**CECS 503**

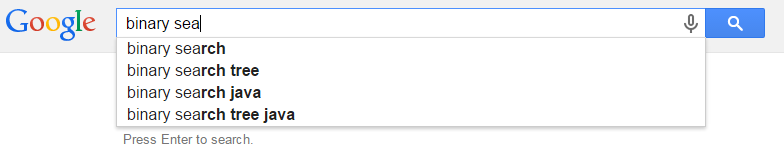
**Fall 2017**

**Project 1**

**Due Date: Oct 16**

**Introduction**:

Google search has a feature called automatic search completion. When you start typing a search query, google will try to auto complete your sentence. An example is shown below:



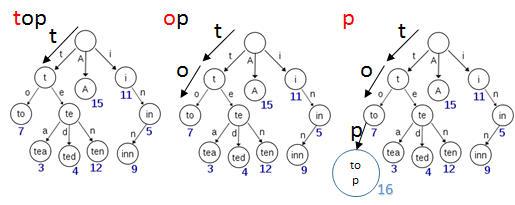
This project will simulate the same functionality of search completion using C++ and command line.

**Implementation:**

A simple search completion can be implemented using a Trie data structure. A Trie is similar to a tree but it can hold more information about data. For detailed explanation of Trie data structure, please see:

<https://en.wikipedia.org/wiki/Trie>

The way we implement the search completion is to use the edge to denote letters in a valid search query and use the node as the resulting complete search string. An example of adding word Top to our Trie is shown below.



First, starting at the root node we find the edge “t”. So we remove “t” from original string and move to the node “t” pointed by edge “t”. From the t node we find letter “o”, so we remove “o” from original string and move to the node “to” pointed by edge “o”. Then we cannot find edge “p”, so we add edge “p” to node “to” and point edge “p” to node “top”.

Notice some nodes are denoted by a number. For example, the node tea is denoted by number 3. These node indicate that the string contained in the node is a valid search query. For example, node “te” is not a valid search query while “tea”, “ted” and “ten” are. Another example is “i”, “in”, and “inn” are all valid search queries. In this project you can use a single Boolean value to indicate whether a node contains a valid search query or not.

To auto complete a query with an existing Trie, you use the depth first traversal on Trie. For example, if I started typing letter “t”, then you will start depth first search on the subtree of node “t”. After you complete your depth first search, you should return search query string “to”, “top”, “tea”, “ted” and “ten” as possible auto completion options.

**Project Details**:

This project will be two parts. Part I is to construct a Trie using a dictionary file provided. Part II is to implement a command-line search auto complete interface.

Part I:

Dictionary.txt is provided to you to construct the Trie. Each line contains a valid search query. Your task is to insert these queries into your Trie. Implement trie classes for part 1.

Part II:

Using the Trie class completed in Part I, create a C++ program that takes an user input and output auto completion options. The interface should be similar to the following:

$> Please type search queries:

$> binary sea

$> Your options are:

$> binary search

$> binary search tree

$> binary search tree java

Implement a search.cpp file to complete part II

***Important! : Please DO NOT change file names or add files on your own. I will compile your code using these file names. If your code does not compile you will get an automatic 0 point on this project.***

**Turn Ins**:

A zip file containing all required files.

**Please remember:**

All assignments and tests must be submitted on Blackboard.

All computer assignments and projects need to be written in C++ and will be submitted as follows:

1. **Visual Studio is not allowed *(IMPORTANT).***  You must use and IDE that allows you to compile and run individual C++ files. You may use Bloodshed Compiler from the link:

<http://www.bloodshed.net/dev/devcpp.html>

or any other similar environment. For linux/unix/mac users, you may use any text editor of your choice and the c/c++ compiler that comes with your system such as gcc.

1. All reports have to be submitted as a PDF report that contains:
   1. Title page with your name, assignment number and the day you are actually submitting this report (Not the assignment due date)
   2. A brief description of the assignment.
   3. A brief description of the logic employed and the needed input and expected output.
   4. A comprehensive set of snapshots showing the inputs submitted, outputs obtained in the case of a successful output or a failure.
   5. Any conclusions, analysis, or answers to any questions I as you as part of the assignment.
   6. A text file that contains all source code.
   7. Please zip both the PDF document with the source code and submit one zip file.