# IOS Assignment Hackathon
# 10/11/19

---

Aishwarya MA Ramanath :PES1201700872
Anagha Ananth: PES1201700301
Sharanya Venkat: PES1201700218

---

## 1. Problem Statement:

Display the contents of any directory in the form a tree.

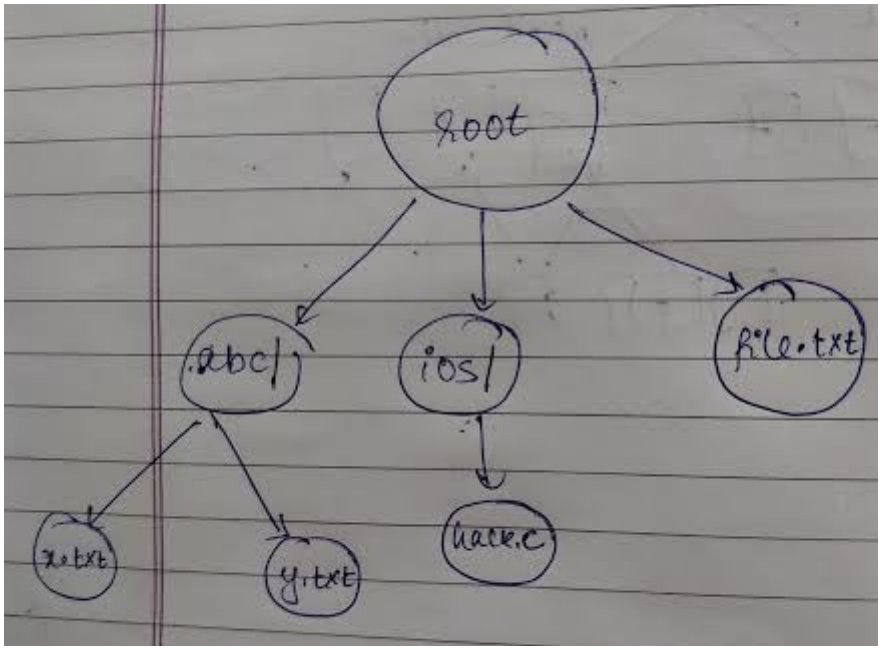
## 2. Approach:

We are traversing through a given directory and storing its contents in the form of a **binary tree**.

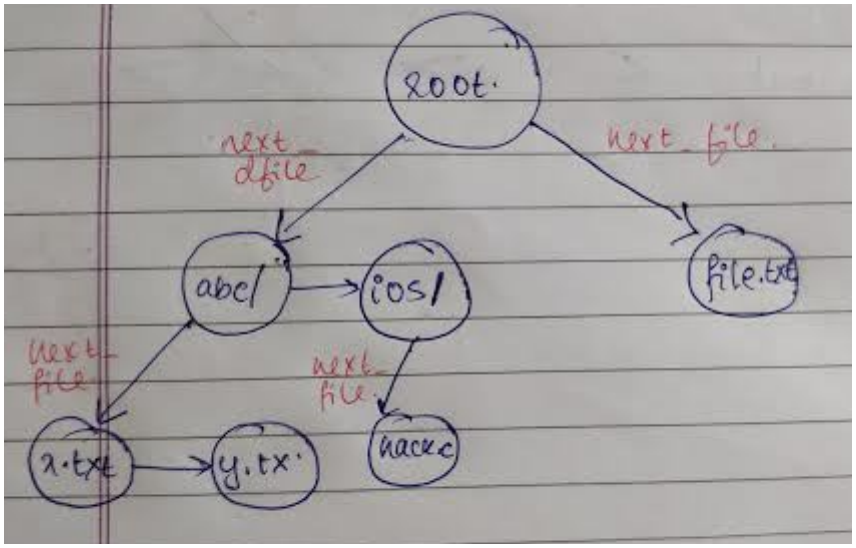
Each node of the tree contains a **linked list** of:

a. All sub-directories in its root
b. All files in its root

**Regular n-ary tree structure used in filesystem:**



**Implemented binary tree structure:**




# 3. Basic Outline:

1. **Structure used:**

```
struct t_node
{
    char* name; //name of the file or directory
    int  isdir; //flag, whose value = 1 if given file is a directory
    struct t_node *next_dfile; //linked list of sub-directories in the
given directory
    struct t_node *next_file;  //inked list of files in the given
directory
    int level; //represents level of each node in the tree
}Node;
```

2. **Main algorithm for creation of tree**

```
create_tree(directory):
   1. Dir = current directory
         a. For dr in dir:
              i.   If dr is a Directory
```

```
            1. Mark dr as a directory type of file
            2. dr-> next_dfile = create_tree(dr)
ii.   else:
            1. Mark dr as non-file
            2. Mark dr->next_dfile as NULL
```

# 4. Functions with descriptions

| **Function Name** | **Input** | **Output** | **Purpose** |
|---|---|---|---|
| create_node | String - Name of a directory | Node - with assigned attribute values | Allocates memory for a new node, and initializes its value with (name, NULL, NULL) accordingly |
| print_tree | String - Name of specified directory | None | Display Function parses through the entire tree and prints its contents. |
| create_tree (non-trivial) | String - Name of specified directory | Node - Root of created directory tree | Function to create the directory tree, which is called recursively. Opens the directory specified, checks iteratively if contents are files or directories : if directory found, recursively calls the function to parse through the directory files. |
| Summary (non-trivial) | Int- Number of directories, Int - number of files | None | Function that summarizes number of directories, files and displays depth of directory tree constructed. |

*Assumptions: Path names do not contain spaces

# 5. Global Variables

Since the display function is called recursively, the variables will keep re-initializing to 0 if not declared globally which is counter-productive, since we need to maintain a count per level in the tree.

Following are the variables defined globally:

int count - keeps track of current tree level in print_tree function

int max - used to calculate the depth of the tree

int dir_in_path - counts the number of directories in the current path

int files_in_path - counts the total number of files, including directories in the current path

# 6. To Run:

i) For a current directory: ./a.out

ii) For a specified directory: ./a.out </path/to/dir>

# 7. Output