|  |  |
| --- | --- |
| **Top Batsmen**  **Solved**  Problem code: BESTBATS | * [SUBMIT](https://www.codechef.com/submit/BESTBATS) * [MY SUBMISSIONS](https://www.codechef.com/status/BESTBATS,nacho0monllor) * [ALL SUBMISSIONS](https://www.codechef.com/status/BESTBATS) |

**All submissions for this problem are available.**

A cricket team consists of 11 players and some are good at batting, others are good at bowling and some of them are good at both batting and bowling. The batting coach wants to select exactly **K** players having maximum possible sum of scores. Given the batting score of each of the 11 players, find the number of ways in which we can select exactly **K** players such that the sum of their scores is the maximum possible. Two ways are different if there is a player who is selected in one of them is not in the other. See explanation of sample cases for more clarity.

**Input**

First line contains T, number of test cases ( 1 ≤ T ≤ 100 ). T cases follow, each having 2 lines. First line of each case contains scores of 11 players ( 1 ≤ score ≤ 100 ) and the second line contains K (1 ≤ K ≤ 11)

**Output**

For each test case, output the answer in a new line.

**Example**

**Input:**

2

1 2 3 4 5 6 7 8 9 10 11

3

2 5 1 2 4 1 6 5 2 2 1

6

**Output:**

1

6

**Explanation**:  
Case 1 : Maximum possible sum of scores = 11 + 10 + 9 = 30 and can be achieved only by selecting the last 3 players. Only one possible way.  
  
Case 2 : Maximum possible sum of scores = 6 + 5 + 5 + 4 + 2 + 2 = 24 and considering the players as p1 p2 p3 ... p11 in that order, the ones with maximum possible sum of scores is as follows  
{p1, p2, p4, p5, p7, p8 }  
{p10, p2, p4, p5, p7, p8 }  
{p1, p2, p10, p5, p7, p8 }  
{p9, p2, p4, p5, p7, p8 }  
{p1, p2, p9, p5, p7, p8 }  
{p10, p2, p9, p5, p7, p8 }

<https://www.codechef.com/problems/BESTBATS>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication2

{

class Program

{

static List<List<int>> Powerset(int[] nums)

{

List<List<int>> ps = new List<List<int>>();

ps.Add(new List<int>()); // add the empty set

// for every item in the original list

foreach (int item in nums)

{

List<List<int>> newPs = new List<List<int>>();

foreach (List<int> subset in ps)

{

// copy all of the current powerset's subsets

newPs.Add(subset);

// plus the subsets appended with the current item

List<int> newSubset = new List<int>(subset);

newSubset.Add(item);

newPs.Add(newSubset);

}

// powerset is now powerset of list.subList(0, list.indexOf(item)+1)

ps = newPs;

}

return ps;

}

static void Main(string[] args)

{

int t = int.Parse(Console.ReadLine());

while (t-- > 0)

{

string[] input = Console.ReadLine().Trim().Split(' ');

int[] puntos = Array.ConvertAll(input, e => int.Parse(e));

int k = int.Parse(Console.ReadLine());

List<List<int>> ps = Powerset(puntos);

int maxSum = 0;

int ans = 0;

foreach (List<int> lista in ps)

{

if (lista.Count == k)

{

int sum = 0;

foreach (int p in lista)

{

sum += p;

}

if (sum > maxSum)

{

ans = 1;

maxSum = sum;

}

else if (sum == maxSum)

{

ans++;

}

}

}

Console.WriteLine(ans);

}

Console.ReadLine();

}

}

}