**2. a)** Preorder: A, B, C, D, E

Inorder: E,D,C,B,A

Postorder: E,D,C,B,A

**b)** Preorder: A,B,D,H,I,E,C,F,G

Inorder: H,D,I,B,E,A,F,C,G

Postorder: H,I,D,E,B,F,G,C,A

**3.**

* Function **wrPost(tree)** is

Objects: tree of *BinaryTree* class, stacks of *Stack* class

Pseudocode:

*Stack1=Stack()*

*Stack2=Stack()*

*Stack1.push(tree)*

*While Stack1 is not empty:*

*tree = first element of Stack1*

*push tree to Stack2*

*if left child of tree is not empty:*

*push it to Stack1*

*if right child of tree is not empty:*

*push it to Stack1*

*print all trees from Stack2*

* Function **wrPre(tree)** is

Objects: tree of *BinaryTree* class, stacks of *Stack* class

Pseudocode:

*Stack = Stack()*

*Push tree to Stack*

*while Stack is not empty:*

*tree = first element of Stack*

*print root value of the tree*

*if tree has right child:*

*push it to Stack*

*if tree has left child:*

*push it to Stack*

Both functions are implemented using binary tree represented as a class. Examples of functions working are implemented using the tree from problem 2.