**Submitted By: Huma Gul Mohmand**

**Department: Computer Science**

**Semester: 3rd**

**Section: C**

**Submitted To: Mr Asadullah khan**

**Course:DSA**

APPLICATIONS OF TREE DATA STRUCTURE

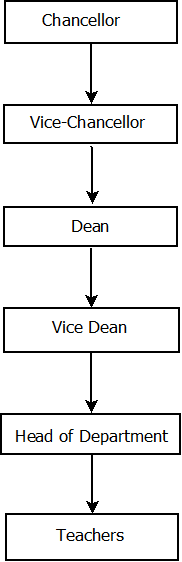
Following are the Applications of Tree Data structure.

1.ORGANISATION:

Tree Data structure in any organisation is used to mentain data and responsibility.

Example:

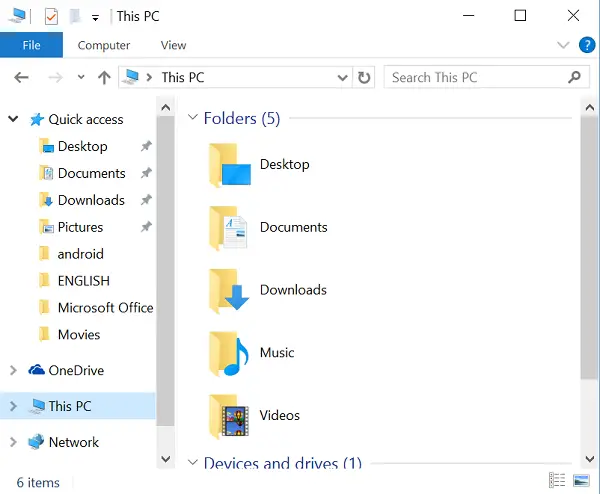
University (Organisation)



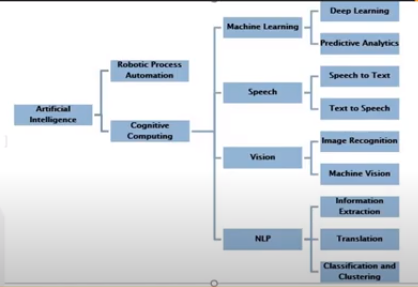
2.System File

Tree data structure is used in system files

Example: This PC is a system Folder having sub folders and sub folders further have sub folders



3.Machine Learning

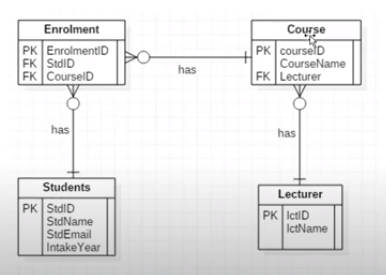


4.Database Table

If we want to design any system ,website or software we use data structure to create multiple database tables of database and data structure is used also to integrate the tables with each other .

For Example:

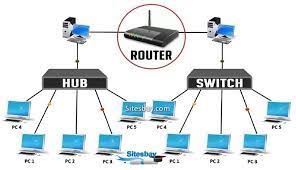
Suppose we are going to design student management system



5.Routing

Routing process works 100% on tree data structure

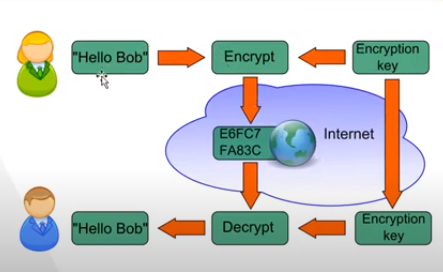
For Example:



6.Encryption

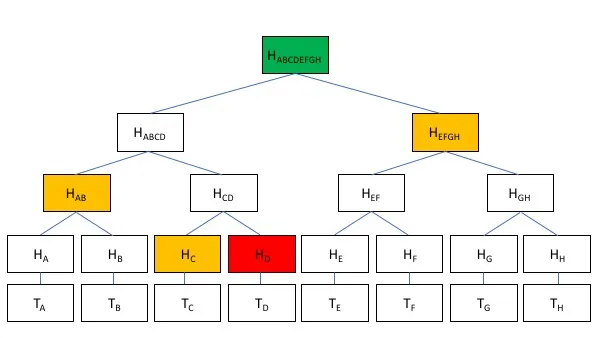
To Encrypt any data , we used thr rules of Tree Data structure

For Example:



7.Block chain

Tree Data structure is used in Block chain



8.Gaming

9.Compiler Syntax

10.Coding

11.NADRA

12.Searching

1. What is Time and Space Complexity Of algorithmAnd Write down time complexity of 7 Algorithmn?

**Time Complexity:**

Time complexity is defined as the amount of time taken by an algorithm to run, as a function of the length of the input. It measures the time taken to execute each statement of code in an algorithm. It is not going to examine the total execution time of an algorithm

**Space Complexity:**

Space complexity refers to the total amount of memory space used by an algorithm/program, including the space of input values for execution

**Time Complexity of 7 Algorithm:**

Complexities Are following

**1.Binary Search**

Worst:Log n

Average:Log n

Good:0(1)

**2:Sequential Search**

Worst:0(n)

Average:n/2

Best:0(n)

**3:Quick Sort**

Worst:0(n\*n)

Average:0(nLogn)

Best: 0(nLogn)

**4:Merge Sort**

Worst+Avg+Best=0(nLogn)

**5.Insertion Sort:**

Worst:0(n\*n)

Average: 0(n\*n)

Best:0(n)

**6: Bubble Sort**

Worst+Avg+Best=0(n\*n)

**7:Heap Sort**

Worst+Avg+Best=n(nlogn)