**BINARY SEARCH TREE**

1 import java.util.\*;  
 2 class Node{  
 3 Node left;  
 4 Node right;  
 5 int data;  
 6 Node(){  
 7 this.data = 0;  
 8 this.left = null;  
 9 this.right = null;  
 10 }  
 11 }  
 12   
 13 public class Binary\_Tree{  
 14 static Node r;  
 15 static int root;  
 16 static int c=0;  
 17 public static void main(String args[]){  
 18 Scanner sc = new Scanner(System.in);  
 19 r = new Node();  
 20 r.data = 10;  
 21 root = r.data;  
 22 while(true){  
 23 System.out.print("\nPress 1 to add elements in Binary Search Tree");  
 24 System.out.print("\nPress 2 to display Pre Order of Binary Search Tree");  
 25 System.out.print("\nPress 3 to display Post Order of Binary Search Tree");  
 26 System.out.print("\nPress 4 to display In Order of Binary Search Tree");  
 27 System.out.print("\nPress 5 to delete element of Binary Search Tree");  
 28 System.out.print("\nPress 6 to exit");  
 29 System.out.print("\nEnter choice :");  
 30 int choice = sc.nextInt();  
 31 sc.nextLine();  
 32 if(choice==1){  
 33 System.out.print("\nEnter the numbers separated by space :");  
 34 String numbers = sc.nextLine();  
 35 String n[] = numbers.split("\\s+");  
 36 int[] num = new int[n.length];  
 37 for(int i = 0 ; i<n.length; i++){  
 38 String temp = n[i];  
 39 num[i] = Integer.parseInt(temp);  
 40 r=insertBst(r,num[i]);  
 41 }  
 42 }   
 43 else if(choice==2){  
 44 System.out.print("\n");  
 45 preorder(r);  
 46 }  
 47 else if(choice==4){  
 48 System.out.print("\n");  
 49 inorder(r);  
 50 }   
 51 else if(choice==3){   
 52 System.out.print("\n");  
 53 postorder(r);  
 54 }  
 55 else if(choice==5){  
 56 System.out.print("\nEnter the node to be deleted :");  
 57 int e = sc.nextInt();  
 58 delete(r,e);  
 59 }  
 60 else if(choice==6){  
 61 System.exit(0);  
 62 }   
 63   
 64 else{  
 65 System.out.print("Invalid input!Enter again");  
 66 }  
 67   
 68   
 69 }  
 70 }  
 71   
 72 public static void preorder(Node r1){  
 73 // System.out.print(" ");  
 74 if(r1!=null){  
 75 System.out.print(" "+r1.data);  
 76 preorder(r1.left);  
 77 preorder(r1.right);  
 78 }  
 79 }  
 80   
 81 public static void postorder(Node r1){  
 82 // System.out.print(" ");  
 83 if(r1!=null){  
 84 postorder(r1.left);  
 85 postorder(r1.right);  
 86 System.out.print(" "+r1.data);  
 87 }  
 88 }  
 89   
 90 public static void inorder(Node r1){  
 91 // System.out.print(" ");  
 92 if(r1!=null){  
 93 inorder(r1.left);  
 94 System.out.print(" "+r1.data);  
 95 inorder(r1.right);  
 96 }  
 97 }  
 98   
 99 public static Node insertBst(Node r,int e){  
100 if(r==null){  
101 Node n = new Node();  
102 n.data = e;  
103 n.left = null;  
104 n.right = null;  
105 r = n;  
106   
107   
108 }  
109 else if(e<=r.data){  
110 r.left = insertBst(r.left,e);  
111   
112   
113 }  
114 else{  
115 r.right = insertBst(r.right,e);  
116 }  
117 return r;  
118   
119 }  
120   
121 public static Node find\_min(Node temp){  
122 if(temp.left!=null){  
123 return find\_min(temp.left);  
124 }  
125 else{  
126 return temp;  
127 }  
128   
129 }  
130   
131 public static void delete(Node temp, int e){  
132 Node n = new Node();  
133   
134 if(c!=0 && temp.left!=null && temp.left.data==e){  
135 if(temp.right!=null){  
136 delete(temp.left,e);  
137 }  
138 else{  
139 temp.left=null;  
140 }   
141 }  
142 else if(temp.left!=null && temp.left.data==e && temp.left.left==null && temp.left.right==null){  
143 temp.left=null;  
144 }  
145 else if(temp.right!=null && temp.right.data==e && temp.right.left==null && temp.right.right==null){  
146 temp.right=null;  
147 }  
148   
149 else {  
150   
151 if(temp==null){  
152 System.out.print("\nElement not found");  
153 }  
154 else if(e<temp.data){  
155 delete(temp.left,e);  
156 }  
157 else if(e>temp.data){  
158 delete(temp.right,e);  
159 }  
160 else if(temp.left!=null && temp.right!=null){  
161 n = find\_min(temp.right);  
162 System.out.print("The deleted node is :"+temp.data);  
163 temp.data = n.data;  
164 if(temp.data==temp.right.data && temp.right.left==null && temp.right.right==null){  
165   
166 temp.right=null;  
167   
168 }  
169 if(temp.right!=null){  
170 c=c+1;  
171 delete(temp.right,temp.data);  
172 }   
173 }  
174 else {   
175 if(temp.left!=null && temp.right==null){  
176 temp.data=temp.left.data;  
177 temp.right = temp.left.right;  
178 temp.left = temp.left.left;  
179 }  
180 else if(temp.right!=null && temp.left==null){  
181 temp.data=temp.right.data;  
182 temp.left = temp.right.left;  
183 temp.right = temp.right.right;  
184 }  
185 else if(temp.left!=null){  
186 temp.data = temp.left.data;  
187 temp.left = null;  
188 }  
189 else{  
190 temp.data = temp.right.data;  
191 temp.right = null;  
192 }  
193   
194 }   
195 }  
196   
197 }  
198   
199   
200 }

**BINAY 19CSU370**

**IOT-A**