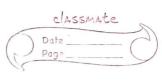
R. Darshoun Norgpal RVCERA HCAOII



	TO GO WET THEIR TO THE THEIR THE THEIR THE
	MIN AND MAX IN BST
	To final the uninum and plaximum value in
	To final the minimum and maximum value in a Binary Search tree we use a property called inorder traversal.
	Called morder travered.
	-> The first (leftmost bottom) element is the
	-> The last (Rightmost bottom) element is the
	Maximum.
\rightarrow	Algorithm: (working of steps):
	Sra John Burtle Land
	Step1: Create as BST
	Step 2: Sorting of BST in "inorder" method. Step 3: Visit each node in inorder traversal of
	technique.
	Step 4: The left most in the bottom is minimum. Step 5: The Rightmost in the bottom is maximum.
	Step 5. The Rightmost in the Dollan is maximum.
-	Program: ('C'):
	U T
	#include < etilio.h>
	#include < linuits. h>
	#include < stdlib.h>
	Strut Node
	2 h dala
	int data;
	Struct Node* left, right;
	J/



	Int rinvalue (struct Node* 150t)
	E STATE CALL WAS NOOLE
	if (root == NULL) {
	return (-1);
,	3
	Struct Norde* Cur = root;
	while (Curs -> left! = NULL)
	cuser = cuser -> left;
	return (cur -> data);
	<u>\$</u>
	int max Value (struct Node * root)
	if (roof @== NULL)
	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
	redugn(-1);
. (12)	· · · · · · · · · · · · · · · · · · ·
· jelvatas	Struct Nodet cus = root;
	while (cus -> right! = NULL)
	{
	Cuer= Cuer -> right;
	return (cor->data);
	· · · · · · · · · · · · · · · · · · ·
	Struct Node+ createNate(int val)
	{
	Struct Node* node = (Struct Node*) mallor (Size of (struct Node))-
	node > data = valo;
	node > left = node > right = NULL;

	7.
	int wain()
	s and
	Strut Node * root = create Node (5);
	root > left = create Node (4);
	root = right = createNode (6);
	root > left -> (eft = create Node (3);
	root -> right -> right = create Node (7);
	root -> right -> right = create Node (7);
	printf("/d/n", minsalue (root));
	print (">dln", manvalue (root));
	refuento);
	3
<u></u>	Output: Complexity: Time = O(h) >h - height
	· Time = O(h) >h - height
	Space = 0(1)
	
	0 11' 0 01
→	Representation of BST:
	(5)
	(4) (6)
	(3) (7)