namework - suptrie of another tree

Week 6 - Trees - 2-08-2017

Gruen two non-empty bunury trees & and +, check whether tree + now exactly the same structure and hode values with a subtree of s. A subtree of s is a tree that consists of a rade in s and all of their rades descendants.

T-TALK

- · Can the trees be empty? No
- espon toos owt? Two root nodes
- · Output? Boolean
- · MANUSCHERN MULICATION WISH SON
- · No need to create tree structure
- * Binary tree can have deplicate values. (not BST)
- o Binary tree which only contains numbers

(S) 10 (A) 2 1) 2 13 / 3 true 1 3 simple 5 success 3 10 (1) 2 X 2) 2 13 1 3 false 5 failure 3) 9 9 5 5 Same tree Single nock single nocle same engle node to 15e both mull true of body false

B- BRUTE FORCE

- o Traverse both trees and store the value in rectors different.
- Find if the subsequence of + is in 5.
- o Needs extra memory for storing values in the vectors.

· Recursinely traverse the

the node of t and S.

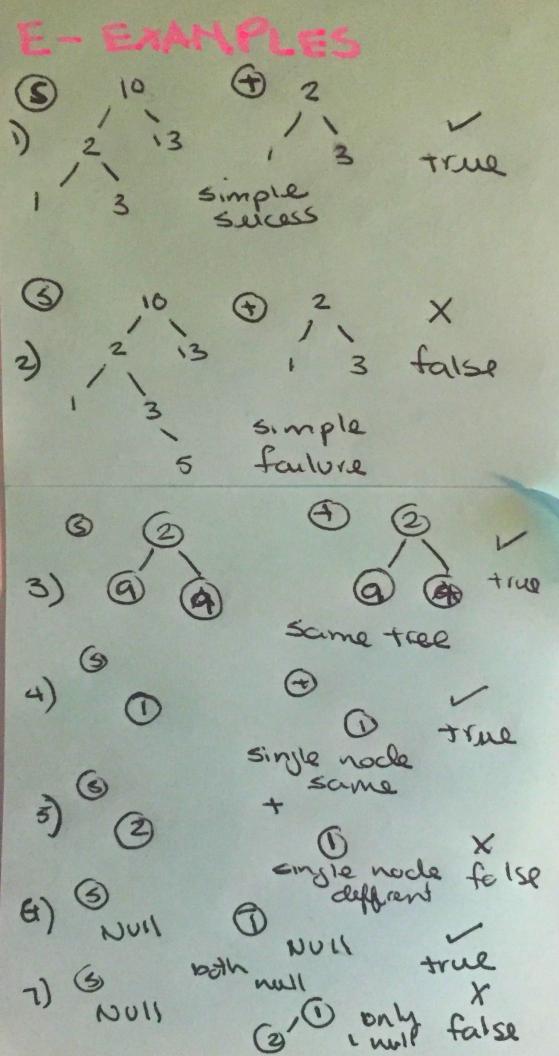
and we fying if they

ore identical.

Tima complexity (m * n) nodes nodes in t

T-TALK

- · Can the trees be empty? No
- cabon toor owt? TugaTe.
- · Output? Boolean
- · MANUALINE MANUAL CONTRACTION
- · No need to create tree structure
- * Binary tree can have duplicate values. (not BST)
- o Binary tree which only countains numbers



B-BRUTE FORCE

- o Traverse both trees and store the values in rectors different.
- Find if the subsequence of + is in 5.
- o Needs extra memory far storing values in the vectors.

MOZFATZHTZON

Recursinally traverse the tree in preorder, componing the nodes of t and S. and usinfying if they are identical.

Time complexity (m " n)
nodes in + " "

W-WALKTHROUGH · Function to see is they ere Equal ties redurns boolean TIF S RR TOWNING NULL - Then true > If only 1 = NUI > their false - " Check both trees structures recursively a Function to traveise in preorder time is 5 != NUII and if not equal traverse 3. · Main function to call traverse) I-IMPLE -MENTATION

bool is Equal (node *5, node *+) {

if (5 == NULL 22 + == NULL) {

return + rue;

if (5 == NULL 1) + == NULL) {

return folse;

return folse;

is Equal (50 left, +0 left) 22

is Equal (50 left, +0 left) 22

is Equal (50 right, +0 right);

Trunction to see is they

cre Equal tiess

Freducts boolson

If S PR T crive NULL

Then true

Then true

The chily 1 = NUII

The folce

The chily 1 = NUII

The child to the course of the child the child the course of the child the child the course of the child the child

bool traverse Preorder (node *s, node *+) {

return s!= NULL &&

(is Equal(s, +) 11 traverse Preorder (s=) left, +) 11

traverse Preorder (s=) ryht, +));

bool is Subtree (node *s, node *+) {

return traverse Preorder (s, +);

5 1 = NUII 7 5= 10 T- TEST 1 is Equal (10,2) 7 faise 57 left = 2 3 1= NUL - SEZ is Equal (1,1) strue } Success is Equal (2,2) strue is Equal (3,3) strue 51= HULL 7 5=10 (1) Equal (10,2) = falso 5-141 = 2 5 = NULL = 5=2 X Foul 15 Equal (2,2) -7+1 is Equal (1, 0 = two 15 Equal(3, UVII) > False X

I-IMPLE - J. S. W. A. MENTATION bool u Equal (node +5, node +7) { if (5 == NULL 22 + == NULL) { 3 return true; # (S== NULL 1) + == NULL) { return false; 5-> val == + > val 1. 12 naturn is Equal (5-) left, += left) el is Equal (5->ryht, +->right) bool traverse Precider (node *5, node ++){ Ketuin S != NULL EE (is Equal(s, +) 11 +ranerse Percoder (s=) left,+) 11

```
o Function to see is
   cre Equal Ties
     - returns boolean
    JIF S PR T CIVING
       - then true
    ">If only 1 = 1
     > they false
   -> Check both tree
     structures recursi
· Function to traveis
 preorder
and return true is a
Main function to call
```

- WALKTHRO

traverse Precider (soryht, +)); bool is Subree (node *s, node ++) { return traveve Preorder (s.+);

51=NUI 7 5= 10 T- TEST is Equal (10,2) - faise B 1= NUL - SEZ 15 Equal (2,2) strue SUCCESS is End (1) stine is Equal (3,3) >+140 51= HU1 7 5=10 nEdrol (10's)stand 5- WH = 2 51= NULL - 5=2 X Foul is Equal (2,2) -7+12 s Equal (1, 0 2 the is Equal(3, UVII) > False X