The goal of this project is to understand the recursive formulas for counting combinatorial objects to the extent that you can adapt those recursive strategies to actually generate those objects.

You will work in groups of 2 or 3s. Make sure that the names of all group members appear as authors in the code. When finished, appoint a group leader who will submit the assignment by attachment to email, with a cc to the other members of the group.

This is due Saturday at the end of Week 5.

1. **Partitions of a set**

Use your understanding of the recursive method for counting partitions of a set of size $n$ that has $k$ parts to generate a list of all such partitions.

Do this by writing a method

```java
/**
 * returns list of all partitions of a set 1 .. n
 * that have k parts
 * @param n
 * @param k
 * @return
 */
static LinkedList<Partition> partitionsList(int n, int k)
```

Here we are declaring a class

```java
class Partition extends LinkedList<TreeSet>
{
}
```

to make the code a little bit more readable.

Test your work using this main method:

```java
public static void main(String[] args)
{
    Scanner sc = new Scanner(System.in);
    int n, k;

    while (true)
    {
        System.out.print("Enter n and k with n >= k: ");
```
```java
n = sc.nextInt();
k = sc.nextInt();

if (n <= 0 || k > n || k == 0) return;

List<Partition> partitions = partitionsList(n, k);
partitions.forEach(System.out::println);
```
Test your work with the following main method:

```java
public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int n, k;

    while (true) {
        System.out.print("Enter n and k with n >= k: ");
        n = sc.nextInt();
        k = sc.nextInt();

        if (n <= 0 || k > n || k == 0) return;

        List<List<Integer>> partitions = numPartitions(n, k);
        partitions.forEach(System.out::println);
    }
}
```

Here is a sample run:

Enter n and k with n >= k: 5 1
[1, 1, 1, 1, 1]
Enter n and k with n >= k: 5 2
[1, 2, 2]
[1, 1, 1, 2]
[1, 1, 1, 1, 1]
Enter n and k with n >= k: 5 3
[1, 1, 3]
[2, 3]
[1, 2, 2]
[1, 1, 1, 2]
[1, 1, 1, 1, 1]
Enter n and k with n >= k: 5 4
[1, 4]
[1, 1, 3]
[2, 3]
[1, 2, 2]
[1, 1, 1, 2]
[1, 1, 1, 1, 1]
Enter n and k with n >= k: 5 5
[1, 4]
[1, 1, 3]
[2, 3]
[1, 2, 2]
[1, 1, 1, 2]
[1, 1, 1, 1, 1]
[5]
Enter n and k with n >= k: 0 0