Accelerating Information Technology Innovation

http://aiti.mit.edu

Nigeria Summer 2012
Lecture 4 – Data Structures
Organizing the Football Universe

- Leagues -> Teams -> Many Players
- Teams and players can be represented by strings
- Build a data structure so that users can:
  - Check whether a team belongs to a league
  - Add and delete teams from leagues (promotion and relegation)
  - Track which players belong to which teams
Fundamentals
Lists

- *Ordered, mutable* collections: like a collection of numbered buckets!
- Can mutate, sort, and access different elements of lists
Lists: Initialization

- Initialize a list of player surnames:
  ```python
  barca=['valdes', 'alves', 'xavi', 'iniesta', 'messi']
  ```

- Access elements (individual player surnames) by index:
  ```python
  >> print barca[0]
  'valdes'
  ```
Lists: Iteration

• How can we print out all elements of the list, using a few lines of code?
  – Iteration over the items in the list
    
    ```python
    for player in barca:
        print player
    ```
  – Iteration over indices
    
    ```python
    for index in range(len(barca)):
        print barca[index]
    ```

• The simpler solution is usually better!
Lists: Operations

- Create new lists by ‘slicing’ existing lists:
  - Given: `example_list = [0,1,1,2,3,5]`
  - `first_three = example_list[:2]`
  - `last_four = example_list[2:]`

<table>
<thead>
<tr>
<th>example_list</th>
<th>first_three</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>last_four</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>5</td>
</tr>
</tbody>
</table>
Lists: Operations

- Example: Relegating teams from and promoting teams to the Premier League
- Promote the top two from FLChamp10 (list)
- Relegate the bottom two from Premier10 (list)
Lists: Operations

- Concatenating lists, we can assign Premier11
  - \( \text{Premier11} = \text{Premier10}[:5] + \text{FLChamp10}[:2] \)
Lists: Operations

• **Add:**
  – `barca.append(‘rossi’)` adds `rossi` to the end of the list
  – `barca.insert(‘rossi’, 0)` adds `rossi` at index 0 of the list (the beginning)

• **Remove:**
  – `barca.remove(‘messi’)` removes the first instance of `messi` from `barca`

• **Sort**
  – `barca.sort()` sorts all elements of the list in alphabetical order

• **Pop**
  – `barca.pop(k)` removes the kth element from the list and returns it.
Tuples: Introduction

- Essentially an **immutable** list
  - **CANNOT** change list items
  - Form: `tuple=('a', 'b', 'c', 'd',...)`

- **We saw an example of this earlier:**
  - `barca_tuple=('valdes', 'alves','xavi','iniesta','messi')`
Tuples: Manipulation

• **NOTICE:**
  - `tuple[0] = 'A'` returns an **error**

• There are **some** ways around this
  – Make new tuple and **add** part of existing tuple
  – `tuple = ('A',,) + tuple[1:]`
  – New Tuple: (`'A', 'b', 'c', 'd', 'e'`)
Lists and Tuples: Limitations

• Suppose ~1000 players in each professional football league

• How do we check that Messi is in the league? Are there any shortcuts?
  – Sorted lists can help
  – Costly to insert new elements into sorted lists

• A different solution: dictionaries, a common Python implementation of hash tables
Dictionaries

- An unordered collection of (key, value) pairs
- (key, value) pairs are mappings
  - key: something you know
  - value: something you want to know that is related to the key
- Key and value can be objects of any type

<table>
<thead>
<tr>
<th>Key</th>
<th>Value (multiple possibilities)</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>messi</code> (string)</td>
<td>‘LaLiga’ (string)</td>
</tr>
<tr>
<td></td>
<td>‘Argentina’ (string)</td>
</tr>
<tr>
<td></td>
<td>goals scored (int)</td>
</tr>
<tr>
<td></td>
<td>jersey number (int)</td>
</tr>
</tbody>
</table>
Dictionaries: Initialization

- Initialization (maps players to teams):
  
  \[
  \text{player\_team} = \{ \text{'messi': 'barca', 'donovan': 'galaxy', 'drogba': 'chelsea'} \}\]

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>messi</td>
<td>barca</td>
</tr>
<tr>
<td>donovan</td>
<td>galaxy</td>
</tr>
<tr>
<td>drogba</td>
<td>chelsea</td>
</tr>
</tbody>
</table>
Dictionaries: Modification

- Modification
  - Change Messi’s team:
    \[
    \text{player	extunderscore team[‘messi’]} = \text{‘real	extunderscore madrid’}
    \]

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>messi</td>
<td>real_madrid</td>
</tr>
<tr>
<td>donovan</td>
<td>galaxy</td>
</tr>
<tr>
<td>drogba</td>
<td>chelsea</td>
</tr>
</tbody>
</table>
Dictionaries: Modification

- Modification:
  
  Add a new player:
  
  ```python
  player_team[‘beckham’] = ‘who_knows’
  ```

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>messi</td>
<td>real_madrid</td>
</tr>
<tr>
<td>donovan</td>
<td>galaxy</td>
</tr>
<tr>
<td>drogba</td>
<td>chelsea</td>
</tr>
<tr>
<td>beckham</td>
<td>who_knows</td>
</tr>
</tbody>
</table>
Suppose someone gives you a list of players, `player_list`.

How can we use our dictionary, `player_team`, to print out the teams of each player on the `player_list`?

We may not know that `player_team` has an entry for an item in `player_list`.

```python
def check_list(player_list):
    player_league = {
        'messi': 'LaLiga',
        'donovan': 'MLS'
    }
    for item in player_list:
        if item in player_league:
            print(player_league[item])
        else:
            print('unknown league')
```

Later on: exception handling
Useful Questions

• Is the data I’m storing going to change?
  – Mutability VS Immutability
  – If NOT → Tuples!

• If data will change? Can it fit into a single list?
  – If YES → Use a List!
  – Recall it has: add, remove and sort methods
Useful Questions

• Will one set of data be mapped to another?
  – Words to definitions, soccer players to jersey sizes, students to grades
  – Dictionary!