Stacks and Queues

a. Write Java code for `public E remove()` for a generic queue implemented with a “circular” array as described in the text and lecture. Don’t forget about the `nextIndex` method (you can assume that you have this available to you).

b. Use a stack to evaluate the following postfix expression. Show your work so that I can see you are using a stack. You will not receive full credit without appropriate stack pictures. You may not use calculators.

```
7 3 + 4 5 * 4 2 + - 3 * 6 / +
```
a. What do you need to do to copyright material you put on your web pages?

b. Give two requirements for obtaining a patent on a software application.

Recursion
For this question, you will write two versions of a function: one iterative and one recursive. The functions will take in two integer parameters named little and big. The function prototype for both functions is:

```java
void pattern(int little, int big)
```

When little is 1 and big is 5, the pattern is 5 4 3 2 1 2 3 4 5 with a space between each value and no newline.
When little is 17 and big is 20, the pattern is 20 19 18 17 18 19 20

a. Write an iterative version of the function using a loop(s). Think simple and straightforward. This is not a trick.

b. Write a recursive version of the same function. It should use NO (absolutely no) loops. What is the base case?

(1) What is the output of the following code if it is called with mystery(4)?

```java
public static void mystery(int n) {
    if (n > 0) {
        System.out.print(n + " ");
        mystery(n-3);
        mystery(n-2);
        System.out.print(n + " ");
    }
}
```
For the binary tree below neatly write your answers to the following:

```
  42
  / \  
17   58
  / \  
12   29   72
     / \  
  68   99
```

a. Is it a binary search tree? Why or why not?

b. Show the result of an inorder traversal of this tree as shown.

c. Show the result of a preorder traversal of this tree as shown.

d. Show the result of a postorder traversal of this tree as shown.

e. Write the recursive code for doing a preorder traversal of a binary tree that stores characters. You can assume that you are given the root node as `node`, as shown below:

```java
public void printPreorder(CharBTNode node)
```

f. The text suggests that a binary search tree would be a more efficient way to represent a Bag than the various ways we studied in class especially when searching. Why?