**PL205 Lab Project**

**Title:** Autocomplete using Trie data structure for a phone directory

**Team Members:**

1904, Shruti Dalvi

1929, Chetana Marathe

1930, Rochelle Martins

**Description:** In computer science, a trie, also called digital tree or prefix tree, is a kind of search tree—an ordered tree data structure used to store a dynamic set or associative array where the keys are usually strings. We are given a Trie with a set of strings stored in it. Now the user types in a prefix of his search query, we need to give him all recommendations to auto-complete his query based on the strings stored in the Trie. We assume that the Trie stores past searches by the users.

For example if the Trie stores {“Aarya”, “Aakash”, “Alia”, “Aaron”, “Aparna”} and the User types in “Aa” then he must be shown contact card of {“Aarya”, “Aakash”, “Aaron”}

Using Trie, we can search a key in O(M) time, where M is maximum string length.

**Details of the Data Structures used:**

* **Trie** is a tree data structure used for storing collections of strings.

struct trie

{

struct trie \*next[CHAR\_SIZE];

int isLeaf;

char phone\_number[3][20];

};

typedef struct trie trie;

Our trie structure has 3 members,

**struct trie \*next[CHAR\_SIZE]** which is pointer to other 55 tree structures,

**int isLeaf** when set to 1 indicates end of word, and

**char phone\_number[3][20]** holds the phone number.

* **Contact** data structure

struct contact

{

char name[25]; // holds the name of the contact

char phone\_num[12]; // holds the phone number

};

typedef struct contact contact;

**Screenshots:**