stuff is {'color': 'black', 1: 2, 'cat': 100, (1, 1): 'yes', 1.5: 3}
N is for ...

• **Nested Loops**
  • All pairs, all pixels, all 2D structures

• **None**
  • Default value for functions if no return

• **Newline**
  • The "\n" in a line
The 21 Most Important Googlers You've Never Heard Of

Georges Harik and Noam Shazeer created the underlying data that led to AdSense

Harik and Shazeer spent years analyzing data on webpages, trying to understand clusters of words and how they worked together. The data they gather wound up being used by Google for its AdSense product, which analyzed webpages for words, and then stuck ads on them.
Announcements

• Assign 3 Transform due Today!
• Assign 4 is out, due Thursday, Nov 3
• APT 4 due Thursday, Oct 27
• Lab 6 tomorrow, do prelab 6 before going
  • Videos of Labs 4 and 5 in Sakai Resources folder
• Mid-Semester Survey complete by 10/23
  • Extra credit for Exam 2
• Do not discuss APT Quiz 1 with anyone until they are handed back
• Exam 2 Nov 1
PFTD

• Guess Word
• Dictionaries
• Solving Problems with Dictionaries
• Solving an APT
Assignment 4: Guess Word

• We give you most of the functions to implement
  • Partially for testing, partially for guiding you
• But still more open ended than prior assignments
• If the doc does not tell you what to do:
  • Your chance to decide on your own!
    • Okay to get it wrong on the first try
  • Discuss with TAs and friends, brainstorm!
• Demo!
Problem: Given a name, what is their favorite ice cream?

• Assume you have a lot of people, over 1 million.

• How is the data stored?

• Assume we have parallel lists
  • students is list of names
  • icecream is list of corresponding favorite ice cream
Code might be

1  if name in students:
2       pos = students.index(name)  # find position of name
3       answer = icecream[pos]     # answer in same pos

If a billion names, this is not efficient
How does this code work?
How does search with .index work?

- **Parallel Lists**
  - Search for name first in students list
  - Use index location of name to find favorite ice cream

<table>
<thead>
<tr>
<th>students</th>
<th>icecream</th>
</tr>
</thead>
<tbody>
<tr>
<td>`['Astrachan',</td>
<td>`['Chocolate',</td>
</tr>
<tr>
<td>'Sun',</td>
<td>'Chocolate Chip',</td>
</tr>
<tr>
<td>'Rodger',</td>
<td>'Chocolate Chip',</td>
</tr>
<tr>
<td>'Forbes']</td>
<td>'Strawberry']</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Compsci 101, Fall 2022 10/20/22
Code was easy

• But for a lot of data could take a long time.

• Let’s see another way, dictionaries
How the Dictionary is made

• Using a dictionary is reasonably straight-forward
  • We will be clients, not implementers
  • Efficiency not a large concern in 101
  • Our goal is to just get stuff done 😊
What is a Dictionary?

• A collection of (key, value) pairs (abstract view)
  • Look up key, find the value

• For list
  • `a[3]` takes same time as `a[3000]`
  • Finding the item is slow
  • Fast once you know the index

• For Dictionary: `d["cake"]`
  • Finding the value associated with "cake"
  • very, very fast
Dictionaries/Maps

• Dictionaries are another way of organizing data
• Dictionaries are sometimes called maps
• Keys and Values
  • Each key maps to a value
  • Some keys can map to the same value
  • Can change the value a key maps to
Example

- Each student could be mapped to their favorite ice cream flavor
How is dictionary different than a list?

- List – have to search for name first
- Dictionary – each key maps to a value
- getting name (or key) is automatic! Fast!
Implementing a Dictionary/Map

Keys map to values

• Create Empty dictionary
  somemap = {}

• Put in a key and its value
  somemap[“Forbes’] = “Strawberry”

• Get a value for a dictionary
  value = somemap[“Forbes”]

• Change a value for a dictionary
  somemap[“Forbes’] = “Chocolate”
Change Astrachan’s value
somemap[“Astrachan”] = Coffee Mocha

Students

Astrachan
Sun
Rodger
Forbes

Ice Cream Flavors

Chocolate
Chocolate Chip
Strawberry
Value could be a set or list

- **Students**
  - Astrachan
  - Sun
  - Rodger
  - Forbes

- **Ice Cream Flavors**
  - Coffee Mocha
  - Chocolate
  - Vanilla
  - Blueberry
  - Chocolate Chip
  - Blueberry
  - Banana
  - Strawberry
  - Coffee Mocha
How to use a Dictionary

• Create: `d = {}`
  • `d = {'a': 10, 'b': 100}`
  • `d = dict([('a', 10), ('b', 100)])`
• Insert: `d[KEY] = VALUE`
• Update/Reassign: `d[KEY] = VALUE`
• Get a value (like list indexing): `d[KEY]`
• *Key* membership (not values): `KEY in d`
  • No membership check for values
Examples

```
stuff={}  
print(stuff) 
print(type(stuff)) 
stuff['color'] = 'black' 
stuff[1] = 2 
stuff['cat'] = 100 
stuff[(1,1)] = 'yes' 
stuff[1.5] = 3 
print(stuff)
```
Examples

```
stuff is {'color': 'black', 1: 2, 'cat': 100, (1, 1): 'yes', 1.5: 3}
```

```
print(len(stuff))
stuff[3] = [6, 3, 2]
```

```
stuff[[4,7]] = 'go'
```
Examples

d={}

d['color'] = 'black'

d['color'] = 'red'

d['red'] = 'color'

r = d[d['red']]

r = d['monkey']
Examples

d = {'a':'cat', 'e':'dog'}

'dog' in d
'a' in d
'pig' in d
WOTO-1 Dictionaries

More on Dictionary

• Like lists, but with keys
• KEY – immutable type, unique within dictionary
• VALUE – any type, not unique within dictionary
• Dictionary is unordered collection of (KEY, VALUE) pairs
More on using a Dictionary/Map

• Get all the keys (as a list)
  • listKeys = somemap.keys()

• Get all the values (as a list)
  • listValues = somemap.values()

• Other methods
  • clear – empty dictionary
  • items – return (key,value) pairs
  • update – update with another dictionary
Examples

d = {'a':4, 'e': 3, 'b':4 }

v = d.values()
k = d.keys()
p = d.items()

for t in d.items():
    print(t)
Problem

• Given a list of names of people who ate at a restaurant, who ate there the most?
• A name appears more than once if they ate their more than once

• names = ['Sarah', 'Beth', 'Sarah', 'Purnima', 'Beth', 'Beth', 'Purnima']
WOTO-2 Problem Solving
APT: EatingGood

Problem Statement

We want to know how many different people have eaten at a restaurant this past week. The parameter meals has strings in the format "name:restaurant" for a period of time. Sometimes a person eats at the same restaurant often.

Return the number of different people who have eaten at the eating establishment specified by parameter restaurant.

For example, "John Doe:Moes" shows that John Doe ate one meal at Moes.

Write function howMany that given meals, a list of strings in the format above indicating where each person ate a meal, and restaurant, the name of a restaurant, returns the number of people that ate at least one meal at that restaurant.

Specification

```python
filename: EatingGood.py
def howMany(meals, restaurant):
    ""
    Parameter meals a list of strings with each in the format "name:place-ate". Parameter restaurant is a string
    return # unique name values where place-ate == restaurant
    ""
    # you write code here
    return 0
```

APT Eating Good Example

meals = ["Sue:Elmos", "Sue:Elmos", "Sue:Elmos"]

restaurant = "Elmos"

returns 1
WOTO-3: APT Eating Good

- https://www2.cs.duke.edu/csed/pythonapt/eatinggood.html

(1) Work an instance yourself

(2) Write down exactly what you just did

(3) Generalize your steps from (2)

(4) Test your steps

Identified problem

Can’t find pattern

Algorithmic
APT Eating Code Idea

- Make an empty list
- Loop over each meal
  - Split the meal into person and restaurant
  - If the restaurant matches
    - If person not already in list
      - Add person to the list
  - Return the length of the list