Problem 1

Suppose that you have a singly linked list of ints without header, with the following class definition:

```java
class IntList {
    int value;
    next IntList;
}
```

Write a recursive, owned method `l.sumOfRest()` which destructively replaces the value at each node in the list with the sum of the values of the list starting at that point. The method `sumOfList()` should return the sum of the values in the original list. For instance, if `l` is initially the list `[3,2,1]`, and you call `l.sumOfList()` then `l` becomes the list `[6=(3+2+1), 3(=2+1), 1]`. If `l` is initially the list `[5,2,1,6,8]` and you call `l.sumOfList()` then `l` becomes the list `[22, 17, 15, 14, 8]`.

Problem 2:

Give the preorder, postorder, and breadth-first order of the following tree.

![Tree Diagram]

Problem 3:

A. Draw the expression tree corresponding to the following expression.

\[ 1 + \frac{(x! + 2)^2}{x^2 + 4} \]

Treat the exponential function \( p^q \) as a two-place function \( \uparrow(p, q) \), and the factorial function \( p! \) as a unary function \(!p\).

B. Write the expression in prefix notation.
Problem 4:

Suppose that you have a binary tree of the following structure:

A. Put the numbers [2,5,9,12,16,19,22,25,28] into this tree so that it forms a binary search tree.
B. Show the final result if you execute add(20) and delete(16) in sequence on this tree.

Problem 5:

Suppose you have a hash table class with methods H.put(T key,U value), and H.get(T key). Here T and U are generic classes. Also, suppose you have a generic singly linked list class defined as follows

```java
class LL<T> {
    public T value;
    public LL<T> next;.
}
```

Write in Java a static method boolean subset(LL<String> x, LL<String> y). The input parameters x and y are unordered linked lists with header, which may have repeated element. The method should return true if every value in x occurs in y; how many times it occurs is unimportant. For instance if x is the list of Strings ["A", "B", "C", "B", "A"] and y is the list ["C", "B", "D", "C", "A"] then subset(x,y) should return true.

You should write the code so that it takes advantage of the hash table to work efficiently. Specifically, you will get full credit if the code runs in time proportional to the sum of the lengths of x and y. If it runs in time proportional to |x| · |y| then you will get 2/3 credit (for that, you don’t need the hash table).