UNIVERSITI TEKNOLOGI PETRONAS

LAB

JAN 2023

TEB1113

Algorithm & Data Structure

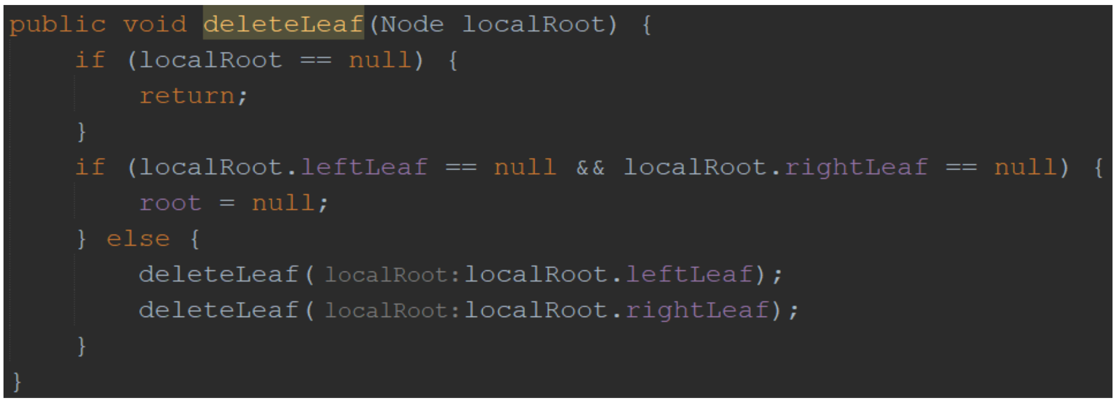
Lab 9

NO.	NAME	STUDENT ID	PROGRAM (IT / IS / CS / BM)
1.	**CHENG PIN-JIE**	**21000548**	**CS**

Assoc. Prof. Dr Manzoor Ahmed Hashmani

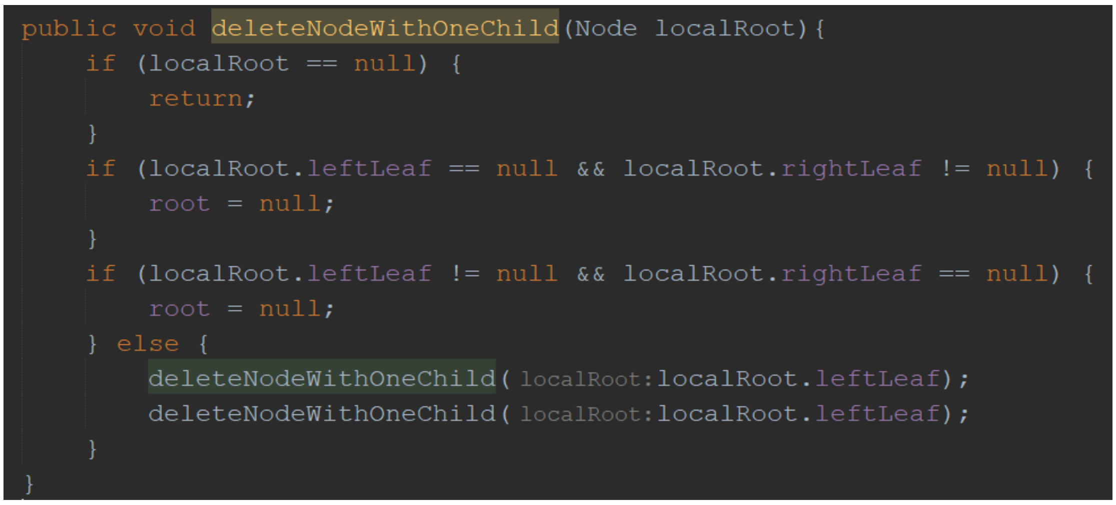
Madam Maryam Omar Abdullah Sawad

1. The node to be deleted is a leaf (has no children).



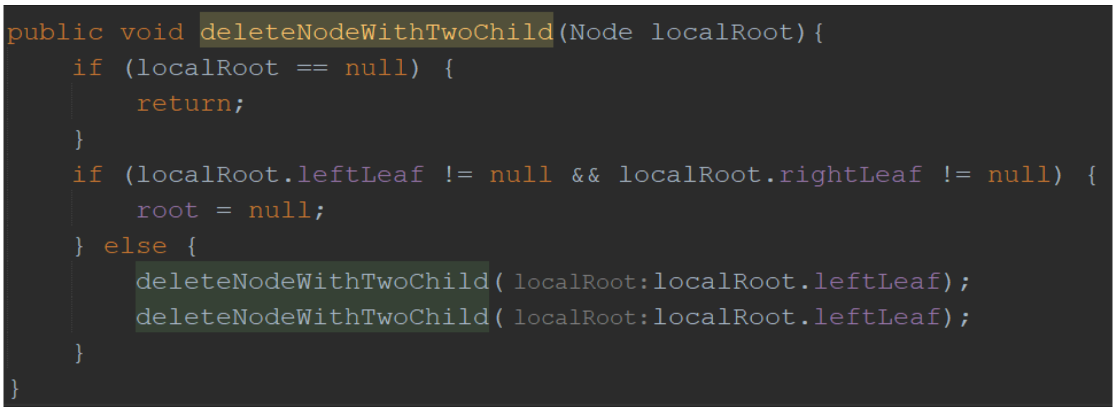
```java
public void deleteLeaf(Node localRoot) {
    if (localRoot == null) {
        return;
    }
    if (localRoot.leftLeaf == null && localRoot.rightLeaf == null) {
        root = null;
    } else {
        deleteLeaf( localRoot:localRoot.leftLeaf);
        deleteLeaf( localRoot:localRoot.rightLeaf);
    }
}
```

2. The node to be deleted has one child.



```java
public void deleteNodeWithOneChild(Node localRoot){
    if (localRoot == null) {
        return;
    }
    if (localRoot.leftLeaf == null && localRoot.rightLeaf != null) {
        root = null;
    }
    if (localRoot.leftLeaf != null && localRoot.rightLeaf == null) {
        root = null;
    } else {
        deleteNodeWithOneChild( localRoot:localRoot.leftLeaf);
        deleteNodeWithOneChild( localRoot:localRoot.leftLeaf);
    }
}
```

3. The node to be deleted has two children.



```java
public void deleteNodeWithTwoChild(Node localRoot){
    if (localRoot == null) {
        return;
    }
    if (localRoot.leftLeaf != null && localRoot.rightLeaf != null) {
        root = null;
    } else {
        deleteNodeWithTwoChild( localRoot:localRoot.leftLeaf);
        deleteNodeWithTwoChild( localRoot:localRoot.leftLeaf);
    }
}
```

4. Design Heap Sort

```java
public class HeapSort
{
    public static void sort(int arr[])
    {
        // Build Max-Heap
        int size = arr.length;
        for (int i = size / 2; i >= 0; i--)
            heapify(arr, size, i);

        // Extract element from heap
        for (int i = size - 1; i >= 0; i--) {
            int temp = arr[0];
            arr[0] = arr[i];
            arr[i] = temp;

            heapify(arr, i, 0);
        }
    }

    public static void heapify(int arr[], int size, int i)
    {
        int largest = i;
        int leftChild = 2 * i;
        int rightChild = 2 * i + 1;

        // if leftChild > root
        if (leftChild < size && arr[leftChild] > arr[largest])
            largest = leftChild;
        // if rightChild > root
        if (rightChild < size && arr[rightChild] > arr[largest])
            largest = rightChild;
        if (largest != i) {
            int temp = arr[i];
            arr[i] = arr[largest];
            arr[largest] = temp;

            heapify(arr, size, largest);
        }
    }

    public static void printArr(int arr[])
    {
        int size = arr.length;
        for (int i = 0; i < size; i++)
            System.out.print(arr[i] + " ");
    }

    public static void main(String args[])
    {
        int arr[] = {1, 10, 20, 35, 4, 6, 14};

        System.out.println("Unsorted Array: ");
        printArr(arr);
        System.out.println(" ");

        sort(arr);

        System.out.println("Sorted Array: ");
        printArr(arr);
    }
}
```

5. Time taken for insertion sort algorithm to sort the array

   n = 5

   $O(n^2) = 25$