```
[] 🕓 🖒 Share
                                                                                                                                                       Clear
                                                                         Run
                                                                                    Output
       main.c
       1 #include <stdio.h>
                                                                                   20 30
                                                                                   Search 20: Found
       2 #include <stdlib.h>
       3 - typedef struct Node {
              int key, height;
              struct Node *left, *right;
                                                                                   === Code Execution Successful ===
       6 } Node;
回
       7 int h(Node *n) { return n ? n->height : 0; }
       8 int max(int a, int b) { return a > b ? a : b; }
       9 int bal(Node *n) { return n ? h(n->left) - h(n->right) : 0; }
       10 - Node* newNode(int key) {
       11
              Node *n = malloc(sizeof(Node));
              n->key = key; n->left = n->right = NULL; n->height = 1;
       12
       13
              return n;
      14 }
       15 - Node* rotR(Node *y) {
0
              Node *x = y->left, *T = x->right;
       16
              x->right = y; y->left = T;
       17
              y->height = 1 + max(h(y->left), h(y->right));
       18
              x->height = 1 + max(h(x->left), h(x->right));
       19
      20
              return x;
      21 }
       22 - Node* rotL(Node *x) {
              Node *y = x - right, *T = y - right;
      23
       24
              y->left = x; x->right = T;
       25
              x \rightarrow height = 1 + max(h(x \rightarrow left), h(x \rightarrow right));
              y->height = 1 + max(h(y->left), h(y->right));
       26
```

```
[] & of Share
                                                                                          Output
                                                                                                                                                                 Clear
       main.c
                                                                                        20 30
       27
               return y;
                                                                                        Search 20: Found
       28 }
       29 - Node* insert(Node *n, int key) {
               if (!n) return newNode(key);
       30
                                                                                        === Code Execution Successful ===
                if (key < n->key) n->left = insert(n->left, key);
       31
               else if (key > n->key) n->right = insert(n->right, key);
       32
9
       33
               else return n;
               n\rightarrow height = 1 + max(h(n\rightarrow left), h(n\rightarrow right));
               int b = bal(n);
       35
       36
               if (b > 1 && key < n->left->key) return rotR(n);
               if (b < -1 && key > n->right->key) return rotL(n);
if (b > 1 && key > n->left->key) { n->left = rotL(n->left);
       37
       38
                   return rotR(n); }
       39
                if (b < -1 && key < n->right->key) { n->right = rotR(n->right);
                   return rotL(n); }
0
       40
               return n;
       41 }
       42 * Node * minVal(Node *n) {
       43
               while (n->left) n = n->left;
       44
               return n;
       45 }
       46 - Node* delete(Node *n, int key) {
       47
               if (!n) return n;
                if (key < n->key) n->left = delete(n->left, key);
       49
               else if (key > n->key) n->right = delete(n->right, key);
      50-
               else {
```

```
[] & & Share
                                                                                      Output
                                                                                                                                                          Clear
       main.c
                                                                                     20 30
                   if (!n->left || !n->right) {
      51 -
                                                                                     Search 20: Found
       52
                       Node *tmp = n->left ? n->left : n->right;
                       if (!tmp) tmp = n, n = NULL;
      53
       54
                       else *n = *tmp;
                                                                                     === Code Execution Successful ===
                       free(tmp);
       55
0
       56 -
                   } else {
       57
                       Node *tmp = minVal(n->right);
       58
                       n->key = tmp->key;
       59
                       n->right = delete(n->right, tmp->key);
       60
       61
               if (!n) return n;
       62
       63
               n\rightarrow height = 1 + max(h(n\rightarrow left), h(n\rightarrow right));
               int b = bal(n);
       64
0
       65
              if (b > 1 && bal(n->left) >= 0) return rotR(n);
       66
              if (b > 1 && bal(n->left) < 0) { n->left = rotL(n->left); return
                   rotR(n); }
       67
               if (b < -1 && bal(n->right) <= 0) return rotL(n);
               if (b < -1 \&\& bal(n->right) > 0) { n->right = rotR(n->right);}
       68
TS
                  return rotL(n); }
       69
               return n;
      70 }
       71 - Node* search(Node *n, int key) {
               if (!n || n->key == key) return n;
       72
      73
               return key < n->key ? search(n->left, key) : search(n->right,
                   key);
```

```
∞ Share
                                                                                                                                                          Clear
                                                                                    Output
  main.c
                                                                         Run
                                                                                  20 30
               rotR(n); }
          if (b < -1 && bal(n->right) <= 0) return rotL(n);
if (b < -1 && bal(n->right) > 0) { n->right = rotR(n->right);
                                                                                  Search 20: Found
  67
  68
              return rotL(n); }
                                                                                  === Code Execution Successful ===
  69
          return n;
  70 }
  71 - Node* search(Node *n, int key) {
          if (!n || n->key == key) return n;
  72
  73
          return key < n->key ? search(n->left, key) : search(n->right,
              key);
  74 }
  75 - void pre(Node *n) {
          if (n) { printf("%d ", n->key); pre(n->left); pre(n->right); }
  76
  77 }
  78 - int main() {
  79
          Node *root = NULL;
          root = insert(root, 30);
  80
  81
          root = insert(root, 10);
  82
          root = insert(root, 20);
          root = delete(root, 10);
  83
          pre(root);
          printf("\nSearch 20: %s\n", search(root, 20) ? "Found" : "Not
  85
  86
          return 0;
  87 }
88
```

9