Задачи для курса Java-basic - рекурсия

1. Given an array of ints, is it possible to choose a group of some of the ints, such that the group sums to the given target? This is a classic backtracking recursion problem. Once you understand the recursive backtracking strategy in this problem, you can use the same pattern for many problems to search a space of choices. Rather than looking at the whole array, our convention is to consider the part of the array starting at index start and continuing to the end of the array. The caller can specify the whole array simply by passing start as 0. No loops are needed -- the recursive calls progress down the array.

```
groupSum(0, [2, 4, 8], 10) → true
groupSum(0, [2, 4, 8], 14) → true
groupSum(0, [2, 4, 8], 9) → false
```

2. Given an array of ints, is it possible to choose a group of some of the ints, such that the group sums to the given target with these additional constraints: all multiples of 5 in the array must be included in the group. If the value immediately following a multiple of 5 is 1, it must not be chosen. (No loops needed.)

```
   groupSum5(0, [2, 5, 10, 4], 19) → true
   groupSum5(0, [2, 5, 10, 4], 17) → true
   groupSum5(0, [2, 5, 10, 4], 12) → false
```

3. Given an array of ints, is it possible to divide the ints into two groups, so that the sum of one group is a multiple of 10, and the sum of the other group is odd. Every int must be in one group or the other. Write a recursive helper method that takes whatever arguments you like, and make the initial call to your recursive helper from splitOdd10(). (No loops needed.)

```
splitOdd10([5, 5, 5]) → true
splitOdd10([5, 5, 6]) → false
splitOdd10([5, 5, 6, 1]) → true
```

4. Given an array of ints, is it possible to choose a group of some of the ints, beginning at the start index, such that the group sums to the given target? However, with the additional constraint that all 6's must be chosen. (No loops needed.)

```
groupSum6(0, [5, 6, 2], 8) → true
groupSum6(0, [5, 6, 2], 9) → false
groupSum6(0, [5, 6, 2], 7) → false
```

5. Given an array of ints, is it possible to choose a group of some of the ints, such that the group sums to the given target, with this additional constraint: if there are numbers in the array that are adjacent and the identical value, they must either all be chosen, or none of them chosen. For example, with the array {1, 2, 2, 2, 5, 2}, either all three 2's in the middle must be chosen or not, all as a group. (one loop can be used to find the extent of the identical values).

```
groupSumClump(0, [2, 4, 8], 10) → true
groupSumClump(0, [1, 2, 4, 8, 1], 14) → true
groupSumClump(0, [2, 4, 4, 8], 14) → false
```

6. Given an array of ints, is it possible to divide the ints into two groups, so that the sum of the two groups is the same, with these constraints: all the values that are multiple of 5 must be in one group, and all the values that are a multiple of 3 (and not a multiple of 5) must be in the other. (No loops needed.)

```
split53([1, 1]) → true
split53([1, 1, 1]) → false
split53([2, 4, 2]) → true
```

7. Given an array of ints, is it possible to choose a group of some of the ints, such that the group sums to the given target with this additional constraint: If a value in the array is chosen to be in the group, the value immediately following it in the array must not be chosen. (No loops needed.)

```
groupNoAdj(0, [2, 5, 10, 4], 12) → true
groupNoAdj(0, [2, 5, 10, 4], 14) → false
groupNoAdj(0, [2, 5, 10, 4], 7) → false
```

8. Given an array of ints, is it possible to divide the ints into two groups, so that the sums of the two groups are the same. Every int must be in one group or the other. Write a recursive helper method that takes whatever arguments you like, and make the initial call to your recursive helper from splitArray(). (No loops needed.)

```
splitArray([2, 2]) → true
splitArray([2, 3]) → false
splitArray([5, 2, 3]) → true
```