http://www.cs.cornell.edu/courses/cs1110/2018sp

Lecture 14: Nested Lists, Tuples, and Dictionaries

(Sections 11.1-11.5, 12.1-12)

CS 1110

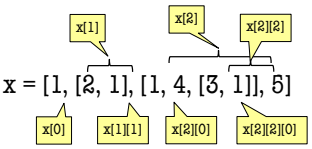
Introduction to Computing Using Python

[E. Andersen, A. Bracy, D. Gries, L. Lee, S. Marschner, C. Van Loan, W. White]

Nested Lists

- Lists can hold any objects
- Lists are objects
- Therefore lists can hold other lists!

```
b = [3, 1]
c = [1, 4, b]
a = [2, 1]
x = [1, a, c, 5]
```



4

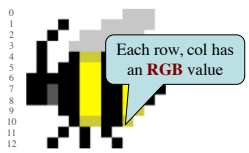
Two Dimensional Lists

Table of Data

	0	1	2	3
0	5	4	7	3
1	4	8	9	7
2	5	1	2	3
3	4	1	2	9
4	6	7	8	0

Each row, col has a value

Images



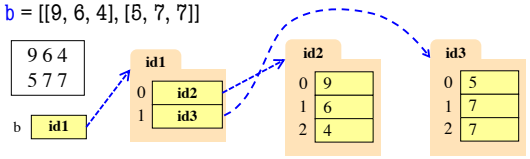
Store them as lists of lists (**row-major order**)

```
d = [[5,4,7,3],[4,8,9,7],[5,1,2,3],[4,1,2,9],[6,7,8,0]]
```

5

How Multidimensional Lists are Stored

- b = [[9, 6, 4], [5, 7, 7]]



- b holds **id** of a one-dimensional list
  - Has len(b) elements
- b[i] holds **id** of a one-dimensional list
  - Has len(b[i]) elements

7

Slices & Multidimensional Lists (Q1)

- Create a nested list
  ```
  >>> b = [[9,6],[4,5],[7,7]]
  ```
- Get a slice
  ```
  >>> x = b[:2]
  ```
- Append to a row of x
  ```
  >>> x[1].append(10)
  ```

- What is now in x?

A: [[9,6,10]]
B: [[9,6],[4,5,10]]
C: [[9,6],[4,5,10],[7,7]]
D: [[9,6],[4,10],[7,7]]
E: I don't know

10

Slices & Multidimensional Lists (Q2)

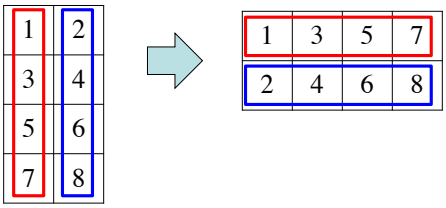
- Create a nested list
  ```
  >>> b = [[9,6],[4,5],[7,7]]
  ```
- Get a slice
  ```
  >>> x = b[:2]
  ```
- Append to a row of x
  ```
  >>> x[1].append(10)
  ```
- x now has nested list
  ```
  [[9, 6], [4, 5, 10]]
  ```

- What is now in b?

A: [[9,6],[4,5],[7,7]]
B: [[9,6],[4,5,10]]
C: [[9,6],[4,5,10],[7,7]]
D: [[9,6],[4,10],[7,7]]
E: I don't know

12

Data Wrangling: Transpose Idea



4 lists: 2 elements in each 2 lists: 4 elements in each

How to transpose?

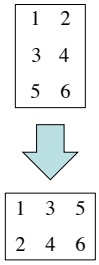
- 1st element of each list gets appended to 1st list
- 2nd element of each list gets appended to 2nd list

14

Data Wrangling: Transpose Code

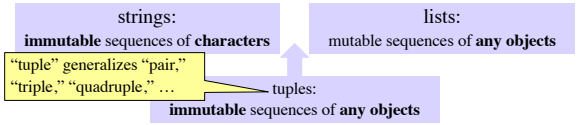
```
def transpose(orig_table):
    """Returns: copy of table with rows and columns swapped

    Precondition: table is a (non-ragged) 2d List"""
    numrows = len(orig_table)
    numcols  = len(orig_table[0]) # All rows have same no. cols
    new_table = [] # Result accumulator
    for m in list(range(numcols)):
        row = [] # Single row accumulator
        for n in range(numrows):
            row.append(old_table[n][m]) # Build up new row
        new_table.append(row) # Add new row to new table
    return new_table
```



15

Tuples



- Tuples fall between strings and lists
  - write them with just commas: 42, 4.0, 'x'
  - often enclosed in parentheses: (42, 4.0, 'x')

Conventionally use lists for:
- long sequences
- homogeneous sequences
- variable length sequences

Conventionally use tuples for:
- short sequences
- heterogeneous sequences
- fixed length sequences

16

Returning multiple values

- Can use lists/tuples to **return** multiple values

```
INCHES_PER_FOOT = 12

def to_feet_and_inches(height_in_inches):
    feet = height_in_inches // INCHES_PER_FOOT
    inches = height_in_inches % INCHES_PER_FOOT
    return (feet, inches)

all_inches = 68
ft,ins = to_feet_and_inches(all_inches)
print(You are "+str(ft)+" feet, "+str(ins)+" inches.")
```

17

Dictionaries (Type dict)

Description

- List of key-value pairs
  - Keys are unique
  - Values need not be
- Example: net-ids
  - net-ids are **unique** (a key)
  - names need not be (values)
  - js1 is John Smith (class '13)
  - js2 is John Smith (class '16)

Python Syntax

- Create with format: {k1:v1, k2:v2, ...}
- Keys must be **immutable**
  - ints, floats, bools, strings
  - **Not** lists or custom objects
- Values can be anything
- Example:

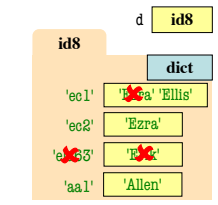
```
d = {'ec1':'Ezra Cornell',
     'ec2':'Ezra Cornell',
     'ela63':'Erik Andersen'}
```

18

Using Dictionaries (Type dict)

- Dictionaries are **mutable**
  - Can reassign values
  - d['ec1'] = 'Ellis'
  - Can add new keys
  - d['aa1'] = 'Allen'
  - Can delete keys
  - del d['ela63']

```
d = {'ec1':'Ezra','ec2':'Ezra',
     'aa1':'Allen'}
```



Deleting key deletes both

25