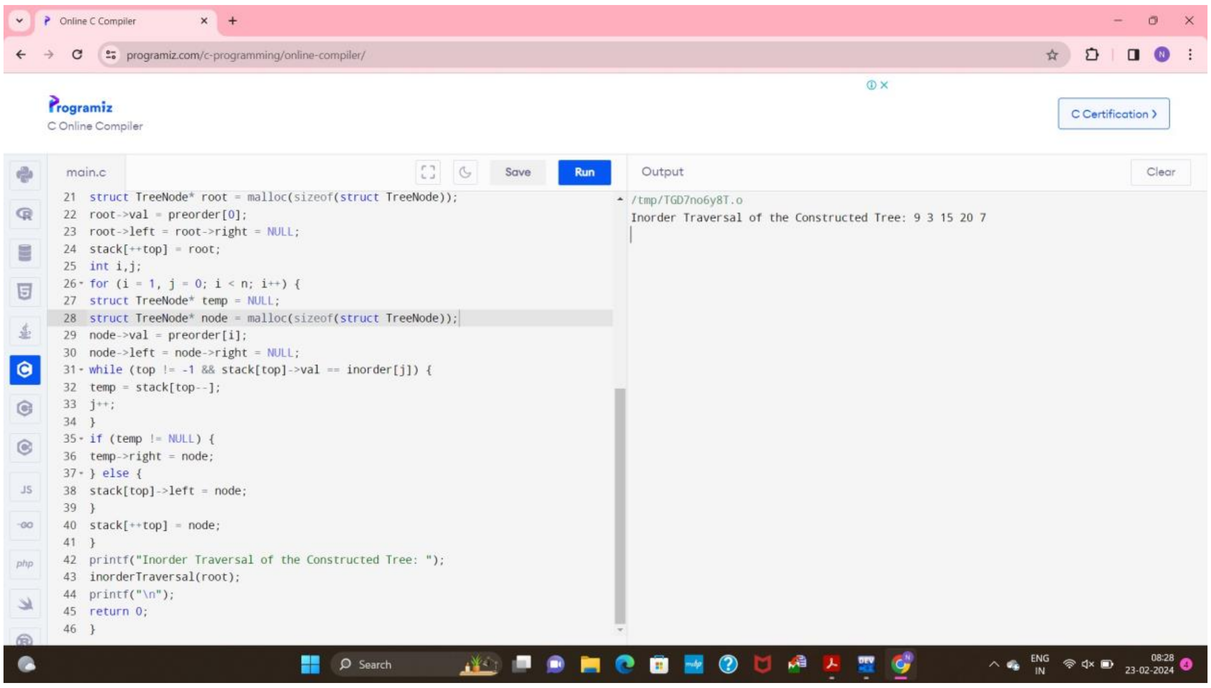
```
#include <stdio.h>
#include <stdlib.h>
struct TreeNode {
int val;
struct TreeNode* left;
struct TreeNode* right;
};
void inorderTraversal(struct TreeNode* root) {
if (root != NULL) {
inorderTraversal(root->left);
printf("%d ", root->val);
inorderTraversal(root->right);
}
}
int main() {
int preorder[] = {3, 9, 20, 15, 7};
int inorder[] = {9, 3, 15, 20, 7};
int n = sizeof(preorder) / sizeof(preorder[0]);
struct TreeNode* stack[100];
int top = -1;
struct TreeNode* root = malloc(sizeof(struct TreeNode));
root->val = preorder[0];
root->left = root->right = NULL;
stack[++top] = root;
int i,j;
for (i = 1, j = 0; i < n; i++) {
struct TreeNode* temp = NULL;
struct TreeNode* node = malloc(sizeof(struct TreeNode));
node->val = preorder[i];
node->left = node->right = NULL;
while (top != -1 && stack[top]->val == inorder[j]) {
```

```
temp = stack[top--];
j++;
}
if (temp != NULL) {
temp->right = node;
} else {
stack[top]->left = node;
}
stack[++top] = node;
}
printf("Inorder Traversal of the Constructed Tree: ");
inorderTraversal(root);
printf("\n");
return 0;
}
```



```
struct TreeNode* root = malloc(sizeof(struct TreeNode));
root->val = preorder[0];
root->left = root->right = NULL;
stack[++top] = root;
int i,j;
for (i = 1, j = 0; i < n; i++) {
struct TreeNode* temp = NULL;
struct TreeNode* node = malloc(sizeof(struct TreeNode));
node->val = preorder[i];
node->left = node->right = NULL;
while (top != -1 && stack[top]->val == inorder[j]) {
temp = stack[top--];
j++;
}
if (temp != NULL) {
temp->right = node;
} else {
stack[top]->left = node;
}
stack[++top] = node;
}
printf("Inorder Traversal of the Constructed Tree: ");
inorderTraversal(root);
printf("\n");
return 0;
}
```

```
/tmp/TGD7no6y8T.o
Inorder Traversal of the Constructed Tree: 9 3 15 20 7
```