(25) Given preorder and inorder strings, print postorder.

(Amazon onsite)

$O(n^2)$

```
void printPost(String preorder, String inOrder, int inStart,
    int inEnd, int preIndex
{
    if(inStart > inEnd) return;
        // find rootIdx in inorder.
    int inIndex = find(inOrder, inStart, inEnd, preOrder.charAt(
                                                preIndex+1));
    // find left side
    printPost(preOrder, inOrder, inStart, inIndex-1, preInd+1);
    // Go right side.
    printPost(preOrder, inOrder, inStart+1, inIndex, preIdx+1);
    // print
    System.out.println(inOrder.charAt(inIndex));
}
```

algo:

(i) elements of preorder are roots;

(ii) find idx of root in inorder.
All elements on left is left subtree. Keep searching until leftmost element is found.
print it;

(iii) Repeat for right subtree.

```
int find(String inOrder, int start, int end, char c){
    int i=0;
    for(i = start; i < end; i++)
        if(inOrder.charAt(i) == c) return i;

    return i;
}
```

$O(n^2)$: for every char in preorder, we are searching for it in inorder.

$O(n)$: use hashmap for inorder to store char and its index.