请20分钟内完成以下题目（以纸笔或者文本形式完成，请勿使用IDE等开发工具）

1、请写出实际代码（可以使用任意熟悉的编码语言）

2、需要考虑时间、空间复杂度

百词斩的Lily非常喜欢旅游，经常和他老婆自驾出游。但是Lily是一个非常讲求性价比的程序员，所以每次在外面吃饭的时候都要控制价格。今年国庆Lily和他老婆出游，Lily规定每顿饭的价格上限为n，他老婆想点k个菜，餐厅提供m个菜。请问，在不超过Lily价格上限的情况下，点k个菜有多少种点法（同一种菜只能点一份）。

输入：

1、第一行为价格上限n，想点菜的个数k，以及菜品个数m

2、接下来m行，每行一个菜品价格

3、以上输入均为正整数

输出：

1、输出为一行，代表点菜的方案数

Sample Input:

10 2 4

10

3

5

12

Sample Output:

1

（Sample说明：因为需要点两个菜，且总价不超过10，所以只能有第二和第三个菜的组合方案。其他方案加起来都会超过10）

题解：

package contest.baichizhan;  
  
import java.util.ArrayList;  
import java.util.Arrays;  
import java.util.List;  
import java.util.Scanner;  
  
*/\*\*  
 \* Created by mercop on 2017/9/1.  
 \*/*public class bcz01 {  
  
 public static void main(String[] args) {  
 Scanner sc = new Scanner(System.*in*);  
 int n = sc.nextInt();  
 int k = sc.nextInt();  
 int m = sc.nextInt();  
 int[] nums= new int[m];  
 for(int i = 0; i < m;i ++){  
 nums[i] = sc.nextInt();  
 }  
 System.*out*.println(*getCountOfOptions*(nums,n,k,m));  
 }  
 public static int getCountOfOptions(int[] nums,int n,int k,int m){  
 if(n <= 0 || k <= 0 || k > m || m < 0) return 0;  
 Arrays.*sort*(nums);  
 *getAllCombinations*(nums,0,k,n,new ArrayList<>());  
 return *results*.size();  
 }  
  
  
 static List<List<Integer>> *results* = new ArrayList<>();  
 public static void getAllCombinations(int[] nums, int index, int k ,int restBudget, List<Integer> res){  
  
 if(k == res.size()){  
 if(restBudget >= 0) {  
 *results*.add(res);  
 }  
 return;  
 }  
 if(restBudget < 0 || nums.length == index ) return ;  
  
 //剪枝操作，从[index, len)到[index,len - (k - res.size()) + 1)  
 //一些分支是无效的，例如最开始index 取最后一个节点，因为我们的总是从index开始选取，则无法在取到第二个节点  
 for(int i = index; i < nums.length - (k - res.size()) + 1; i ++){  
 if(nums[i] <= restBudget){  
 res.add(nums[i]);  
 *getAllCombinations*(nums,i + 1,k ,restBudget - nums[i],new ArrayList<>(res));  
 res.remove(res.size() -1);  
 }  
 //减枝操作，在保证nums为有序数组情况下，如果当前元素大于剩余的预算，则说明后面的都不可能满足条件直接跳出  
 else break;  
 }  
 }  
  
}

复杂的分析：该函数递归深度为k，未做剪枝每一层结点个数为m的时间复杂度为O(k \* m)。