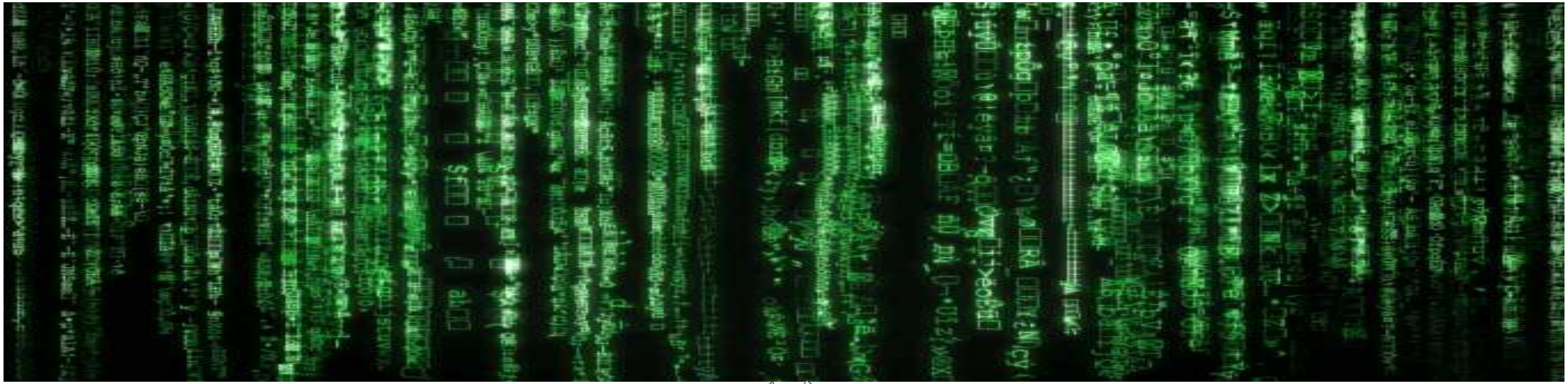


INF 110 Discovering Informatics

Python Expressions (part 1)

NAU NORTHERN ARIZONA UNIVERSITY

Why Python?

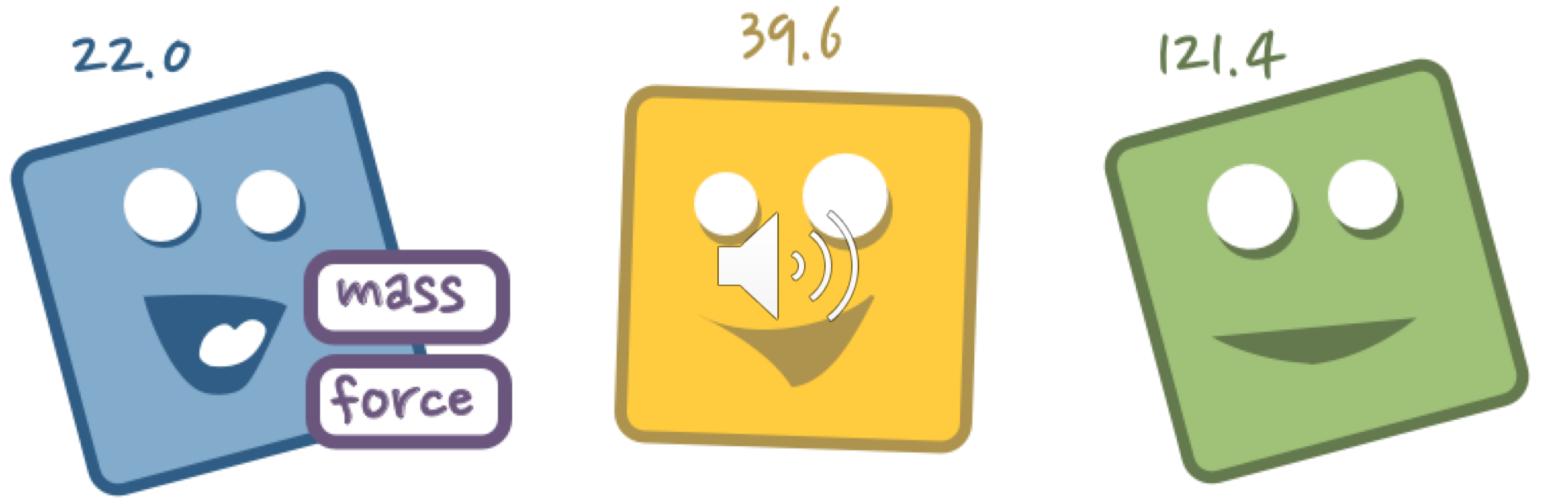
- Python is *simple*
- Python is *easy to learn*
- Python is *free*
- Python is a *community*
- Python is a *high-level language*
- Python is commonly used in *informatics & data science*

Variables

22.0
mass
39.6
force
121.4

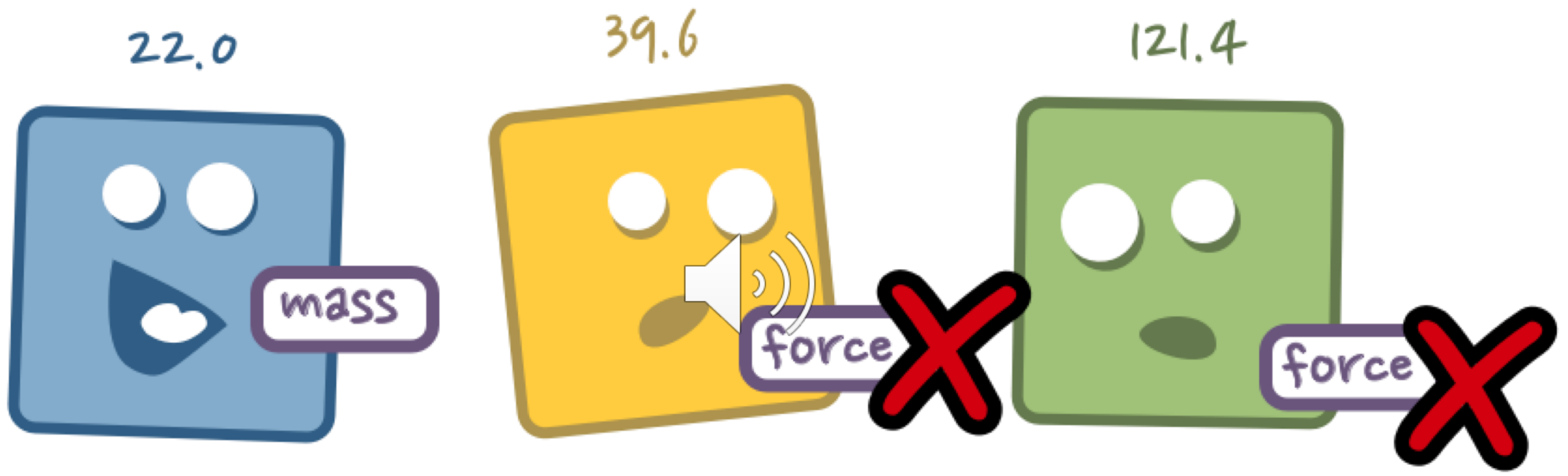
A variable connects a name to one value

Variables



One value could also wear multiple labels - meaning that a single value is connected to multiple variables.

Variables



But one important rule at this party is that labels must be **unique** - two values can't have the same label (i.e., variable name) at once.

Variable Assignment

Variables are connected to values through through **assignment** - this is how we make a value wear a label.

```
mass = 22.0
```

Consider this example where a new value associated with the variable force (39.6) is derived from a standard physics relationship:

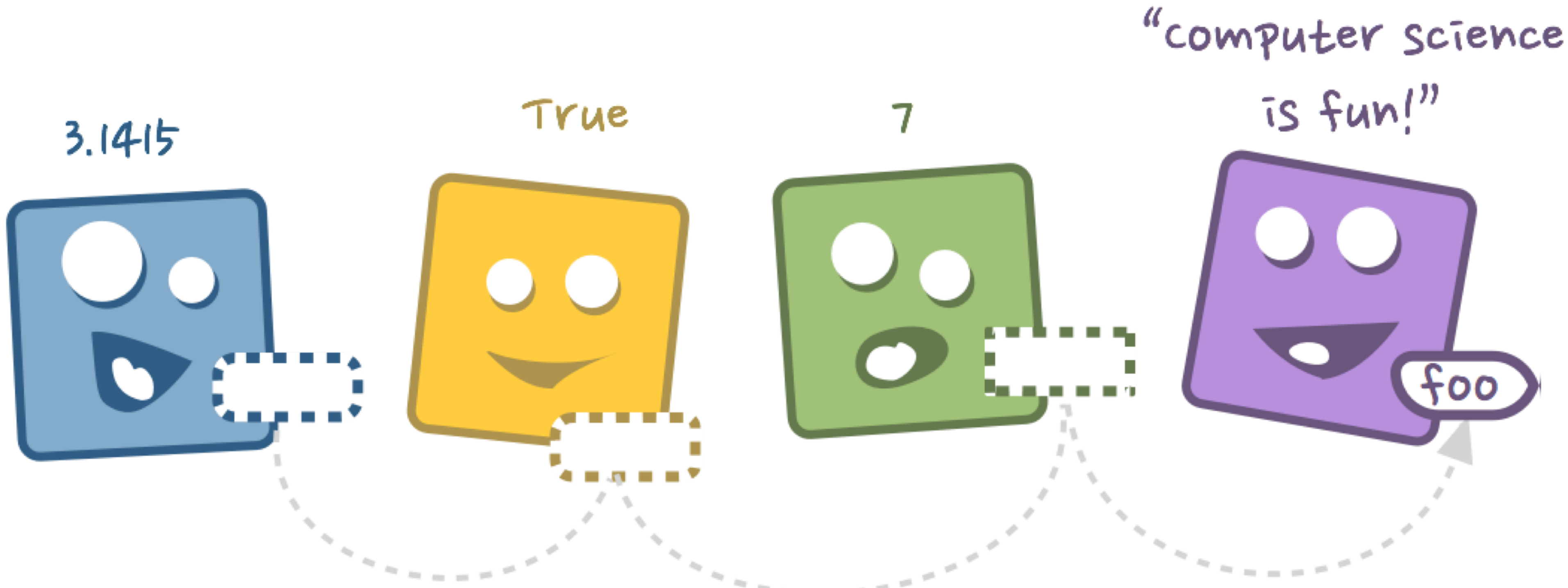
```
mass = 22.0
acceleration = 1.8
force = mass * acceleration
force -> 39.6
```

Variable Assignment

Here we illustrate dynamic typing by letting the variable foo take on four different values and four different corresponding types:

```
foo = 3.1415
foo = True
foo = 7
foo = "Computer␣science␣is␣fun!"
```

Variable Assignment



Rules for Naming Variables

- variable names are lower case and words are separated with underscores (e.g., standard_deviation),
- class names are title case (e.g., ColorMatrix),
- identifiers that begin with one or more underscores have special meaning
- identifiers shouldn't have the same name as built-in identifiers (e.g., int , float, list, tuple, dir).

Choosing **Good** Variable Names

- Is the name consistent with existing naming conventions?
- Does this value have important units (grams, meters, moles, etc.) that are not obvious from its type or usage?
- Does the name unnecessarily use negative logic or other counter intuitive conventions? You should consider using is_enabled instead of is_not_enabled.

Choosing **Good** Variable Names

- Is the name descriptive?
- If you had seen this variable for the first time would the name make sense?
- Is the name too wordy, long, or redundant?
- Is the name too short or does it use uncommon abbreviations?

Code Should Read Like Poetry

Consider this perfectly correct piece of code:

```
a = (1/2) * b * c
```

Choose names that reveal the codes purpose:

```
triangle_area = (1/2) * base * height
```

Some Useful Types

- **Strings** - sequences of characters

```
"Discovering Informatics!”
```

- **Integers** – whole numbers

```
1138
```

- **Floats** – real numbers

```
3.1415
```

Basic Math Operations

`x + y`	Addition
`x - y`	Subtraction
`x * y`	Multiplication
`x / y`	Division
`x ** y`	Exponentiation
`abs(x)`	Absolute Value

Basic Comparison Operations

`x < y`	Less than
`x <= y`	Less than or equal to
`x > y`	Greater than
`x >= y`	Greater than or equal to
`x == y`	Equal to

Warning: Assignment (=) and comparison (==) are different!

Basic Conversions

• `str(x)`	Convert to a string
• `int(x)`	Convert to an integer
• `float(x)`	Convert to a float

Lists and Dictionaries

```
>>> fruit = ["Apples", "Bananas", "Mangoes"]
```

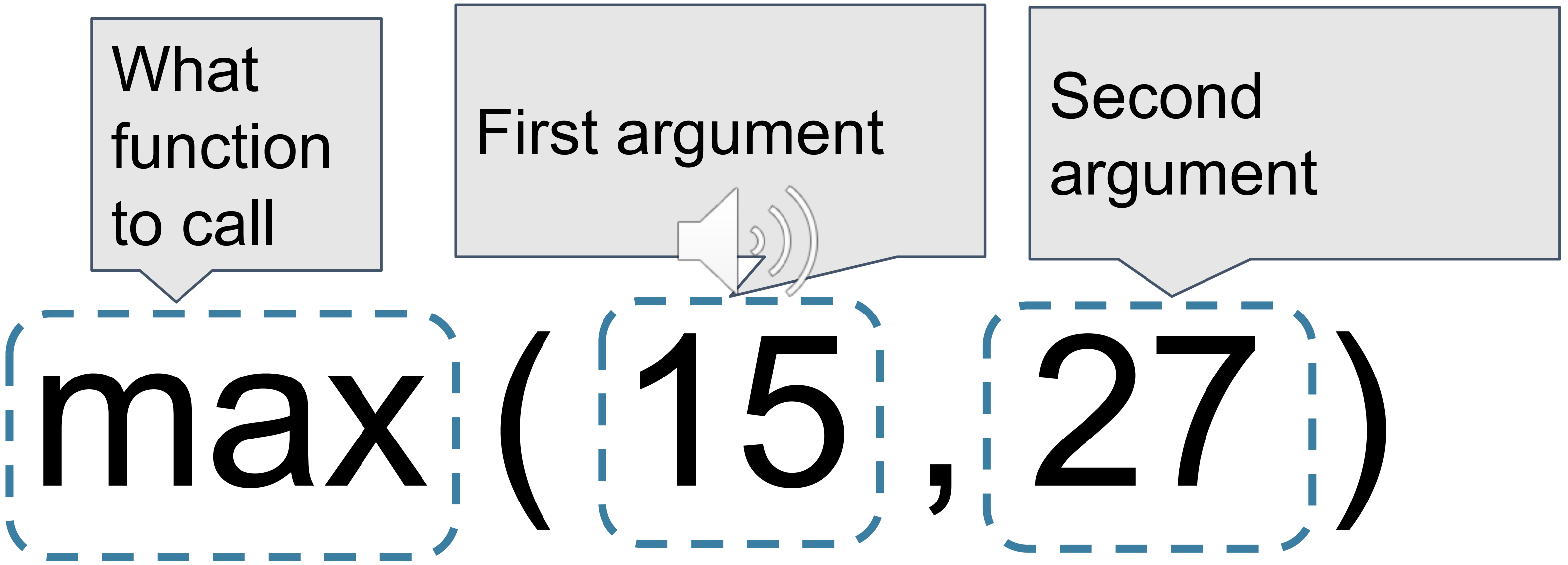
```
>>> fruit[0]
'Apples'
```

Lists and Dictionaries

```
>>> color_frequency = {"red": 650,
                       "green": 510,
                       "blue": 475}
```

```
>>> color_frequency["red"]
650
```

Calling Functions



Calling Methods

- Methods are special functions attached to objects with a dot

```
title = "Gone west"
title.upper()
```

- Methods can also be chained:

```
title.upper().replace("WEST", "FISHING")
```

Note: Case for strings and variable names matters!

end