**DSA Lab 09**

**Name:** Hafsa Salman

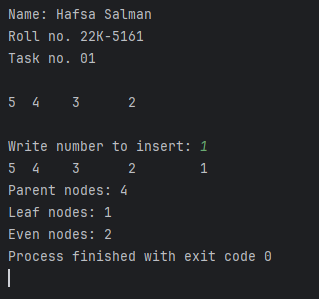
**Roll no.** 22K-5161

**Task no. 01**

Code:

//Hafsa Salman  
//22K-5161  
//Task no. 01  
  
import java.util.Scanner;  
  
public class Task\_01  
{  
 public static void main(String[] args)  
 {  
 System.*out*.println("Name: Hafsa Salman");  
 System.*out*.println("Roll no. 22K-5161");  
 System.*out*.println("Task no. 01");  
 System.*out*.println();  
  
 Scanner s = new Scanner(System.*in*);  
  
 Binary BST = new Binary();  
  
 BST.Insert(5);  
 BST.Insert(4);  
 BST.Insert(3);  
 BST.Insert(2);  
  
 BST.Print();  
  
 int node;  
  
 System.*out*.print("\n\nWrite number to insert: ");  
 node = s.nextInt();  
  
 BST.Insert(node);  
  
 BST.Print();  
  
 int a, b, c;  
  
 a = BST.Parent(BST.root);  
 b = BST.Leaf(BST.root);  
 c = BST.Even(BST.root);  
  
 System.*out*.print("\nParent nodes: " + a);  
 System.*out*.print("\nLeaf nodes: " + b);  
 System.*out*.print("\nEven nodes: " + c);  
 }  
}  
  
class TreeNode  
{  
 int data;  
 TreeNode left;  
 TreeNode right;  
  
 public TreeNode (int data)  
 {  
 this.data = data;  
 this.right = null;  
 this.left = null;  
 }  
}  
  
class Binary  
{  
 TreeNode root;  
  
 public Binary ()  
 {  
 this.root = null;  
 }  
  
 public void Insert (int data)  
 {  
 root = insertRec(root,data);  
 }  
  
 TreeNode insertRec (TreeNode root, int data)  
 {  
 if (root == null)  
 {  
 root = new TreeNode(data);  
  
 return root;  
 }  
  
 if (data < root.data)  
 {  
 root.left = insertRec(root.left, data);  
 }  
  
 else if (data > root.data)  
 {  
 root.right = insertRec(root.right, data);  
 }  
  
 return root;  
 }  
  
 public void Delete (int data)  
 {  
 root = deleteRec(root, data);  
 }  
  
 TreeNode deleteRec (TreeNode root, int data)  
 {  
 if (root == null)  
 {  
 System.*out*.println("Tree is empty");  
  
 return root;  
 }  
  
 if (data < root.data)  
 {  
 root.left = deleteRec(root.left, data);  
 }  
  
 else if(data > root.data)  
 {  
 root.right = deleteRec(root.right, data);  
 }  
  
 else  
 {  
 if(root.left == null)  
 {  
 return root.right;  
 }  
  
 else if(root.right == null)  
 {  
 return root.left;  
 }  
  
  
 root.data = minVal(root.right);  
 root.right = deleteRec(root.right, data);  
 }  
  
 return root;  
 }  
  
 int minVal(TreeNode root)  
 {  
 int min;  
  
 min = root.data;  
  
 while(root.left != null)  
 {  
 min = root.left.data;  
 root = root.left;  
 }  
  
 return min;  
 }  
  
 public void Print ()  
 {  
 PrintTree (root, 0);  
 }  
  
 public void PrintTree (TreeNode n, int level)  
 {  
 if (n == null)  
 {  
 return;  
 }  
  
 PrintTree(n.right, level + 1);  
  
 for (int i=0; i<level; i++)  
 {  
 System.*out*.print(" ");  
 }  
  
 System.*out*.print(n.data);  
  
 PrintTree(n.left, level + 2);  
 }  
  
 public int Parent(TreeNode n)  
 {  
 if (n == null)  
 {  
 return 0;  
 }  
  
 int count;  
  
 count = 0;  
  
 if (n.left != null || n.right != null)  
 {  
 count = 1;  
 }  
  
 count = count + Parent(n.left) + Parent(n.right);  
 {  
 return count;  
 }  
 }  
  
 public int Leaf (TreeNode n)  
 {  
 if (n == null)  
 {  
 return 0;  
 }  
  
 if (n.left == null && n.right == null)  
 {  
 return 1;  
 }  
  
 return Leaf(n.left) + Leaf(n.right);  
 }  
  
 public int Even (TreeNode n)  
 {  
 if (n == null)  
 {  
 return 0;  
 }  
  
 int count;  
  
 count = 0;  
  
 if (n.data % 2 == 0)  
 {  
 count = 1;  
 }  
  
 count = count + Even(n.left) + Even(n.right);  
  
 return count;  
 }  
}

Output:

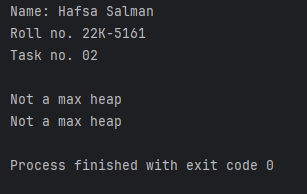


**Task no. 02**

Code:

//Hafsa Salman  
//22K-5161  
//Task no. 02  
  
import java.util.Arrays;  
  
public class Task\_02  
{  
 public static int Max (int[] arr)  
 {  
 int n;  
  
 n = arr.length;  
  
 for (int i=0; i<=(n-2)/2; i++)  
 {  
 if (arr[2\*i+1] > arr[i])  
 {  
 return -1;  
 }  
  
 if (arr[2\*i+2] < n && arr[2\*i+2] > arr[i])  
 {  
 return -1;  
 }  
 }  
  
 return 1;  
 }  
  
 public static int Min (int[] arr)  
 {  
 int n;  
  
 n = arr.length;  
  
 for (int i=0; i<=(n-2)/2; i++)  
 {  
 if (arr[2\*i+1] < arr[i])  
 {  
 return -1;  
 }  
  
 if (arr[2\*i+2] < n && arr[2\*i+2] < arr[i])  
 {  
 return -1;  
 }  
 }  
  
 return 1;  
 }  
  
 public static void main(String[] args)  
 {  
 System.*out*.println("Name: Hafsa Salman");  
 System.*out*.println("Roll no. 22K-5161");  
 System.*out*.println("Task no. 02");  
 System.*out*.println();  
  
 int [] arr = {4, 10, 3, 5, 1};  
  
 int a;  
  
 a = *Max*(arr);  
  
 if (a == 1)  
 {  
 System.*out*.println("Printing Max Heap: ");  
 System.*out*.println();  
  
 System.*out*.println(Arrays.*toString*(arr));  
 }  
  
 else  
 {  
 System.*out*.println("Not a max heap");  
 }  
  
 int b;  
  
 b = *Min*(arr);  
  
 if (b == 1)  
 {  
 System.*out*.println("Printing Min Heap: ");  
 System.*out*.println();  
  
 System.*out*.println(Arrays.*toString*(arr));  
 }  
  
 else  
 {  
 System.*out*.println("Not a max heap");  
 }  
 }  
}

Output:



**Task no. 03**

Code:

//Hafsa Salman  
//22K-5161  
//Task no. 03  
public class Task\_03  
{  
 static int[] *tree* = new int[10];  
  
 static  
 {  
 for (int i = 0; i < *tree*.length; i++)  
 {  
 *tree*[i] = -1;  
 }  
 }  
  
 static void root(int key)  
 {  
 if (*tree*[0] != -1)  
 {  
 System.*out*.println("tree already has a root");  
 }  
  
 else  
 {  
 *tree*[0] = key;  
 }  
 }  
  
 static void setLeft(int key, int parent)  
 {  
 int leftchild;  
  
 leftchild = (parent \* 2) + 1;  
  
 if (leftchild >= *tree*.length)  
 {  
 System.*out*.println("left child index out of bounds");  
 }  
  
 else if (*tree*[parent] == -1)  
 {  
 System.*out*.println("can't set child, no parent found");  
 }  
  
 else  
 {  
 *tree*[leftchild] = key;  
 }  
 }  
  
 static void setRight(int key, int parent)  
 {  
 int rightchild;  
  
 rightchild = (parent \* 2) + 2;  
  
 if (rightchild >= *tree*.length)  
 {  
 System.*out*.println("right child index out of bounds");  
 }  
  
 else if (*tree*[parent] == -1)  
 {  
 System.*out*.println("can't set child, no parent found");  
 }  
  
 else  
 {  
 *tree*[rightchild] = key;  
 }  
 }  
  
 static void insert(int key)  
 {  
 for (int i = 0; i < *tree*.length; i++)  
 {  
 if (*tree*[i] == -1)  
 {  
 *tree*[i] = key;  
  
 return;  
 }  
 }  
  
 System.*out*.println("Tree is full");  
 }  
  
 static void delete(int key)  
 {  
 for (int i = 0; i < *tree*.length; i++)  
 {  
 if (*tree*[i] == key)  
 {  
 *tree*[i] = -1;  
  
 return;  
 }  
 }  
  
 System.*out*.println("key not found");  
 }  
  
 static void printTree()  
 {  
 System.*out*.println();  
  
 for (int i = 0; i < 10; i++)  
 {  
 if (*tree*[i] != -1)  
 {  
 System.*out*.print(*tree*[i]);  
 }  
  
 else  
 {  
 System.*out*.print("-");  
 }  
 }  
  
 System.*out*.println();  
 }  
  
 public static void main(String[] args)  
 {  
 System.*out*.println("Name: Hafsa Salman");  
 System.*out*.println("Roll no. 22K-5161");  
 System.*out*.println("Task no. 03");  
 System.*out*.println();  
  
 *root*(1);  
 *setLeft*(2, 0);  
 *setRight*(3, 0);  
 *setLeft*(4, 1);  
 *setRight*(5, 1);  
 *setRight*(6, 2);  
 *setRight*(7, 2);  
  
 System.*out*.print("Printing Tree: ");  
 *printTree*();  
  
 System.*out*.println("\nDeleting node \"5\"");  
 *delete*(5);  
  
 System.*out*.print("\nPrinting Tree: ");  
 *printTree*();  
 }  
}

Output:

