**Birla Institute of Technology & Science, Pilani**
**2nd Semester 2016-17 - CS F211 – Data Structures and Algorithms**

## Lab 10 – 6th April
## Topics – N-ary Trees

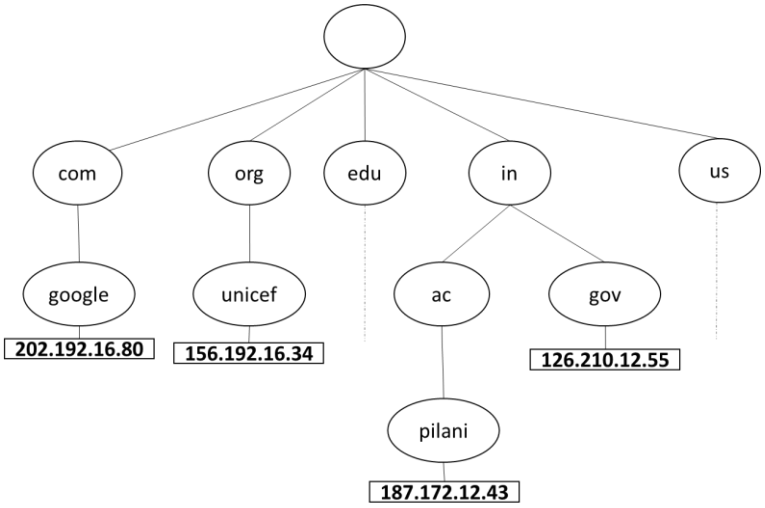
**Objective**

Domain Name System (DNS) is a hierarchical, and dynamic database that provides the mapping from hostnames to IP addresses. DNS is composed of a hierarchical domain name (ex. pilani.ac.in) space that contains a tree-like data structure of linked domain names.

In this exercise, we will construct a S tree data structure to store the domain names in a hierarchical manner so that the search of an IP address against a domain name can be performed in the number of comparisons equal to the number of levels present in the domain name. Input data will be provided as form of a file containing the list of domain names along with their IP addresses.

**Implementation**

1. Let $X_i$ be the part of a domain name $X$ present between $i^{th}$ and $i+1^{st}$ "dots", when dots are counted from right to left. For example, if $X = pilani.ac.in$ then the value of $X_3$ = $pilani$. Assume two "dots" are always present as the first and last character of a domain name.
2. An internal node (excluding root node) at level $i$ will contain a string value $Y$ as the key, if the DNS tree contains at least one domain name X such that $X_i = Y$ .
3. The leaf nodes will store the IP address of the domain name defined by the path starting from the root node to corresponding leaf node.
4. The key value for the root node can be empty or NULL.
5. Construct the DNS tree by reading the input file one line at a time (each line containing the domain name, IP address pair) and inserting the IP address of the domain name at the leaf node.
6. A single domain name may contain multiple IP address. In that case, while inserting the IP address for a domain name which is already present in the tree, you just have to add a leaf node as the child of internal node corresponding to the given domain name.
7. When first time a domain name is being inserted, it may be required to create the internal nodes at different levels of the tree. For example, (considering figure 1), if we need to add a domain name "gov.uk" then an internal node at level 1 with key value "uk" and another internal node at level 2 with key value "gov" have to be created first before creating a leaf node to enter the IP address correspond to domain name gov.uk.



**Input format**

Each line will start with a one of the following key values (1, 2, or -1). Each key corresponds to a function and has a pattern. Implement following function according to given pattern and write a driver function which can call the functions according to given key value.

| Key | Function to call | Input Format | Description |
|---|---|---|---|
| 1 | *readData* | 1 N<br>$name_1$ $IP_1$<br>$name_2$ $IP_2$<br><br>…..<br>$name_N$ $IP_N$ | Read next N lines having a single pair of <name, IP> in each line. "name" will be a valid website name (containing all lower case letters and hyphen, and arbitrary number of dots). "IP" will be a IPv4 address of the form W.X.Y.Z where W, X, Y, and Z will be integers in range [0, 255]. Keep inserting each entry pair in a tree as mentioned above. |
| 2 | *lookup* | 2 name | search for "name" in the tree. Print its IP, if entry is present. Otherwise print the child number at each level of the tree (except root) with a space separating each number. For example, if in lookup of nalanda.pilani.ac.in , there is no entry for nalanda in the node Pilani, then it should print "0 2 5" (assuming "in" is $0^{th}$ child of root, "ac" is $2^{nd}$ child of "in", and "pilani" is $5^{th}$ child of "ac"). |
| -1 | | | stop the program. |

**Test Case 1:**

| Input | Output |
|---|---|
| 1 10<br>swd.bits-pilani.ac.in 192.168.1.1<br>google.co.in 192.168.1.2<br>local.facebook.com 192.168.25.37<br>admin.nalanda.bits-pilani.ac.in 168.169.245.263<br>personal.data.yahoo.com 45.5.0.2<br>office.data.yahoo.com 45.5.0.25<br>student.csis.bits-pilani.ac.in 36.32.33.33<br>faculty.csis.bits-pilani.ac.in 37.36.33.25<br>a-b.c-d.e-f.g-h 125.145.15.25<br>a-b.c-d.e-f.g-h 125.145.15.26<br>2 google.co.in<br>2 facebook.co.in<br>2 twitter.com<br>-1 | |