[301] JSON

Tyler Caraza-Harter

Learning Objectives Today

JSON

- differences with Python syntax
- creating JSON files
- reading JSON files

Python Data Structures and File Formats

Python

```
[
 ["name", "x", "y"],
 ["alice", 100, 150],
 ["bob", -10, 80]
]
```

list of lists

File

```
name,x,y
alice,100,150
bob,-10,80
```

CSV file

We can use CSV files to store data we would want in lists of lists

Python Data Structures and File Formats

Python

File

```
[
 [“name”, “x”, “y”],
 [“alice”, 100, 150],
 [“bob”, -10, 80]
]
```

list of lists

```
name,x,y
alice,100,150
bob,-10,80
```

CSV file

```
{
  “alice”: {
    “age”: 40,
    “scores”: [10,20,19]},
  “bob”: {
    “age”: 45,
    “scores”: [15,23,17,15]}
}
```

dict of dicts

?

Python Data Structures and File Formats

Python

File

```
[
 ["name", "x", "y"],
 ["alice", 100, 150],
 ["bob", -10, 80]
]
```

list of lists

⟷

```
name,x,y
alice,100,150
bob,-10,80
```

CSV file

```
{
  "alice": {
    "age": 40,
    "scores": [10,20,19]},
  "bob": {
    "age": 45,
    "scores": [15,23,17,15]}
}
```

dict of dicts

⟷

```
{
  "alice": {
    "age": 40,
    "scores": [10,20,19]},
  "bob": {
    "age": 45,
    "scores": [15,23,17,15]}
}
```

JSON file

Python Data Structures and File Formats

Python | **File**

```
[
  [“name”, “x”, “y”],
  [“alice”, 100, 150],
  [“bob”, -10, 80]
]
```

list of lists

```
name,x,y
alice,100,150
bob,-10,80
```

CSV file

JSON files look almost identical to Python code for data structures!

```
{
  “alice”: {
    “age”: 40,
    “scores”: [10,20,19]},
  “bob”: {
    “age”: 45,
    “scores”: [15,23,17,15]}
}
```

dict of dicts

```
{
  “alice”: {
    “age”: 40,
    “scores”: [10,20,19]},
  “bob”: {
    “age”: 45,
    “scores”: [15,23,17,15]}
}
```

JSON file

Python Data Structures and File Formats

Python

File

JSON files look almost identical to Python code for data structures!

```
[
  ["name", "x", "y"],
  ["alice", 100, 150],
  ["bob", -10, 80]
]
```

list of lists

```
name,x,y
alice,100,150
bob,-10,80
```

CSV file

dicts use curly braces

```
{
  "alice": {
    "age": 40,
    "scores": [10,20,19]},
  "bob": {
    "age": 45,
    "scores": [15,23,17,15]}
}
```

dict of dicts

```
{
  "alice": {
    "age": 40,
    "scores": [10,20,19]},
  "bob": {
    "age": 45,
    "scores": [15,23,17,15]}
}
```

JSON file

Python Data Structures and File Formats

Python | **File**

```
[
  ["name", "x", "y"],
  ["alice", 100, 150],
  ["bob", -10, 80]
]
```

list of lists

```
name,x,y
alice,100,150
bob,-10,80
```

CSV file

JSON files look almost identical to Python code for data structures!

keys are separated from values with a colon

```
{
  "alice": {
    "age": 40,
    "scores": [10,20,19]},
  "bob": {
    "age": 45,
    "scores": [15,23,17,15]}
}
```

dict of dicts

```
{
  "alice": {
    "age": 40,
    "scores": [10,20,19]},
  "bob": {
    "age": 45,
    "scores": [15,23,17,15]}
}
```

JSON file

Python Data Structures and File Formats

Python **File**

```
[
 ["name", "x", "y"],
 ["alice", 100, 150],
 ["bob", -10, 80]
]
```

list of lists

```
name,x,y
alice,100,150
bob,-10,80
```

CSV file

JSON files look almost identical to Python code for data structures!

```
{
  "alice": {
    "age": 40,
    "scores": [10,20,19]},
  "bob": {
    "age": 45,
    "scores": [15,23,17,15]}
}
```

dict of dicts

lists use square brackets

```
{
  "alice": {
    "age": 40,
    "scores": [10,20,19]},
  "bob": {
    "age": 45,
    "scores": [15,23,17,15]}
}
```

JSON file

Python Data Structures and File Formats

Python

File

JSON files look almost identical to Python code for data structures!

```
[
  ["name", "x", "y"],
  ["alice", 100, 150],
  ["bob", -10, 80]
]
```

list of lists

```
name,x,y
alice,100,150
bob,-10,80
```

CSV file

strings are in quotes

```
{
  "alice": {
    "age": 40,
    "scores": [10,20,19]},
  "bob": {
    "age": 45,
    "scores": [15,23,17,15]}
}
```

dict of dicts

```
{
  "alice": {
    "age": 40,
    "scores": [10,20,19]},
  "bob": {
    "age": 45,
    "scores": [15,23,17,15]}
}
```

JSON file

Python Data Structures and File Formats

Python | **File**

```
[
  ["name", "x", "y"],
  ["alice", 100, 150],
  ["bob", -10, 80]
]
```

list of lists

```
name,x,y
alice,100,150
bob,-10,80
```

CSV file

JSON files look almost identical to Python code for data structures!

integers look like integers

```
{
  "alice": {
    "age": 40,
    "scores": [10,20,19]},
  "bob": {
    "age": 45,
    "scores": [15,23,17,15]}
}
```

dict of dicts

```
{
  "alice": {
    "age": 40,
    "scores": [10,20,19]},
  "bob": {
    "age": 45,
    "scores": [15,23,17,15]}
}
```

JSON file

JSON

Stands for **JavaScript Object Notation**

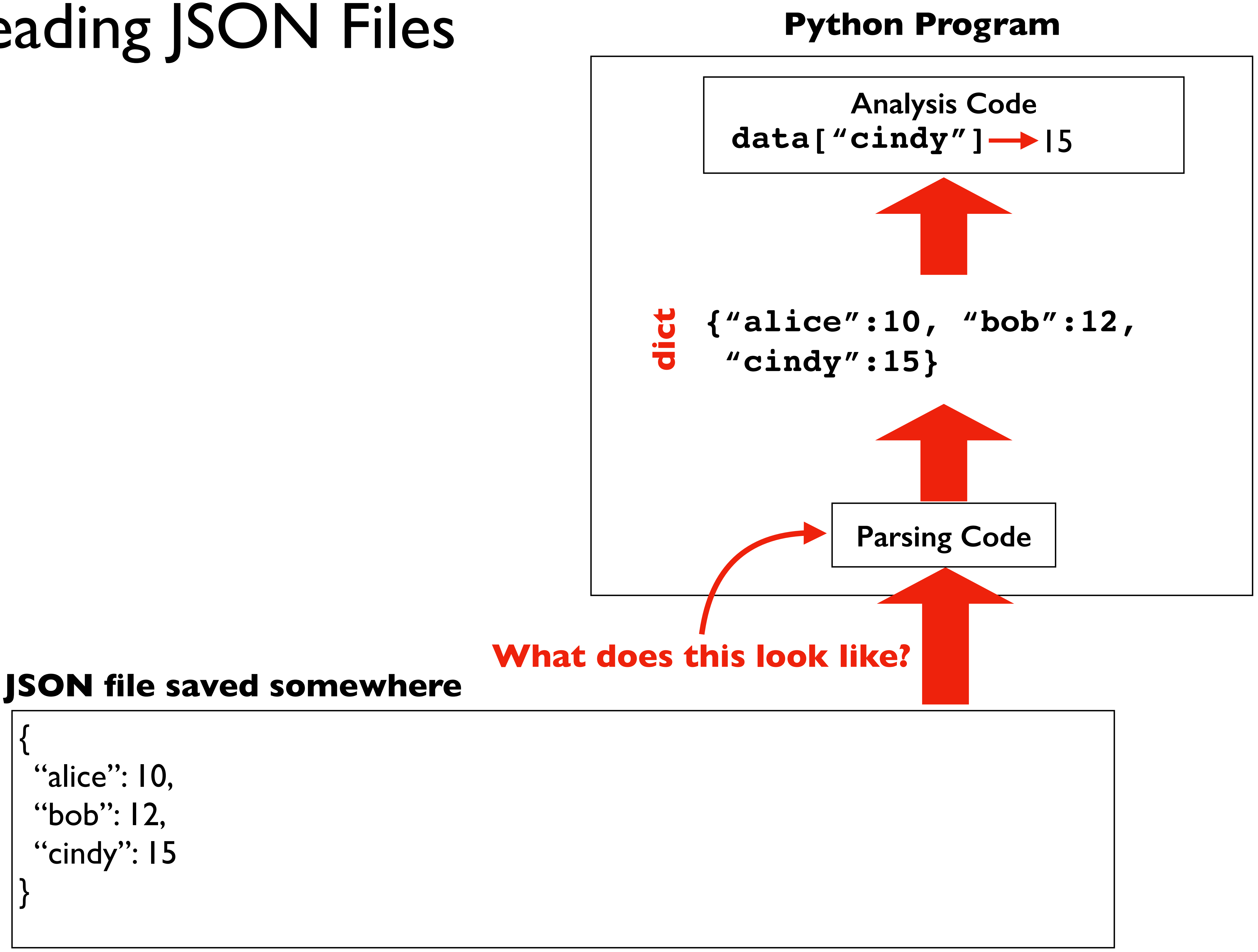
- JavaScript is a language for web development
- JSON was developed JavaScript programs to store/share data
- JSON looks like Python code because JavaScript is similar to Python

Minor JavaScript vs. Python differences:

	Python	**JSON**
Booleans	True, False	true, false
No value	None	null
Quotes	Single (‘) or double (“)	Only double (“)
Commas	Extra allowed: [1,2,]	No extra: [1,2]
Keys	Any type: {3:“three”}	Str only: {“3”:“three”}

remember these!

Reading JSON Files


Python Program
Analysis Code
data["cindy"] → 15
dict
{"alice":10, "bob":12, "cindy":15}
Parsing Code
What does this look like?
JSON file saved somewhere

```
{
  "alice": 10,
  "bob": 12,
  "cindy": 15
}
```

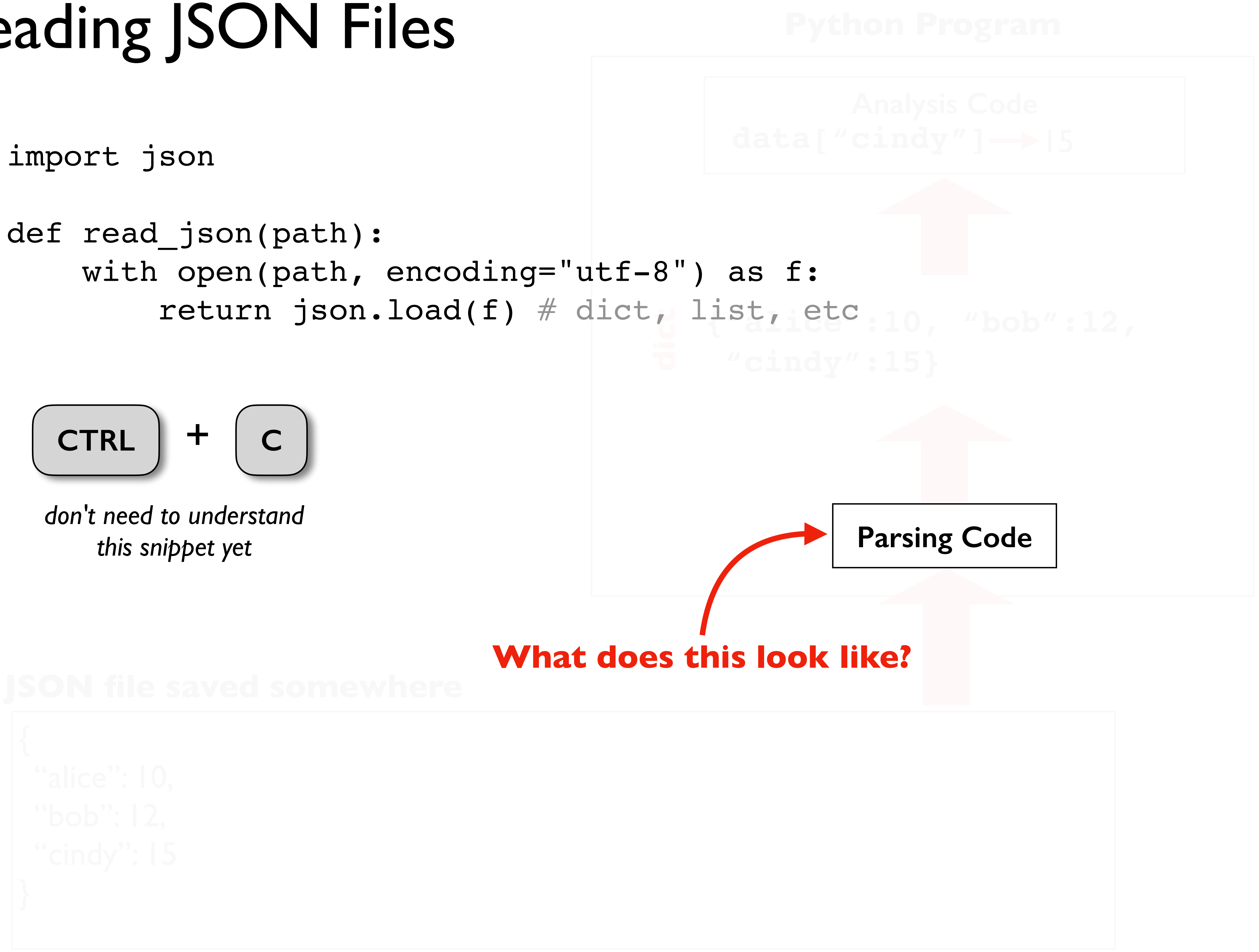
Reading JSON Files

```
import json

def read_json(path):
    with open(path, encoding="utf-8") as f:
        return json.load(f) # dict, list, etc
```

CTRL + C

don't need to understand this snippet yet



Reading JSON Files

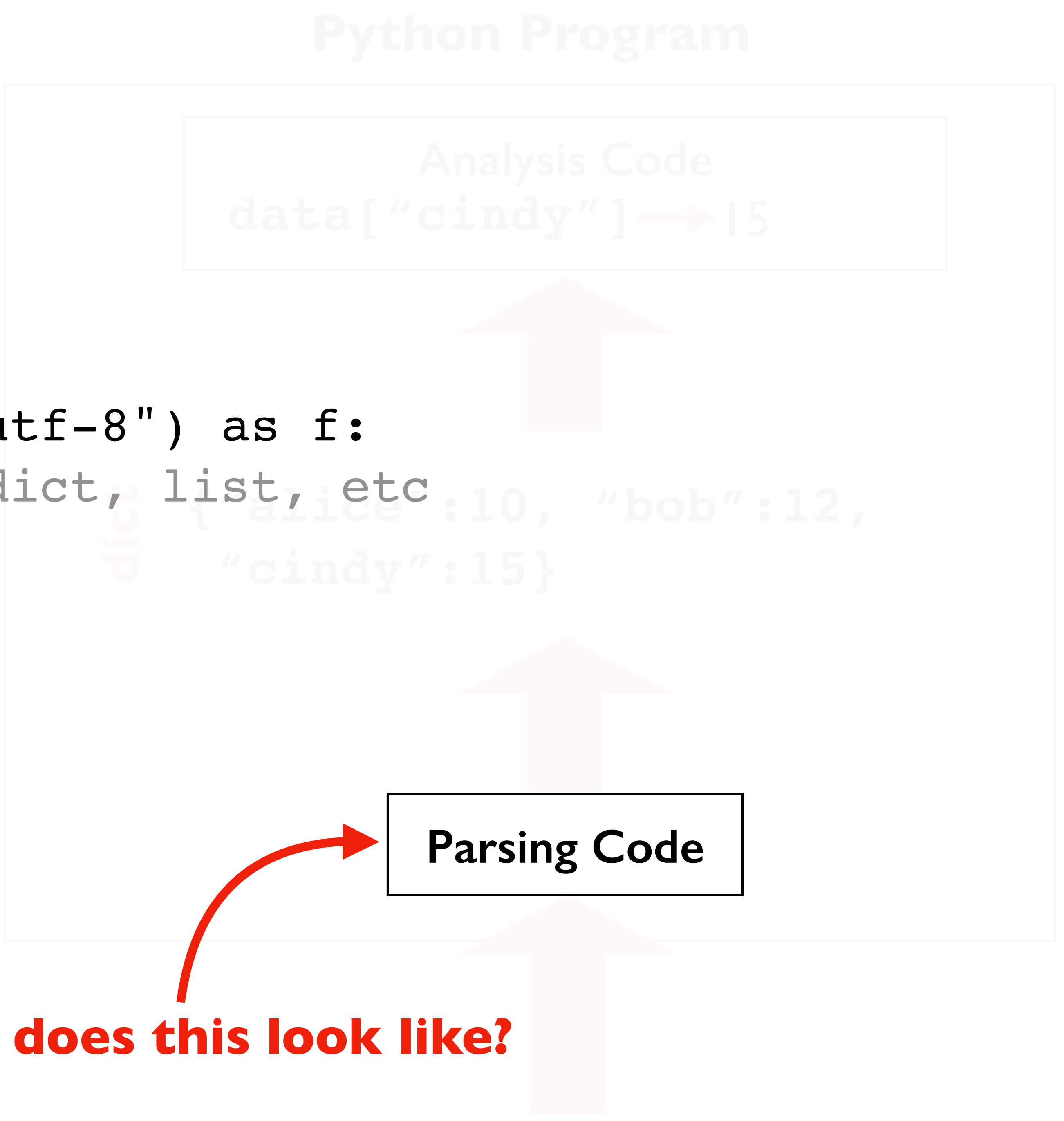
```
import json

def read_json(path):
    with open(path, encoding="utf-8") as f:
        return json.load(f) # dict, list, etc
```

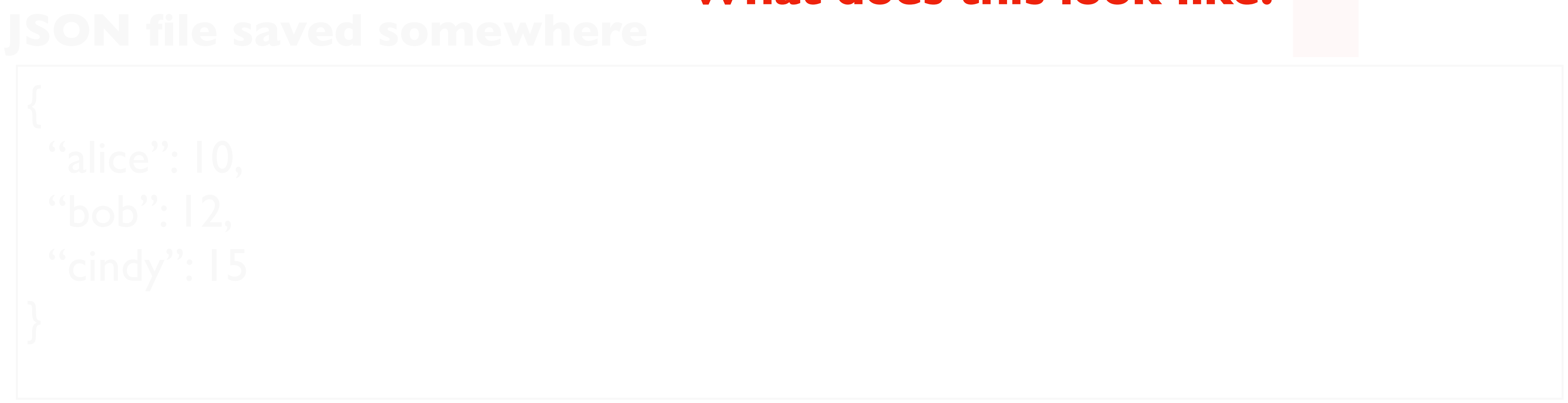
CTRL + C

don't need to understand this snippet yet

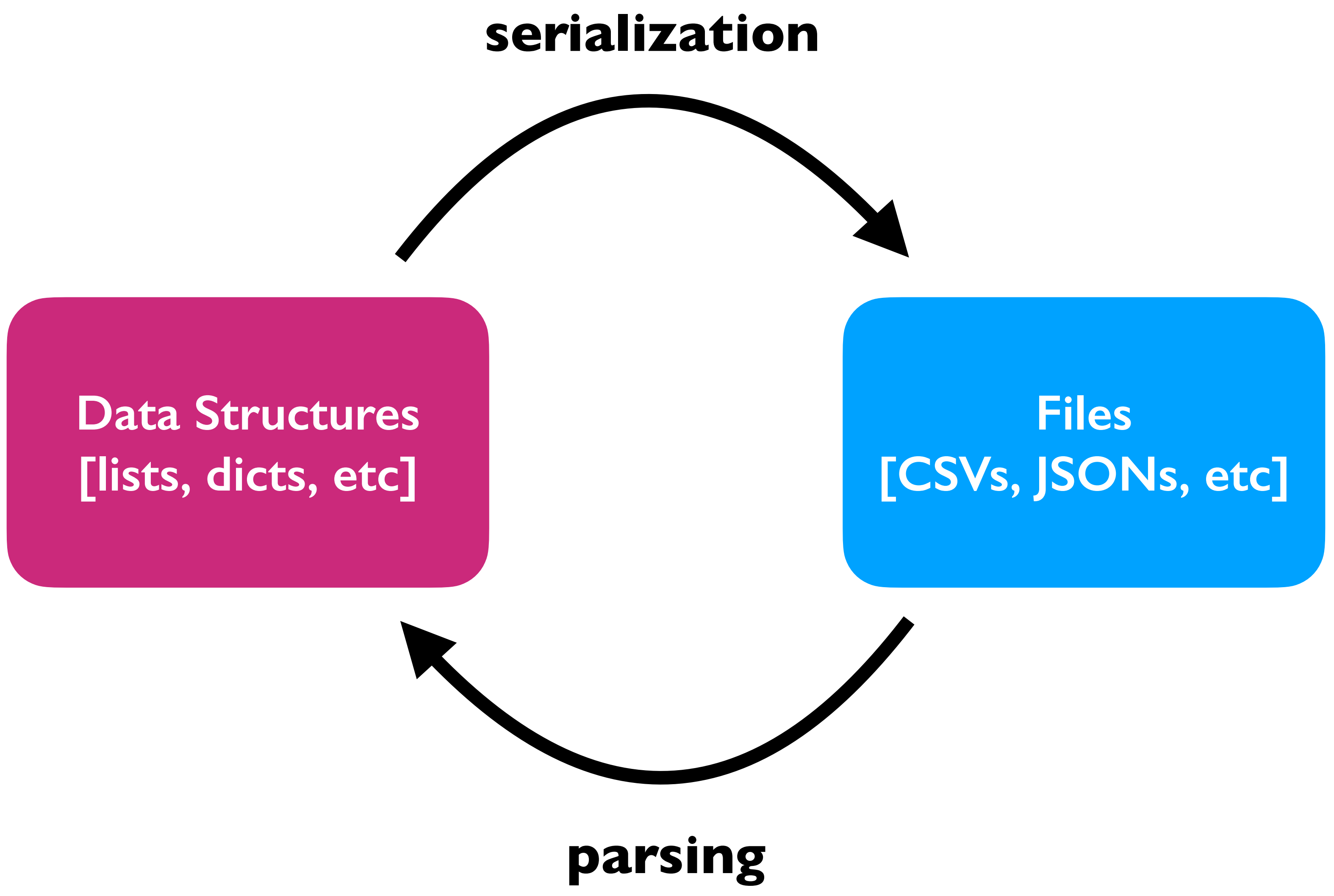
what about writing new files?



What does this look like?



Data Structures and Files



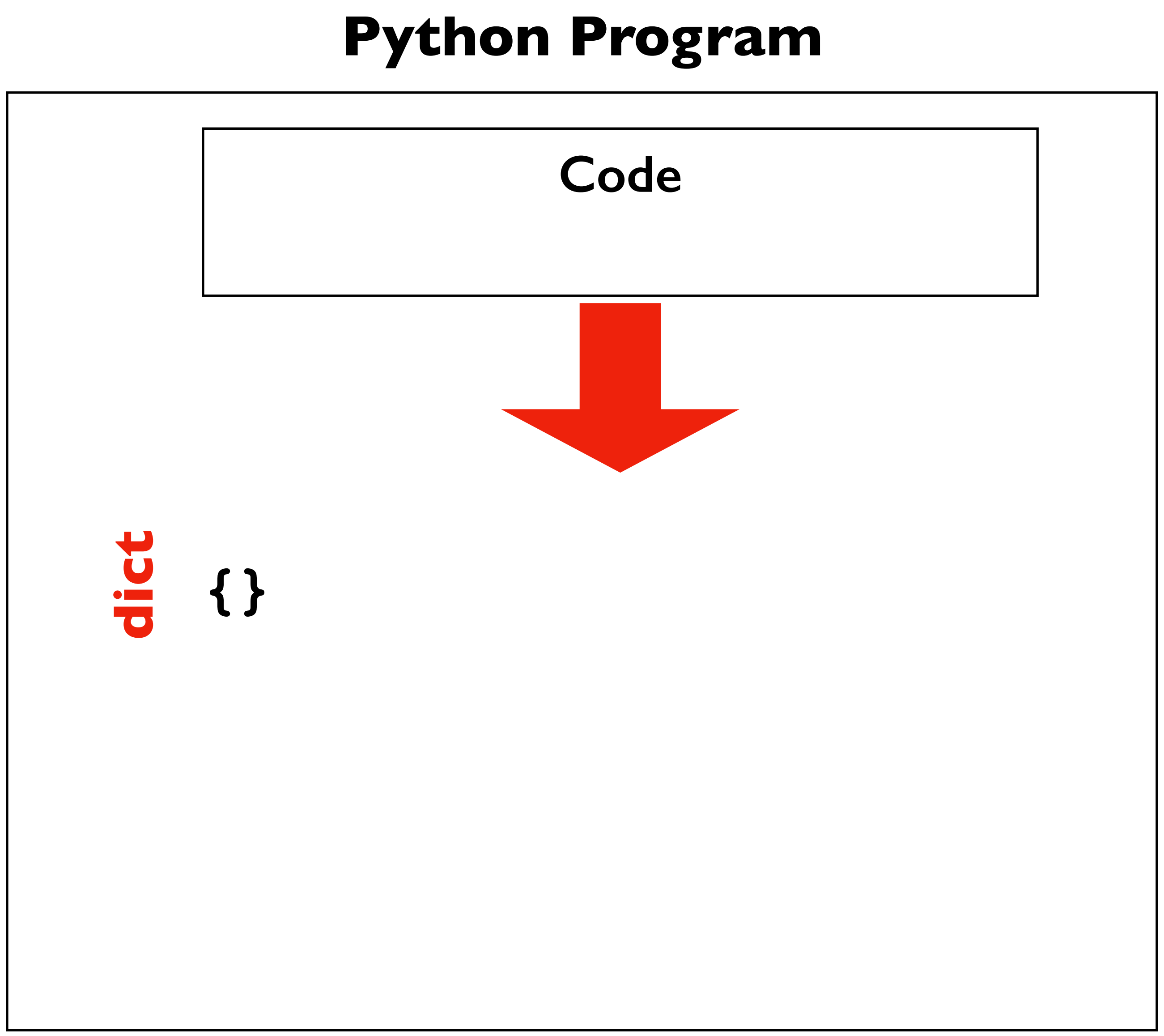
why not just have data structures?

because our data needs to live somewhere when our programs aren't running

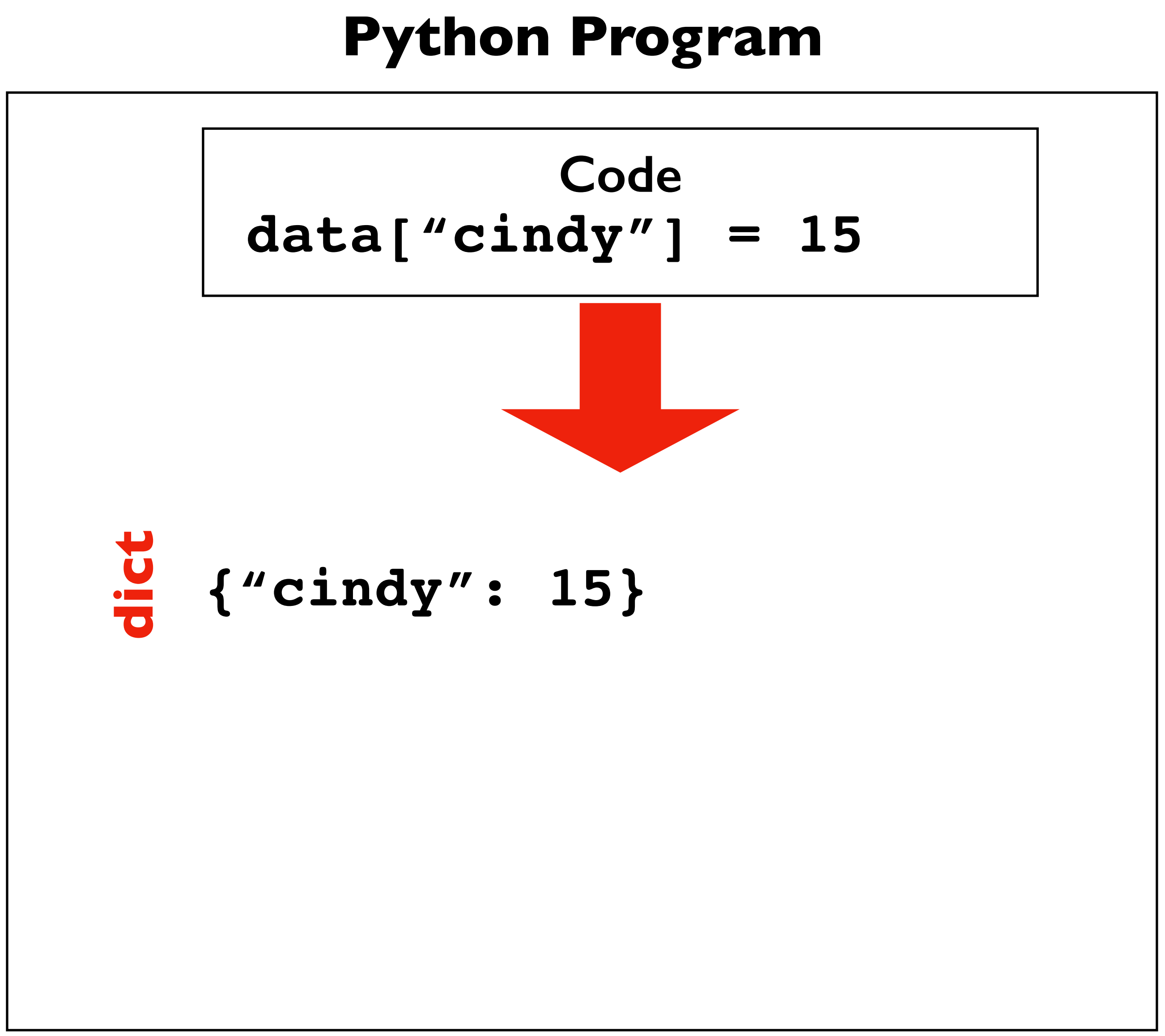
why not just have files?

slow, and Python doesn't understand structure until it is parsed

Writing JSON Files



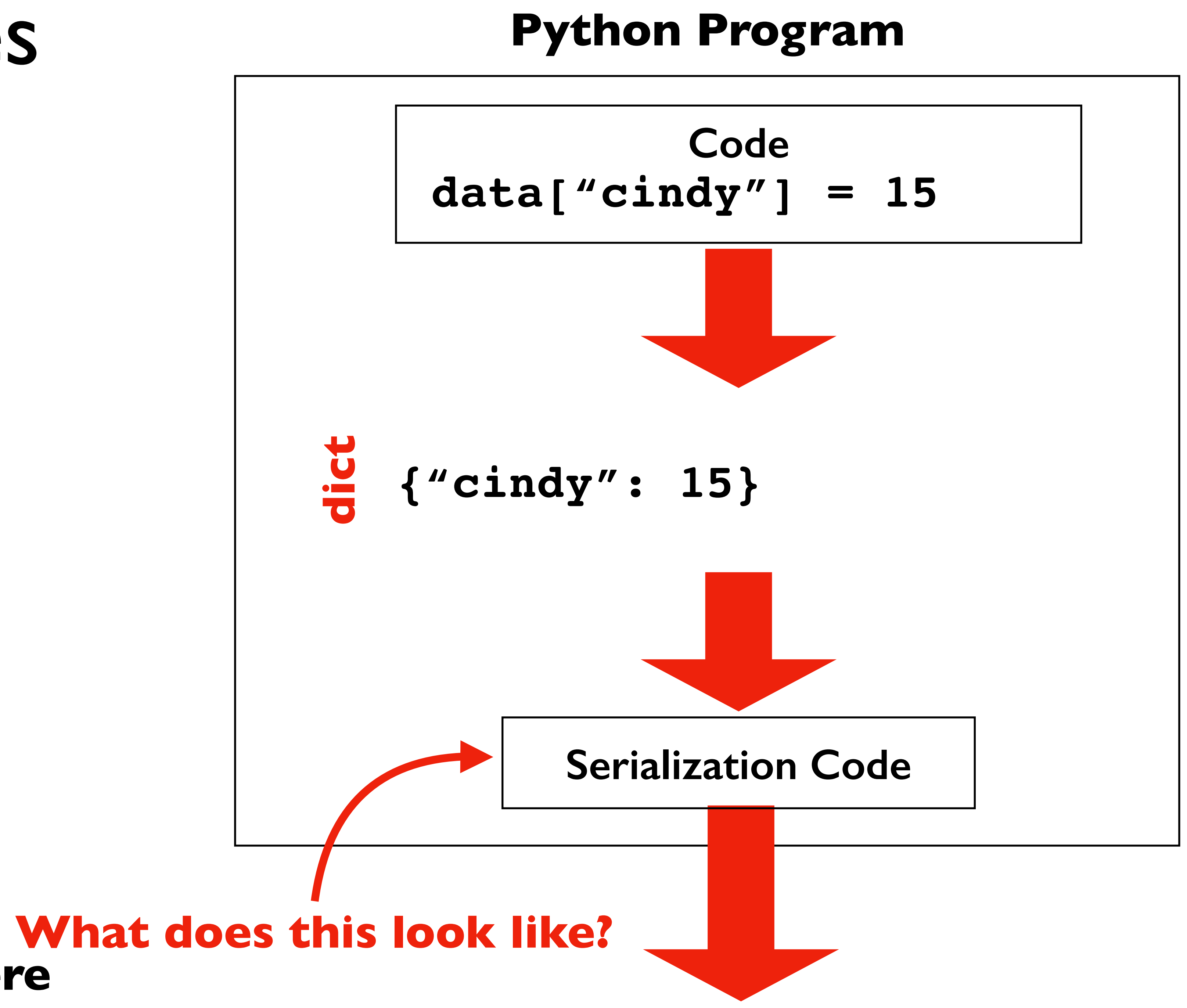
Writing JSON Files



```
data["cindy"] = 15
```

```
{"cindy": 15}
```

Writing JSON Files



JSON file saved somewhere

```
{
  "cindy": 15
}
```

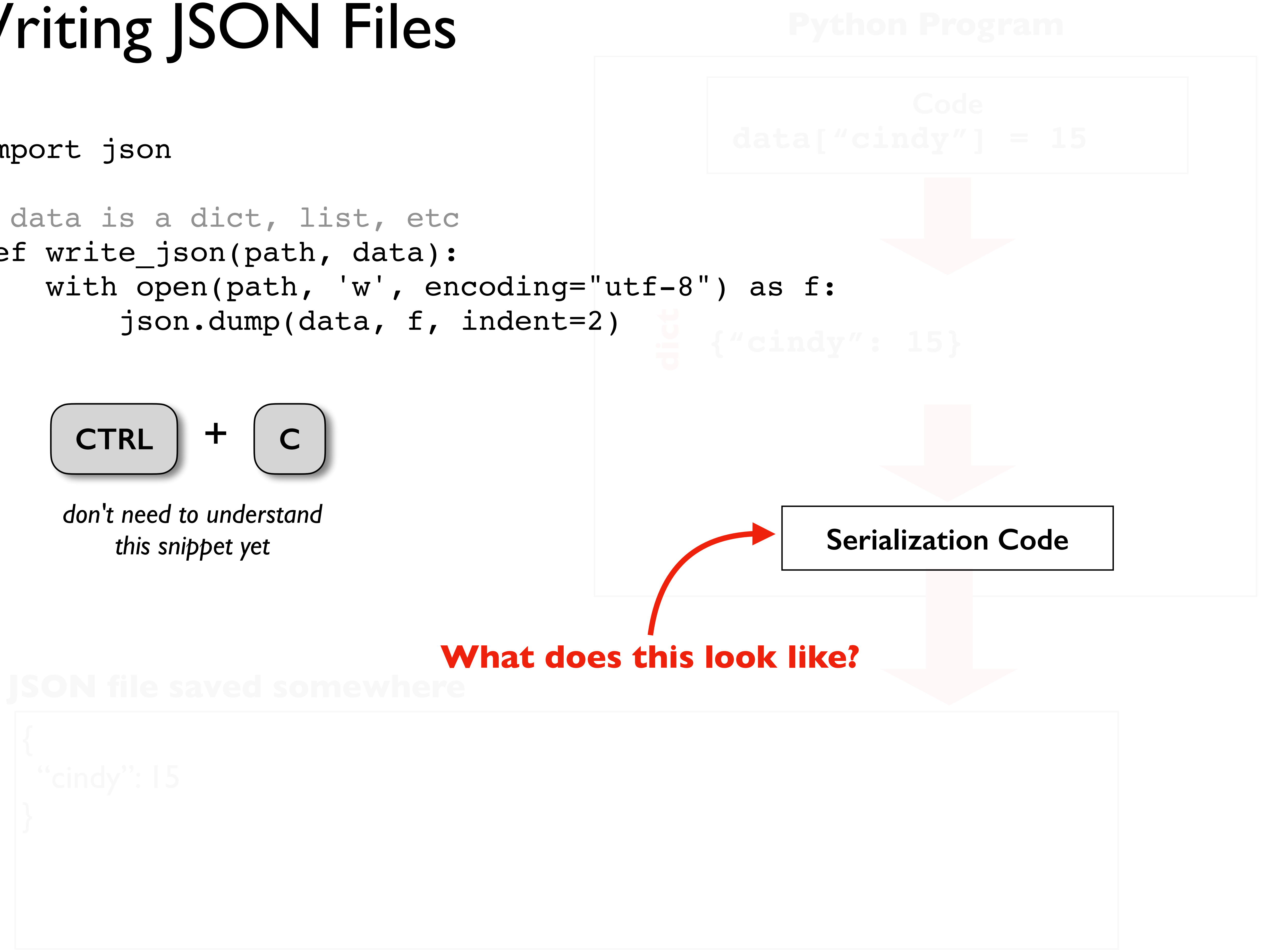
Writing JSON Files

```python
import json

# data is a dict, list, etc
def write_json(path, data):
    with open(path, 'w', encoding="utf-8") as f:
        json.dump(data, f, indent=2)
```

CTRL + C

don't need to understand this snippet yet



What does this look like?

Practice 1: Number Count

Goal: count the numbers in a list saved as a JSON file

Input:
- Location of the file

Output:
- The sum

Example:

```
prompt> python sum.py fileA.json
6
```

fileA.json

```
[1,2,3]
```

Practice 2: Score Tracker

Goal: record scores (save across runs) and print average

Input:

- A **name** and a **score** to record

Output:

- Running average for that person

Example:

```
prompt> python record.py alice 10
Alice Avg: 10
prompt> python record.py alice 20
Alice Avg: 15
prompt> python record.py bob 13
Bob Avg: 13
```