题目描述：

给定 M(0<M<=30)个字符（a-z），从中取出任意字符（每个字符只能用一次）拼接成长度为 N(0<N<=5)的字符串，要求相同的字符不能相邻，计算出给定的字符列表能拼接出多少种满足条件的字符串，输入非法或者无法拼接出满足条件的字符串则返回 0。

输入描述：

给定的字符列表和结果字符串长度，中间使用空格（" "）拼接

输出描述：

满足条件的字符串个数

示例 1

输入：

```
abc 1
```

输出：

```
3
```

说明：

给定的字符为 a,b,c，结果字符串长度为 1，可以拼接成 a,b,c，共 3 种

示例 2

输入：

```
dde 2
```

输出：

```
2
```

说明：

给定的字符为 dde，结果字符串长度为 2，可以拼接成 de,ed，共 2 种

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashSet;
import java.util.Scanner;

public class Main {
	static int num;
	static int sum;

	public static void main(String[] args) {
		Scanner scanner = new Scanner(System.in);
		String s = scanner.nextLine();
		try {
			num=Integer.valueOf(s.substring(s.length()-1));
			s=s.substring(0,s.length()-2);
		} catch (NumberFormatException e) {
			System.out.println(0);
			return;
		}


		char[] chars = s.toCharArray();
		Arrays.sort(chars);
		ArrayList<Integer> integers = new ArrayList<>();
		int[] bo = new int[chars.length];
		sum = 0;
		dfs123(chars, integers, bo);
		System.out.println(sum);


	}

	private static void dfs123(char[] chars, ArrayList<Integer> integers, int[] bo) {
		if (integers.size() == num) {
			sum++;

			return;

		}
		for (int i = 0; i < chars.length; i++) {
			if (bo[i] == 0) {
				if (integers.size() == 0 || chars[i] != chars[integers.get(integers.size() - 1)]) {
					if (i > 0 && chars[i] == chars[i - 1] && bo[i - 1] == 0) {
						continue;
```

```
                    }
                    bo[i] = 1;
                    integers.add(i);

                    dfs123(chars,    integers, bo);
                    bo[i] = 0;
                    integers.remove(integers.size() - 1);

                }
            }

        }
        return;


    }

}
```