Programming assignment for candidates applying to join the Apple Siri Team

Introduction

This assignment is designed to test your general programming capability including in particular:

a. ability to design reliable, scalable and efficient systems using appropriate data structures and algorithms
b. ability to write code which is easy to understand by others and easy to maintain
c. ability to structure your code so that it is reusable, modular and testable

You are free to use any programming language that you wish. However, note that we do expect to be able to run and test your submitted code so make sure that your choice is readily available on Mac OS X. Note also that whilst it is fine to use standard libraries (eg C++ STL) we do want to see your implementation of the core data structure. We suggest you budget 3-4 hours for this assignment, and should you need more please tell us why. Please clearly indicate the actual amount of time spent when you return your solution.

Problem Description

When a user interacts with Siri we must produce a valid and reasonable response, and the first step to achieving this is to understand and interpret what the user has just said. One aspect of this problem is the need to map spoken phrases to potential concepts within that utterance. This programming assignment is therefore to write code which receives as input a string that is the sentence spoken by the user and returns any and all matches from a given list of concepts.

You can assume that there will be at most 20 words in the input string and that the words are separated by one or more whitespace characters. Any punctuation in the input should be treated as not significant. *Note that the list of concepts to check against could run into the millions.*

We have provided a sample concept list and few example inputs at the end of this document for you to use in testing your solution. Feel free to extend testing beyond the samples provided.

What to Submit

Please return your solution in the form of an archive (zip, tar etc.). When submitting please return all your code (including any automated tests you might have developed) along with documentation you feel might be appropriate. It should be easy for us to compile, run and verify the correctness of your solution.

There are lots of complications outside the scope of this exercise so please document any important assumptions you have made. Try to estimate the computational complexity of your solution and indicate this in your documentation. You should also briefly highlight any aspects that could affect accuracy, speed, scalability or reliability of your solution should it be taken into a live production environment. Also briefly describe any ideas you might have on how to address such aspects.

Sample concept list

- Indian
- Thai
- Sushi
- Caribbean
- Italian
- West Indian
- Pub
- East Asian
- BBQ
- Chinese
- Portuguese
- Spanish
- French
- East European

Using this sample concept list, example string inputs and the corresponding matches might be:

Input “I would like some thai food”,	Matches “Thai”
Input “Where can I find good sushi”,	Matches “Sushi”
Input “Find me a place that does tapas”	Matches none
Input “Which restaurants do East Asian food”	Matches “East Asian”
Input “Which restaurants do West Indian food”	Matches “West Indian”, “Indian”
Input “What is the weather like today”	Matches none

SJY Revised June 17