Write a recursive method named clear that removes all elements from a Queue. Do NOT use any loops. You may assume that the instance of Queue implements the following interface and that the methods are as have been discussed in class.

```java
public interface QueueInterface<E> {
    public boolean empty();
    public E offer(E e);
    public E poll();
    public int size();
    public E peek();
}
```

The clear method should be generic and should accept an object that implements QueueInterface as it only input argument. (clear is NOT to be written within the class that implements Queue.)

Suppose that you are given a class ShortStack that has only the methods

```java
private class ShortStack<V> {
    public V pop();
    public void push(V v);
}
```

These two methods follow the documentation of Stack discussed in class.

You are then asked to implement a method with the following documentation and signature.

```java
/**
 * Determine the number of items contained in the provided instance of ShortStack.
 * This method may modify the provided instance while it is running. However, when the method
 * is complete the provided stack will have exactly the same contents in exactly the same order
 * as prior to the execution of the method.
 * @param sStack the stack whose count is to be determined.
 * @return the number of items in the provided stack
 */
public int ssSize(ShortStack<E> sStack) { return -1; }
```

Provide the implementation of this method. You may use any additional data structures you would find useful.
Problem 3: You are given the following array.

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>8</td>
<td>13</td>
<td>21</td>
<td>34</td>
<td>55</td>
<td>89</td>
<td>144</td>
<td>233</td>
<td>377</td>
<td>610</td>
<td>987</td>
<td>1597</td>
<td>2584</td>
<td>4181</td>
<td>6765</td>
</tr>
</tbody>
</table>

Show the sequence of recursive function calls for the searchUtil function discussed in class when searching for 34 and 4182.

Problem 4. You are given an array list. Write a recursive function that returns an array list in reversed order from the provided array list. NO LOOPS

Problem 5: You are given an array of integers. Write a method that finds the location of smallest element in the array.

Problem 6. Use the function from problem 5 in a function to, given an array, returns an array that is one shorter than the given array and which is missing the smallest element. What is the computational complexity of this function.

Problem 7: You are given a 3-d array of integers. Each of the dimensions of the array is the same size. (It is a cube). Write a method to compute the sum of the positive items in the array. What is the computational complexity of this method?