**LAB: 12**

**NAME: NOOR FATIMA**

**ROLL NO: 21sw062**

**SECTION: II**

**TASK:01**

Task # 1:Create a Binary Search Tree for the following data.6,8,22,3,7,5,12,10,9,20,35,40,42

class BST {  
  
 private Object root;  
 private BST left,right;  
 private int size;  
 public BST(Object root)  
 {  
 this.root=root;  
 }  
 public BST(Object root,BST left,BST right){  
 this.root=root;  
 this.left=left;  
 this.right=right;  
 }  
 public void add(BST tree,Object data){  
 BST temp=tree;  
 while (true){  
 if(Integer.*parseInt*(temp.root.toString())>Integer.*parseInt*(data.toString())){  
 if(temp.left==null){  
 temp.left=new BST(data);  
 size++;  
 return;  
 }  
 temp=temp.left;  
 }  
 else if(Integer.*parseInt*(temp.root.toString())<Integer.*parseInt*(data.toString())){  
 if(temp.right==null){  
 temp.right=new BST(data);  
 size++;  
 return;  
 }  
 temp=temp.right;  
 }  
 }  
  
 }  
 //In Order Traversing(default case)  
 @Override  
 public String toString(){  
 StringBuilder s=new StringBuilder();  
 if(left!=null)s.append(left).append(" ");  
 s.append(root).append(" ");  
 if(right!=null)s.append(right).append(" ");  
 return s+"";  
 }  
 //pre Order Traversing  
 public String preOrder(){  
 StringBuilder s=new StringBuilder();  
 s.append(root).append(" ");  
 if(left!=null)s.append(left.preOrder()).append(" ");  
 if(right!=null)s.append(right.preOrder()).append(" ");  
 return s+"";  
 }  
  
 //Post Order Traversing  
 public String postOrder(){  
 StringBuilder s=new StringBuilder();  
 if(left!=null)s.append(left.postOrder()).append(" ");  
 if(right!=null)s.append(right.postOrder()).append(" ");  
 s.append(root).append(" ");  
 return s+"";  
 }  
  
}

class LAb12{  
 public static void main(String[] args) {  
 //6,8,22,3,7,5,12,10,9,20,35,40,42  
 BST tree=new BST(6);  
 tree.add(tree,8);  
 tree.add(tree,22);  
 tree.add(tree,3);  
 tree.add(tree,7);  
 tree.add(tree,5);  
 tree.add(tree,12);  
 tree.add(tree,10);  
 tree.add(tree,9);  
 tree.add(tree,20);  
 tree.add(tree,35);  
 tree.add(tree,40);  
 tree.add(tree,42);  
 System.*out*.println("In order: "+tree);  
 System.*out*.println("Pre Order: "+tree.preOrder());  
 System.*out*.println("Post Order: "+tree.postOrder());  
  
  
 }  
 }

**Output:**

"C:\Program Files\Java\jdk-17.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1\lib\idea\_rt.jar=50762:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\hp\IdeaProjects\DSA\_ALL\_LABS\out\production\DSA\_ALL\_LABS LAb12  
 In order: 3 5 6 7 8 9 10 12 20 22 35 40 42  
 Pre Order: 6 3 5 8 7 22 12 10 9 20 35 40 42  
 Post Order: 5 3 7 9 10 20 12 42 40 35 22 8 6  
  
 Process finished with exit code 0

**Task:02**

Task # 2:Create a Binary Search Tree for the following data.A B C D E F G H I J K L M N O P

class BST1 {  
  
  
 private Object root;  
 private BST1 left,right;  
 private int size;  
 public BST1(Object root)  
 {  
 this.root=root;  
 }  
 public BST1(Object root, BST1 left, BST1 right){  
 this.root=root;  
 this.left=left;  
 this.right=right;  
 }  
 public void add(BST1 tree,Object data){  
 BST1 temp=tree;  
 while (true){  
 if(temp.root.toString().compareTo(data.toString())>0){  
 if(temp.left==null){  
 temp.left=new BST1(data);  
 size++;  
 return;  
 }  
 temp=temp.left;  
 }  
 else if(temp.root.toString().compareTo(data.toString())<0){  
 if(temp.right==null){  
 temp.right=new BST1(data);  
 size++;  
 return;  
 }  
 temp=temp.right;  
 }  
 }  
  
 }  
 //In Order Traversing(default case)  
 @Override  
 public String toString(){  
 StringBuilder s=new StringBuilder();  
 if(left!=null)s.append(left).append(" ");  
 s.append(root).append(" ");  
 if(right!=null)s.append(right).append(" ");  
 return s+"";  
 }  
 //pre Order Traversing  
 public String preOrder(){  
 StringBuilder s=new StringBuilder();  
 s.append(root).append(" ");  
 if(left!=null)s.append(left.preOrder()).append(" ");  
 if(right!=null)s.append(right.preOrder()).append(" ");  
 return s+"";  
 }  
  
 //Post Order Traversing  
 public String postOrder(){  
 StringBuilder s=new StringBuilder();  
 if(left!=null)s.append(left.postOrder()).append(" ");  
 if(right!=null)s.append(right.postOrder()).append(" ");  
 s.append(root).append(" ");  
 return s+"";  
 }  
  
}  
class LAb12{

public static void main(String[] args) {  
 BST1 tree=new BST1("A");  
 tree.add(tree,"B");  
 tree.add(tree,"C");  
 tree.add(tree,"D");  
 tree.add(tree,"E");  
 tree.add(tree,"F");  
 tree.add(tree,"G");  
 tree.add(tree,"H");  
 tree.add(tree,"I");  
 tree.add(tree,"J");  
 tree.add(tree,"K");  
 tree.add(tree,"L");  
 tree.add(tree,"M");  
 tree.add(tree,"N");  
 tree.add(tree,"O");  
 tree.add(tree,"P");  
 System.*out*.println("In order: "+tree);  
 System.*out*.println("Pre Order: "+tree.preOrder());  
 System.*out*.println("Post Order: "+tree.postOrder());  
  
}  
}

**OUTPUT:**

"C:\Program Files\Java\jdk-17.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1\lib\idea\_rt.jar=50787:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.1\bin" -Dfile.encoding=UTF-8 -classpath C:\Users\hp\IdeaProjects\DSA\_ALL\_LABS\out\production\DSA\_ALL\_LABS LAb12  
 In order: A B C D E F G H I J K L M N O P  
 Pre Order: A B C D E F G H I J K L M N O P  
 Post Order: P O N M L K J I H G F E D C B A  
  
 Process finished with exit code 0