Lab Question 15: Binary Search Tree

Aim:
To write a C program to implement BST with search, find min, and find max.

Algorithm:

1. Start the program.
2. Define BST node structure.
3. For insert: insert as in BST.
4. For search: traverse left/right until found.
5. For min: go left until null.
6. For max: go right until null.
7. Stop.

Code:

```
#include <stdio.h>
#include <stdlib.h>

struct Node{int key; struct Node* left,*right;};

struct Node* newNode(int key){
   struct Node* n=(struct Node*)malloc(sizeof(struct Node));
   n->key=key; n->left=n->right=NULL; return n;
}

struct Node* insert(struct Node* root,int key){
   if(!root) return newNode(key);
   if(key<root->key) root->left=insert(root->left,key);
   else if(key>root->key) root->right=insert(root->right,key);
   return root;
}

int search(struct Node* root,int key){
   if(!root) return 0;
```

```
    if(root->key==key) return 1;
    if(key<root->key) return search(root->left,key);
    return search(root->right,key);
}

int findMin(struct Node* root){ while(root->left) root=root->left; return root->key; }
int findMax(struct Node* root){ while(root->right) root=root->right; return root->key; }

int main(){
    struct Node* root=NULL;
    root=insert(root,50); root=insert(root,30); root=insert(root,70);
    root=insert(root,20); root=insert(root,40); root=insert(root,60); root=insert(root,80);
    printf("Search 40: %s\n", search(root,40)?"Found":"Not Found");
    printf("Min: %d\n",findMin(root));
    printf("Max: %d\n",findMax(root));
    return 0;
}
```

Output:

- BST from {50,30,70,20,40,60,80}
- Search 40 → Found
- Min = 20, Max = 80

Result:
The program implements BST operations successfully.