CIS 1100

Nested Data

Python
Fall 2024
University of Pennsylvania

JSON & XML

JSON:

- Javascript Object Notation
- It's basically just Python dictionaries that get printed out. Convenient!
- Use the `json` library to read it.

XML:

- Extensible Markup Language
- Sort of complicated tree structure of elements
- Use the `xml.etree.ElementTree` library to read it

Questions???

JSON

```
[
    {'name' : 'CIS1100',
     'section' : 1
     'days' : ['M', 'W', 'F'],
     'time' : '12:00pm',
     'instructors' : [
        {'name' : 'Harry', 'dept' : 'CIS', 'started' : 2020},
        {'name' : 'Jessica', 'dept' : 'CIS', 'started' : 2021}
     ]
    },
    {'name' : 'CIS1100',
     'section' : 2
     'days' : ['M', 'W', 'F'],
     'time' : '1:45pm',
     'instructors' : [
        {'name' : 'Harry', 'dept' : 'CIS', 'started' : 2020},
        {'name' : 'Travis', 'dept' : 'CIS', 'started' : 2021}
     ]
    },
]
```

(S7) How many courses are represented? If we parse this JSON into a dictionary `d`, can you write an expression that produces that value?

JSON

```
[
    {'name' : "CIS1100",
     'section' : 1
     'days' : ["M", "W", "F"],
     'time' : "12:00pm",
     'instructors' : [
        {'name' : "Harry", 'dept' : "CIS", 'started' : 2020},
        {'name' : "Jessica", 'dept' : "CIS", 'started' : 2022}
     ]
    },
    {'name' : "CIS1100",
     'section' : 2
     'days' : ["M", "W", "F"],
     'time' : '1:45pm',
     'instructors' : [
        {'name' : "Harry", 'dept' : "CIS", 'started' : 2020},
        {'name' : "Travis", 'dept' : "CIS", 'started' : 2022}
     ]
    },
]
```

(S8) What time does CIS 1100 001 meet? If we parse this JSON into a dictionary `d`, can you write an expression that produces that value?

Describing the Structure

```
[
    {'name' : "CIS1100",
     'section' : 1
     'days' : ["M", "W", "F"],
     'time' : "12:00pm",
     'instructors' : [
        {'name' : "Harry", 'dept' : "CIS", 'started' : 2020},
        {'name' : "Jessica", 'dept' : "CIS", 'started' : 2022}
     ]
    },
    {'name' : "CIS1100",
     'section' : 2
     'days' : ["M", "W", "F"],
     'time' : '1:45pm',
     'instructors' : [
        {'name' : "Harry", 'dept' : "CIS", 'started' : 2020},
        {'name' : "Travis", 'dept' : "CIS", 'started' : 2022}
     ]
    }
]
```

(L11) What keys do the upper level dictionaries have?

What keys do the lower level dictionaries have?

Complete the Program

(C12) Finish this snippet so that it prints out a set of every instructor's name.

- Don't assume you know how many courses there are
- Don't assume you know how many instructors each course has

```
d = json.load(json_string_of_courses) # dict representing prev. JSON
```

Some XML Terminology

- **Elements** are the entities being represented in the XML tree, e.g. an inventory or a price.
- **Tags** are the names that we give to the elements, e.g. `<inventory>` or `<price>`
- **Attributes** are properties that individual elements can have, stored in the tags
  - If the pop element is specifically a Pepsi, we could have its tag be `<pop brand="Pepsi">`.

```
<fruits>
    <berries>
        <fruit color="red">strawberry</fruit>
        <fruit color="blue">blueberry</fruit>
    </berries>
    <stonefruit>
        <fruit color="purple">plum</fruit>
        <fruit color="orange">peach</fruit>
    </stonefruit>
</fruits>
```

(S9) How many elements? What are the different tags? How many elements have attributes?

Some Tree Terminology

- The **tree** is the collection of elements being represented and the connections between them
- The **root** is the element of the tree that has no ancestors.
- An **ancestor** is an element that contains another element.
  - A **parent** is a direct ancestor.
- A **descendant** is an element that is contained by another element.
  - A **child** is a direct descendant.

```
<fruits>
    <berries>
        <fruit color="red">strawberry</fruit>
        <fruit color="blue">blueberry</fruit>
    </berries>
    <stonefruit>
        <fruit color="purple">plum</fruit>
        <fruit color="orange">peach</fruit>
    </stonefruit>
</fruits>
```

(S10) Which element is the root?

Which elements have no children?

Parsing & Traversing XML

Parsing XML:

```
import xml.etree.ElementTree as ET
tree = ET.parse('your_file.xml') # parse is used for reading a file
root = tree.getroot()                  # usually we work with the root directly
```

Iterating through children and printing the name of the

tag, the attribute dictionary, and the text of the children.:

```
for child in root:
    print(child.tag, child.attrib, child.text)
```

Describe the Structure

(L13) Describe the structure of the fruits XML

```
<fruits>
    <berries>
        <fruit color="red">strawberry</fruit>
        <fruit color="blue">blueberry</fruit>
    </berries>
    <stonefruit>
        <fruit color="purple">plum</fruit>
        <fruit color="orange">peach</fruit>
    </stonefruit>
</fruits>
```

Finish the Snippet

(C14) Finish the snippet to print out **just the names** of all the fruits inside of a given file `fruits.xml` that has the same structure as before. Don't assume that the file has the same number of fruits & categories. You can assume that the structure is the same.

```
import xml.etree.ElementTree as ET
tree = ET.parse('fruits.xml')
root = tree.getroot()

...
```