HACETTEPE UNIVERSITY DEPARTMENT OF COMPUTER ENGINEERING
BBM203

Prepared by Burak YILMAZ
Number : 21627868
E-Mail : burak040898@gmail.com
Subject : Assignment 4

MAIN GOAL

In this experiment, our aim is design a Login System with Character Tree. The primary goal of this experiment is to get you to practice on the tree data structure.

PROBLEM

In this assignment, we are expected to write an application that constructs a Login System with Character Tree. The application will take the input.txt file in the current directory and read its contents. In order to create the Character tree, there are some commands for adding, deleting and refreshing.

SOLUTION

I created a character tree in alphabetic order and basically it is an ordered character tree because I consider the ascii table of each character. I first added each name's character and then for the final letter of the given username that letter has a password. After creating the tree I wrote different functions to execute the other commands such as –s,-q,-l...

ALGORITHM

Firstly, I read input.txt file line by line and for each different command I execute different functions. Creating the tree is similar to creating a linked list so it was not so hard to implement the character tree. For searching algorithm on that tree, I consider the requested username and searched that name letter by letter. So, I consider the rules that given on the pdf and gave some warning messages. Same algorithm for query operation. For delete operation I checked the given name and if the whole name is on the tree and if the last letter has the password I called an auxiliary function on that function I check that if each letter generates another name (it means that it may be the root of another name). If the letter has no children I delete that letter if it has another children or password for another name I did not delete that letter. For the listing operation, my function determines all the branches on the tree. I stored the branches on an array and that function works recursively.

FUNCTIONS/METHODS/STRUCTS

typedef struct Login_System{

```
    char letter;
    struct Login_System *children[max_letter];
    char *password;
}Login_System;
```

I used that struct for storing the data.

void File_reader(char*,Login_System*,FILE*);

-This function for reading input file line by line and calling an auxiliary function to create the tree or the generates the other auxiliary functions.

int Find_Location(char);

-This function is finding the correct location for each letter as considering the ascii corresponding integer values.

void Create_User(char*,char*,Login_System*,FILE*);

-This function creates each user. First taken each letter on that name and calling the add_tree function to add each letter to character tree. For the last letter on the given name it associates the password with that letter. If the username already taken it gives a warning message to the user.

Login_System* Add_tree(Login_System*,Login_System*,int*);

-This function is adding each letter to character tree. It works with create_user function. If the name already taken it warns the create_user function.

void Search_tree(char*,Login_System*,FILE*);

-This function searches the entire tree considering the given username. For the different computations it gives different messages or warnings which that given as a rule on pdf.

void Check_tree(Login_System*,char*,FILE*,char*);
-This function is works very similar to search tree. It checks the entire tree considering the given username. For the different computations it gives different messages or warnings which that given as a rule on pdf.

Login_System* Delete_tree(char*,Login_System*,FILE *);
-This functions is checking all the letters that given by user to delete. If the all

letters on the tree and if the last character has the password. It calls an auxiliary function called removing_name for the final checking.

void Removing_name(char*,Login_System,int);**

-This function controls the given name from delete_tree function for each letter if the letter is a part of another name It does not delete that name. It just skips. If the letter is not a part of another name and if It has no password it deletes the letter.

int Find_Child_Number(Login_System*);

-This function finds the children of each letter (children means the letters that associates with that letter.) It returns the number of children for that letter.

void List_tree_second(Login_System *root,char*,int,FILE*);

-This function list the all names on the tree and detects the each branches and prints them. It works recursively the recursive calls conditions based on the children numbers of each letter. If the letter has no children that mean it is the leaf so basically it returns. Otherwise it does some calculations and operations.