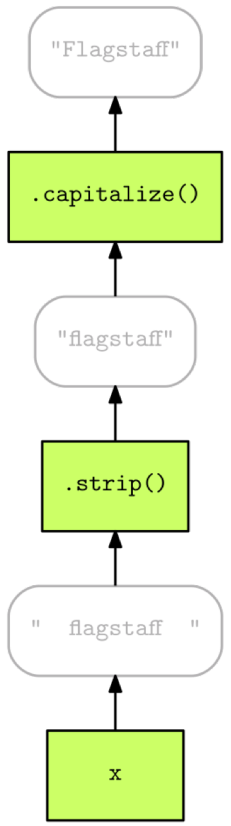
Method Chaining

The same thing can be accomplished using method chaining:

```
x = "␣␣␣flagstaff␣␣␣␣"

print(x.strip().capitalize())
```

Method Chaining Expression Tree



Operator Precedence

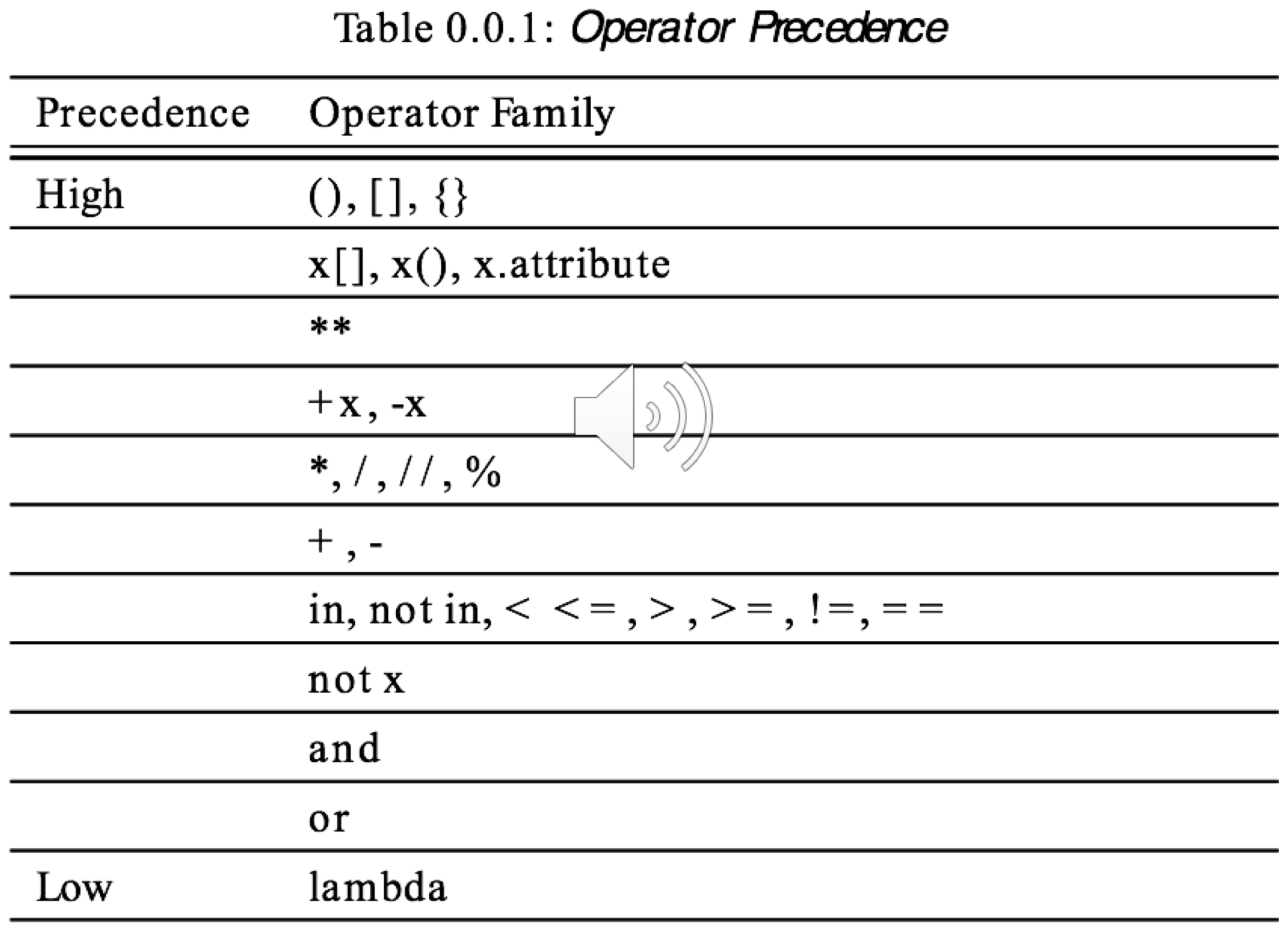
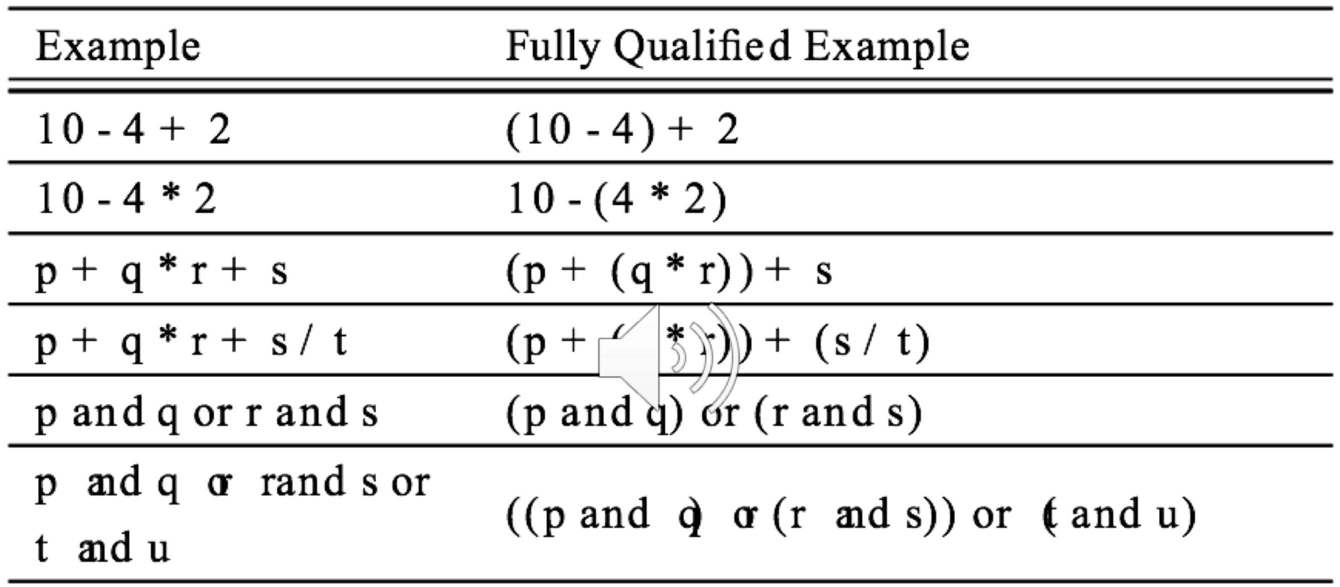


Table 0.0.1: *Operator Precedence*

Precedence	Operator Family
High	(), [], {}
	x[], x(), x.attribute
	**
	+x, -x
	*, / , // , %
	+ , -
	in, not in, < <= , > , >= , != , ==
	not x
	and
	or
Low	lambda

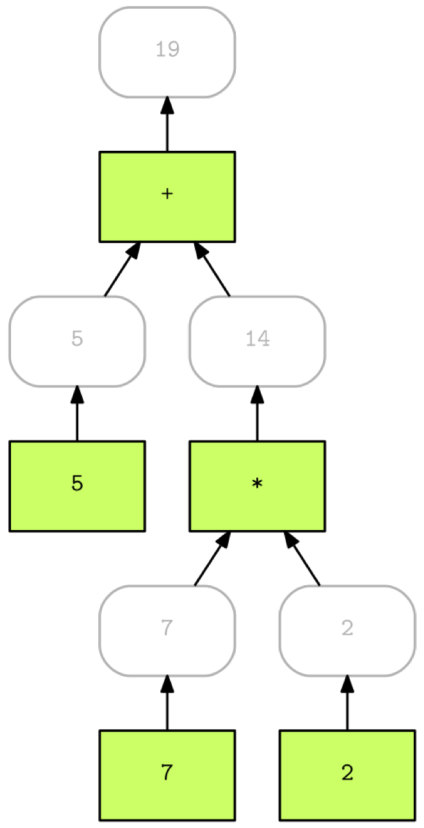
Operator Precedence

Table 0.0.1: *Operator Precedence Examples*



Example	Fully Qualified Example
10 - 4 + 2	(10 - 4) + 2
10 - 4 * 2	10 - (4 * 2)
p + q * r + s	(p + (q * r)) + s
p + q * r + s / t	(p + (q * r)) + (s / t)
p and q or r and s	(p and q) or (r and s)
p and q or r and s or t and u	((p and q) or (r and s)) or (t and u)

Operator Precedence



Augmented Assignment Operators

- `x += y` means `x = x + y`
- `x -= y` means `x = x - y`
- `x *= y` means `x = x * y`
- `x /= y` means `x = x / y`

Augmented Assignment Operators

```
number_of_widgets = 7

# We can add one the long way..
number_of_widgets = number_widgets + 1

# Or the short way..
number_of_widgets += 1
```

Live Code Vinyl Record Sales?

Task: What is the **percent difference** in record sales for these two years?

14.32 million in 2017

13.1 million in 2016

Learning Outcomes

- Solving problems with math operators
- Using data to make inferences

end