Count all subsequences having product less than K

> 给定一个数组，找出满足条件的子集的数量——子集中元素的乘积小于K

dp[i][j]表示arr前j个元素的乘积小于i的子序列的个数，对于dp[i][j]，有：

1. dp[i][j] = dp[i][j - 1] （使用前j - 1个元素且不使用第 j - 1个元素）
2. 如果 arr[j - 1] ≤ i，则 dp[i][j] = dp[i][j] + dp[i / arr[j - 1]][j - 1] （使用了第j个元素(arr[j- 1])，所以前j - 1个元素的乘积必须小于 i / arr[j - 1]）

```
#include <bits/stdc++.h>

using namespace std;

int countSubSeq(vector<int> &arr, int &N, int &K) {
    sort(arr.begin(), arr.end());
    vector<vector<int>> dp(K + 1, vector<int>(N + 1));
    // 注意下标 dp[i][j] = count(product(arr[0~j - 1]) < i)
    for (int i = 1; i <= K; ++i) {
        for (int j = 1; j <= N; ++j) {
            // 不使用第j个元素
            dp[i][j] = dp[i][j - 1];
            if (arr[j - 1] <= i) {
                // 使用了第j个元素
                dp[i][j] += dp[i / arr[j - 1]][j - 1] + 1;
            }
        }
    }
    return dp[K][N];
}

int main() {
    int T;
    scanf("%d", &T);
    while (T--) {
        int N, num, K;
        scanf("%d %d", &N, &K);
        vector<int> arr;
        for (int i = 0; i < N; ++i) {
            scanf("%d", &num);
            arr.push_back(num);
        }
        printf("%d\n", countSubSeq(arr, N, K));
    }
}
```