Week: 06 Date: 2/24/2022

15-110 Recitation Week 6

Reminders

- HW3 due Monday 2/28 at Noon
- How was Quiz 2?
- How were code reviews?

Overview

- Recursive Code Tracing
- Dictionary Review
- Hashing
- Dictionary: Code Writing

Problems

RECURSIVE CODE TRACING

Consider this recursive function:

```
def f(a, b):
      if a == []:
            return []
      else:
          return [a[0]] + [b[0]] + f(a[1:], b[1:])
```

If we call the function with these values:

```
print(f([1,2,3], [4,5,6]))
```

Trace through the code to determine what will be printed.

Tracing:

DICTIONARY REVIEW

Notes on dictionaries:

Here is an example of a type of problem that uses dictionaries. Read through the problem statement and solution and note the key points of the code.

Problem:

Kelly's Bakery is doing an inventory of their freshly baked goods. This morning, they baked new items and now they need to update their inventory to represent these items. You are given a dictionary that represents the inventory at Kelly's Bakery, which maps the name of the item to how many items of that baked good are available. Write the function updateInventory(d, newItems) that takes the current inventory and a new dictionary called newItems and updates it accordingly. The function should also handle the case that there is an item in newItems that doesn't exist in d.

Solution:

```
def updateInventory(d, newItems):
  for item in newItems:
     if item in d:
        d[item] += newItems[item]
     else:
        d[item] = newItems[item]
  return d
```

HASHING

Neeraj wants to keep track of what fruits his friends like, and he decides a dictionary would be the best way to do this. He has collected data from his friends and has run these lines of code to initialize the dictionary using the hash function below. The corresponding hash table of length 5 is also shown below:

Note: Remember we do hash(value) % size of table if the hash is larger than the size of the table

```
def hash(s):
    return len(s)

d = {}
d["pineapple"] = 10
d["pears"] = 7
d["bananas"] = 14
```

“pears” : 7		“bananas” : 14		“pineapple” “: 10
Bucket 0	Bucket 1	Bucket 2	Bucket 3	Bucket 4

1) Neeraj realizes he forgot to account for the 50 of his friends that love figs.
    a) What line of code should he use to add this to the dictionary?

    b) Where will it be added in the hash table?

2) What makes this a bad hash function, and how can we make this hash function better?

3) Neeraj comes back to this dictionary later and wants to check if any of his friends like apples.
    a) What line of code should Neeraj run?

    b) How do we check if an element is in the hashtable?
    In general:

For “apples” specifically:

4) What would happen if Neeraj tried to run these lines of code?

```
print(d["bananas"])

print(d[“pomegranate”])
```

5) Describe one concrete situation in which a hash table would not be an appropriate choice of data structure. Explain why a hash table would not be appropriate in that situation.

DICTIONARY CODE WRITING

Given a list of wins by CMU, Pitt, OSU, PennState's, and another unspecified number of football teams, return the team with the most wins. There will be no ties. For example,

mostWins(["CMU", "Pitt", "OSU", "OSU", "PennState", "OSU"]) returns "OSU".

We have provided the general form of the function, fill in the blanks with the code.

```
def mostWins(L):
    # Initialize an empty dictionary
    wins = dict()
    # Loop through the input list
    for _____________ in _____________:
        team = L[i]
        # What do we do if the team is in our dictionary?
        if team in wins:
           wins[team] = _____________
        # Otherwise?
        else:
           wins[team] = _____________
    # Initialize variables to store the team that has won
    # the most so far and how many times they had won
    mostWinTeam = _____________
    mostWins = _____________
    # Loop through the dictionary
    for _____________ in _____________:
       # What do we do if the current team has won more
       # than the team with the most wins so far?
       if _____________ > mostWins:
          mostWins = _____________
          mostWinTeam = _____________
    # Return the team that has won the most
    return mostWinTeam
```