**1)Python:**

**def min\_value\_node(node):**

current = node

while current.left is not None:

current = current.left

return current

**def delete\_node(t, key):**

if t is None:

return t

if key < t.data:

t.left = delete\_node(t.left, key)

elif key > t.data:

t.right = delete\_node(t.right, key)

else:

if t.left is None:

temp = t.right

t = None

return temp

elif t.right is None:

temp = t.left

t = None

return temp

temp = min\_value\_node(t.right)

t.data = temp.data

t.right = delete\_node(t.right, temp.data)

return t

**3)Java:**

**public static int minValue(BinarySearchTreeNode t) {**

int minv = t.data;

while (t.left != null) {

minv = t.left.data;

t = t.left;

}

return minv;

}

**public static BinarySearchTreeNode delete\_node(BinarySearchTreeNode t, int key) {**

if (t == null)

return t;

if (key < t.data)

t.left = delete\_node(t.left, key);

else if (key > t.data)

t.right = delete\_node(t.right, key);

else {

if (t.left == null)

return t.right;

else if (t.right == null)

return t.left;

t.data = minValue(t.right);

t.right = delete\_node(t.right, t.data);

}

return t;

}

**3. C**

**struct BinarySearchTreeNode \*min\_value\_node(struct BinarySearchTreeNode \*node) {**

struct BinarySearchTreeNode \*current = node;

while (current && current->left != NULL)

current = current->left;

return current;

}

**BinarySearchTreeNode\* delete\_node(BinarySearchTreeNode\* t, int key) {**

if (t == NULL) return t;

if (key < t->data)

t->left = delete\_node(t->left, key);

else if (key > t->data)

t->right = delete\_node(t->right, key);

else {

if (t->left == NULL) {

struct BinarySearchTreeNode \*temp = t->right;

free(t);

return temp;

} else if (t->right == NULL) {

struct BinarySearchTreeNode \*temp = t->left;

free(t);

return temp;

}

struct BinarySearchTreeNode \*temp = min\_value\_node(t->right);

t->data = temp->data;

t->right = delete\_node(t->right, temp->data);

}

return t;

}

**4. C++:**

**struct BinarySearchTreeNode \*min\_value\_node(struct BinarySearchTreeNode \*node) {**

struct BinarySearchTreeNode \*current = node;

while (current && current->left != NULL)

current = current->left;

return current;

}

**BinarySearchTreeNode\* delete\_node(BinarySearchTreeNode\* t, int key) {**

if (t == NULL) return t;

if (key < t->data)

t->left = delete\_node(t->left, key);

else if (key > t->data)

t->right = delete\_node(t->right, key);

else {

if (t->left == NULL) {

struct BinarySearchTreeNode \*temp = t->right;

free(t);

return temp;

} else if (t->right == NULL) {

struct BinarySearchTreeNode \*temp = t->left;

free(t);

return temp;

}

struct BinarySearchTreeNode \*temp = min\_value\_node(t->right);

t->data = temp->data;

t->right = delete\_node(t->right, temp->data);

}

return t;

}