

```

1: // C program to find maximum and minimum in a Binary Tree
2: #include <limits.h>
3: #include <stdio.h>
4: #include <stdlib.h>
5:
6: // A tree node
7: struct Node {
8:     int data;
9:     struct Node *left, *right;
10: };
11:
12: // A utility function to create a new node
13: struct Node* newNode(int data)
14: {
15:     struct Node* node
16:         = (struct Node*)malloc(sizeof(struct Node));
17:     node->data = data;
18:     node->left = node->right = NULL;
19:     return (node);
20: }
21:
22: // Returns maximum value in a given Binary Tree
23: int findMax(struct Node* root)
24: {
25:     // Base case
26:     if (root == NULL)
27:         return INT_MIN;
28:
29:     // Return maximum of 3 values:
30:     // 1) Root's data 2) Max in Left Subtree
31:     // 3) Max in right subtree
32:     int res = root->data;
33:     int lres = findMax(root->left);
34:     int rres = findMax(root->right);
35:     if (lres > res)
36:         res = lres;
37:     if (rres > res)
38:         res = rres;
39:     return res;

```

```

40: }
41:
42: // Returns minimum value in a given Binary Tree
43: int findMin(struct Node* root)
44: {
45:     // Base case
46:     if (root == NULL)
47:         return INT_MAX;
48:
49:     // Return minimum of 3 values:
50:     // 1) Root's data 2) Max in Left Subtree
51:     // 3) Max in right subtree
52:     int res = root->data;
53:     int lres = findMin(root->left);
54:     int rres = findMin(root->right);
55:     if (lres < res)
56:         res = lres;
57:     if (rres < res)
58:         res = rres;
59:     return res;
60: }
61:
62:
63: // Driver code
64: int main(void)
65: {
66:     struct Node* NewRoot = NULL;
67:     struct Node* root = newNode(2);
68:     root->left = newNode(7);
69:     root->right = newNode(5);
70:     root->left->right = newNode(6);
71:     root->left->right->left = newNode(1);
72:     root->left->right->right = newNode(11);
73:     root->right->right = newNode(9);
74:     root->right->right->left = newNode(4);
75:
76:     // Function call
77:     printf("Maximum element is %d \n", findMax(root));
78:     printf("Minimum element is %d \n", findMin(root));

```

```
79:
80:     return 0;
81: }
82:
```