CS214: Project

Contact Manager – Spelling Checker

(10 marks + 3 Bonus)

Cairo University, Faculty of Computers and Information

"ومن يتق الله يجعل له مخرجا ويرزقه من حيث لا يحتسب"

Dear student, we like to thank you very much for your effort on this course. We wish you a pleasant learning experience. In shaa Allah, years later in this career, you remember this course and its easy times and its hard times and smile ☺. Take the time to (1) help those who need help and are having difficult time with their study and (2) thank those who taught you or work at FCI to run the educational process.

Here we present to you the course project that will help you deeply understand, implement and use a few very useful data structures.

Deadline

The deadline for submitting the electronic solutions:

1. **20 April** 2013 @ 10 pm *Initial Version of Task 1* [-2 marks if not submitted on time]
2. **1 May** 2013 @ 10 pm *Initial version of Task 2* [-2 marks if not submitted on time]
3. **12 May** 2013 *Final Version Final Version of Task 1 & Task 2 & Task 3*

Task 1 [5 marks] Task 2 [5 marks] Task 3 [3 marks][optional]

Note: Initial version should be: 1- Complete data structure built 2- But the functionality may be incomplete (some special cases not handled etc.). When submitting the initial version you must write what is done and what is remaining in the task.

Objectives:

- Designing and implementing data structures.
- Applying different types of data structures in real world problems.
- Thinking creatively to solve problems using Data structures.

Details:

1. **Project will be solved in groups of three students from any group.**
2. There are two options for the project: Contact Manager or Spelling Checker. Each team should choose one project. **Each project will take maximum of 65 teams.**
3. **Each team must register their project by 9 April on this form. Registration will close after that date.** http://bit.ly/Yv1rXM
4. Each project consists of 3 tasks: Two tasks are essential and one is optional.
5. **Team members have collective responsibility to plan the project and work together and** support each other and help each other to understand concepts and fix code**.**
6. **Team leader is responsible of uploading on acadox.**
7. **Discussions will be done based on schedule that will be announced, between 14-17 May.**
8. **You may NOT use any built in data structures or algorithms in C++ in this project. YOU MUST BUILD all your data structures by yourself from scratch and follow the code templates we send to you.**

CS214: Project

Contact Manager – Spelling Checker

(10 marks + 3 Bonus)

Cairo University, Faculty of Computers and Information

9. Each team will write a report explaining:
   a. The design of their solution for each task (classes and their relations, data structure layout and drawing, algorithms, etc.)
   b. The strengths and weaknesses of their work in each task.
   c. Comparison of enhancement of complexity at each task from the previous one(s).
   d. With the submission of each task of the project, the team will submit the relevant part of the report in pdf format.

أكتب على غلاف التقرير المقدم
الله يرانى فى كل مكان ، أنا أمين و لا أغش
أنا أحصل على الدرجات بتعبى و شقايا و عرق جبينى

10. Each student will be marked as follows:
    a. The quality of the solution
    b. The collaboration of the team and communication with each other.
    c. The contribution of the student to the project and the quality of the parts he did.
    d. Ability of the student to present the project and to explain each point in the project.
    e. Understanding of the data structure design and the complexity of their operations.

CS214: Project

Contact Manager – Spelling Checker

(10 marks + 3 Bonus)

Cairo University, Faculty of Computers and Information

Project 1: Contact Manager

Introduction

You and your friends are building a Smartphone Operating System (choose its name). You are going to compete with iOS, Android, Blackberry and Windows phone. Competition is very hard, and you need to build strong features. One of the most critical parts of the Smartphone OS is the contact manager, and they decided you are the one to develop it. Are you up to the task?

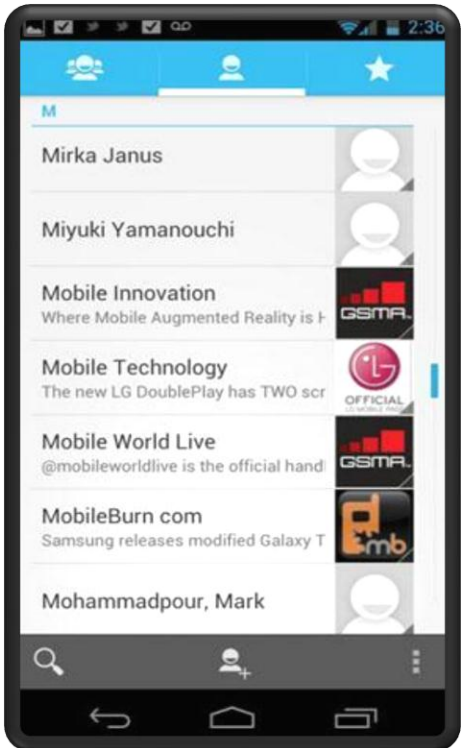

Figure 1: Android ICS Contact Manager

Problem Statement

You are required to implement the contact manager for a smartphone. This is a program that keeps track of all your contacts' information (phone number, email ..) as well as keeping a call log (the recently sent and received calls). You can also quickly reach a contact through search.

Task 1

You are given a file with all your contacts. This file is called "all-contacts.in". Each line of the file will be in the following format (without quotes):

```
"phone number" "contact name"
```

Phone numbers contain only digits (**no** + or -) and can start with 0 (e.g. 010...). Contact names are in English (lower-case or upper-case) and **may** contain spaces. A contact name **may** appear more than once in the file, which means this person has multiple phone numbers.

You are required to build a contacts list using an ordered “linked list”, with all the contacts **sorted** by contact name **alphabetically**. Each contact is an object, carrying the name of the contact, and carries all phone numbers as a “linked list”. All phone numbers in the linked list must be **sorted** as well. A phone number will only appear **once** in the file. The linked list should be sorted at all times during the execution (always add elements in sorted order).

After building this data structure, print its contents in the format given below.

CS214: Project

Contact Manager – Spelling Checker

(10 marks + 3 Bonus)

Cairo University, Faculty of Computers and Information

Sample "all-contacts.in" file:

002011111 Sayed Hassan
002022222 Mohamed Hesham
002054321 Mina Zaki
002012345 Mina Zaki
002098765 Bob

Sample output:

Bob: 002098765
Mina Zaki: 002012345 - 002054321
Mohamed Hesham: 002022222
Sayed Hassan: 002011111

Task 2

Once you have prepared the contact list, the call log keeps track of the numbers you recently contacted. Given a list of recently contacted phone numbers, you are required to do this:

1- If the number belongs to a contact, display the contact name.

2- Otherwise display the number unchanged.

The Linked List used in Level 1 is inefficient for this task. We need a better data structure to efficiently search for a phone number and get the name. We will use a “Balanced Binary Search Tree” because it allows us to search in O(log *n*) while keeping the phone list dynamic. Each node in the tree will carry a phone number and a pointer to the contact. For this task you are required to implement a "**Treap**". It is a Randomized Balanced Binary Search Tree that performs very well in practice. Read the document attached parts 8.1 to 8.4 to understand more about them.

We won’t need all of the functionality of the tree for this problem. The required operations are:

- Add (phone number, pointer to contact): adds a phone number and pointer to contact in its right position in the tree.
- Find (phone number): Searches for a phone number in the tree. If found, returns the contact, otherwise returns null.

Sample input:

002098765
002054321
002054321
012521212
002011111

Sample output:

Bob
Mina Zaki
Mina Zaki
012521212
Sayed Hassan

CS214: Project

Contact Manager – Spelling Checker

(10 marks + 3 Bonus)

Cairo University, Faculty of Computers and Information

Task 3 (Bonus):

As a final feature in the contact manager, we need to help users find contacts quickly. **Smart Dial™** is exactly what we need. When a person begins typing a number, we start matching not only numbers, but also names. Each number has characters associated with it (e.g. 2 has "ABC", 6 has "MNO"). So if the user typed 26, it could mean any combination of the letters of these letters ("AM", "AN", "AO", "BM" …). For each combination of numbers, a list of contacts is displayed, with names that begin with one of the possible combinations. You can see a demo of the feature here http://www.youtube.com/watch?v=d9vVxuFGGZw

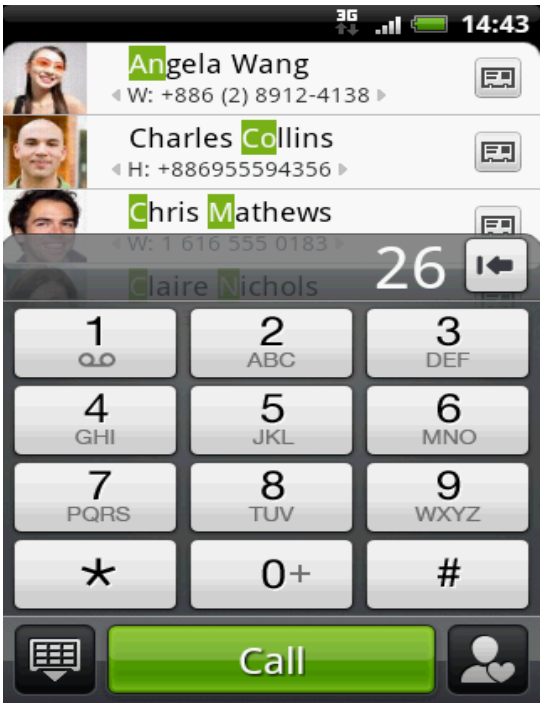

Figure 2: Smart Dial

Digit: Combination of letters

1:
2: ABC
3: DEF
4: GHI
5: JKL
6: MNO
7: PQRS
8: TUV
9: WXYZ
0: (space)

That's your task. Given a series digits you are required to print a list of all contacts that match the possible combinations **(ignore lower-case and upper-case differences)**. Of course generating all possible combinations and searching for them is too expensive (e.g. 79797979 have 65,536 combinations). We need to something better.

Think of an innovative way to solve this task.

Hint 1: A great data structure that may help you with this task is a **Trie** http://en.wikipedia.org/wiki/Trie.

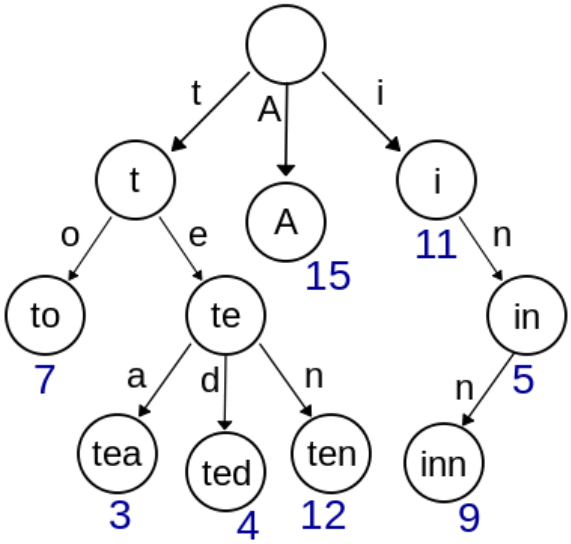

Figure 3: Example of Trie

Hint 2: Preprocessing of the names can speed up searching.

Sample Input:

6
26
646209254

Sample output:

Contacts that match 6:
Mohamed Hesham: 002022222
Mina Zaki: 002054321 – 002012345

Contacts that match 26:
Bob: 002098765

Contacts that match 646209254:
Mina Zaki: 002054321 – 002012345

CS214: Project

Contact Manager – Spelling Checker

(10 marks + 3 Bonus)

Cairo University, Faculty of Computers and Information

Project 2: Spelling Checkers

Introduction

Spelling checker is an important application program that helps writers while writing. The system flags words in a document that may not be spelled correctly. Spelling checkers may be stand-alone; operates on a block of text, or as part of a larger application; such as a word processor, email client, electronic dictionary, search engine, etc…

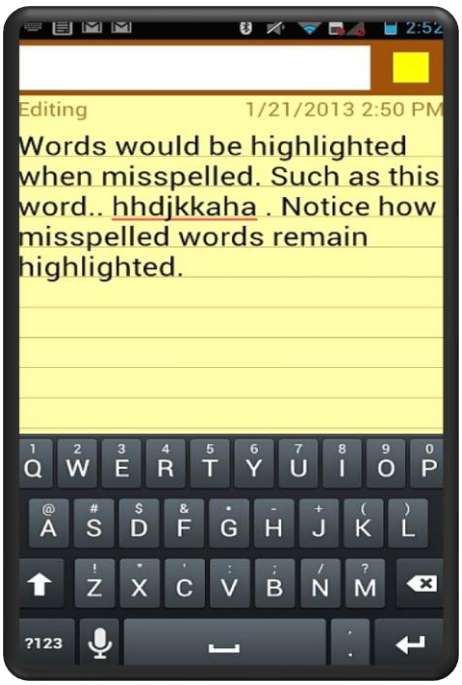

Figure 4: Android spell checker

Problem Statement

It is required to build a simple and efficient spelling checker for English text. The spelling checker will be based on verifying that all words in a given English text are well spelled.
The system is required to take a file from the user to apply spelling checking on it. If the spell checker doesn't find an exact match for the word in the dictionary, it will simply indicate the word isn't spelled correctly.

Task 1

You are given a dictionary that will be just a **plain text** file with a **word on each line**. The dictionary will be **loaded once** at the beginning of the program, it will be loaded in your data structure. The file will be named "dictionary.txt".

Another file will be given by system user to apply the spell checker on it. The file may contain punctuation, and have conventional capitalization. You will be required to handle those cases to make valid comparisons. The file name will be given by the user to the system as input.

(e.g. ***Enter the file you want to check:*** *check.txt)*

The main data structure that will be used here to load the dictionary in order to ease the searching process will be **Trie**. It is one of the best data structures to save runtime complexity and time complexity.

The figure below illustrates a Trie holding a small dictionary with 9 words.
a - at - ate - on - one - out - me - mud – my

CS214: Project

Contact Manager – Spelling Checker

(10 marks + 3 Bonus)

Cairo University, Faculty of Computers and Information

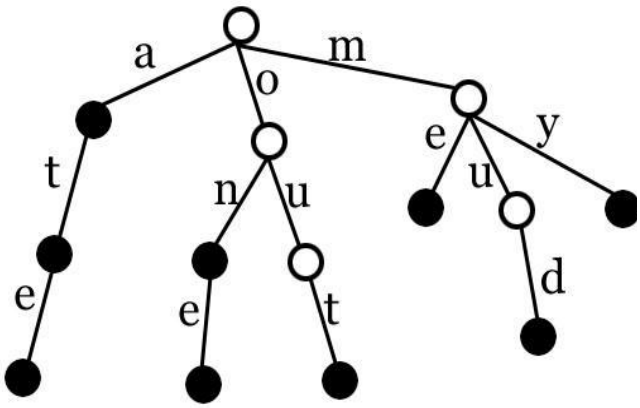

Figure 2: A Trie with a Small Dictionary

- Searching a word will need just time equals to length of the given word (complexity is **O(*l*)** where n is the word length) unlike using traditional way that will need **O(*l* * *n*)** where

  l is the word length
  n is the number of words in dictionary

- Saving this dictionary will just need 13 characters (plus some pointers) rather than traditional way that will need 21 characters.

- It is required from the system to print all the misspelled words in the file that have no matches in the dictionary. Each misspelled will be printed for the user on a separate line on the console or on a file.

Sample "check.txt":

Words will be highlighted when misspelled. Such as this word: *hhdjkkaha*.

Sample output:

hhdjkkaha

Task 2

The system will be modified to provide the user with a list of suggested spellings to choose one. When the user chooses a suggested word it will replace the misspelled word in the input file.

The suggestions will be given to the system user based on some rules which indicates whether a match is close enough to the written word.

Consider having 2 words: the word need to be checked (***word***) and a candidate word from the dictionary to replace the wrong word (***candidate***).

- If ***word*** is a prefix of ***candidate***, or ***candidate*** is a prefix of ***word***
  - e.g. *word = carp,* then *candidate* will be *carpet*
- If there are no more than third of the word size different characters between ***word*** and ***candidate***
  - e.g. *word = med candidate* will be *mud*

CS214: Project

Contact Manager – Spelling Checker

(10 marks + 3 Bonus)

Cairo University, Faculty of Computers and Information

It is required from the system to print all the wrong words with the suggestions associated to each word on a file. The file structure will be

<Wrong word> ":" <list of suggestions>
where each line holds only one word

Sample input:

We put the mu on the floor. We aty a delicious cake.

Sample output:

mu : mud - mug
aty : ate - at

Task 3 (Bonus):

Let's consider that the Trie will not only be based on English words with 26 alphabetic letters, instead it will hold Arabic words with 28 alphabetic letters or it will hold all the ASCII characters 128 characters.

Calculate the memory space needed to create the tree in figure 1 if only English letters are used or ASCII characters.

…

You will find that the memory space will be increased and based on that the Trie will be enhanced to avoid the space increasing.

You are required to modify your data structure to avoid increasing in space using hash tables. You are required to implement the Trie data structure using hashes to save space and don't lose the benefit of saving time on searching using

Think of an innovative way to solve this task.

Hint: Data structure name that will help you is called **Hash Array Mapped Trie**

CS214: Project

Contact Manager – Spelling Checker

(10 marks + 3 Bonus)

Cairo University, Faculty of Computers and Information

Submission/ Discussion Instructions

1- Each student group will submit and upload file and one zip file with their report (detailed cover page and answer to any questions that needed answers in the programming assignments). The file will be named **CS214-Project#-Task#-ID1-ID2-ID3.zip**.

   Put project ID at the first # and Task number at the second #

2- Team leader is responsible of uploading the solution. Others may also upload it as a backup.

3- TAs will determine for you a time for discussing the assignment.

4- All group members must attend to get the mark.

5- Team members can be asked about any part even if they did not do it.

Policy on Plagiarism السياسة المتبعة إزاء الغش (إقرأ جيدا جدا)

تتبنى كلية الحاسبات و المعلومات سياسة صارمة تجاه الأمانة العلمية و الغش و تشمل ما يلى:

١. تشجع الكلية على مناقشة الأفكار و تبادل المعلومات و مناقشات الطلاب حيث يعتبر هذا جوهريا لعملية تعليمية سليمة
٢. ساعد زملاءك على قدر ما تستطيع و حل لهم مشاكلهم فى الكود و لكن تبادل الحلول غير مقبول و يعتبر غشا.
٣. أى حل يتشابه مع أى حل آخر بدرجة تقطع بأنهما منقولان من نفس المصدر سيعتبر أن صاحبيهما قد قاما بالغش.
٤. قد توجد على النت برامج مشابهة لما نكتبه هنا أى نسخ من على النت يعتبر غشا يحاسب عليه صاحبه.
٥. إذا لم تكن متأكدا أن فعلا ما يعد غشا فلتسأل المعيد أو أستاذ المادة.
٦. فى حالة ثبوت الغش سيأخذ الطالب سالب درجة المسألة ، و فى حالة تكرار الغش سيرسب الطالب فى المقرر و يرفع أمره للكلية لتطبيق عقوبات مناسبة.