Coin Change

回溯写法

该方法作为备选项，易超时

```
#include <iostream>
#include <vector>
#include <stdio.h>
#include <climits>

using namespace std;

void getWays(vector<int>& coin, int N, int& ways, int cur) {
    if (N < 0) return;
    if (N == 0) {
        ways++;
        return;
    }
    else {
        for (int i = cur; i < coin.size(); ++i) {
            N -= coin[i];
            getWays(coin, N, ways, i);
            N += coin[i];
        }
    }
}


int main() {
    int T;
    scanf("%d", &T);
    while (T--) {
        unsigned int coinSize;
        int N;
        scanf("%d", &coinSize);
        vector<int> coin;
        for (int i = 0; i < coinSize; ++i) {
            int coinType;
            scanf("%d", &coinType);
            coin.push_back(coinType);
        }
        scanf("%d", &N);
        vector<vector<int>> path;
        vector<int> curPath;
        int ways = 0;
        getWays(coin, N, ways, 0);
        printf("%d\n", ways);
    }
    return 0;
}
```

递归

可以把整个过程划分为两部分（从后往前枚举硬币）：

1. count(s, m - 1, n) 不包含第m个硬币的解决方案
2. count(s, m, n - Sm) 包含第m个硬币的解决方案

该方案存在大量重复计算

```
C() --> count()
                             C({1,2,3}, 5)
                           /             \
                         /                 \
             C({1,2,3}, 2)                 C({1,2}, 5)
            /       \                      /      \
           /         \                    /         \
C({1,2,3}, -1)  C({1,2}, 2)        C({1,2}, 3)    C({1}, 5)
               /    \             /     \           /     \
             /       \           /       \         /        \
    C({1,2},0)  C({1},2)   C({1,2},1) C({1},3)    C({1}, 4)  C({}, 5)
                   / \     / \        /\         /     \
                  /   \   /   \     /   \       /       \
                .      .  .     .   .     .   C({1}, 3) C({}, 4)
                                               / \
                                              /   \
                                             .      .
```

```
C() --> count()
                             C({1,2,3}, 5)
                           /             \
                         /                 \
             C({1,2,3}, 2)                 C({1,2}, 5)
            /       \                      /      \
           /         \                    /         \
C({1,2,3}, -1)  C({1,2}, 2)        C({1,2}, 3)    C({1}, 5)
               /    \             /     \           /     \
             /       \           /       \         /        \
    C({1,2},0)  C({1},2)   C({1,2},1) C({1},3)    C({1}, 4)  C({}, 5)
                   / \     / \        /\         /     \
                  /   \   /   \     /   \       /       \
                .      .  .     .   .     .   C({1}, 3) C({}, 4)
                                               / \
                                              /   \
                                             .      .
```

```cpp
#include <iostream>
#include <vector>
#include <stdio.h>
#include <climits>

using namespace std;

int countWays(vector<int>& coin, int coinSize, int N) {
    if (N == 0) return 1;
    if (N < 0) return 0;
    if (coinSize <= 0 && N >= 1) return 0;
    return countWays(coin, coinSize - 1, N) + countWays(coin, coinSize, N - coin[coinSize - 1]);
}

int main() {
    int T;
    scanf("%d", &T);
    while (T--) {
        unsigned int coinSize;
        int N;
        scanf("%d", &coinSize);
        vector<int> coin;
        for (int i = 0; i < coinSize; ++i) {
            int coinType;
            scanf("%d", &coinType);
            coin.push_back(coinType);
        }
        scanf("%d", &N);
        printf("%d\n", countWays(coin, coinSize, N));
    }
    return 0;
}
```

DP

```
#include <iostream>
#include <vector>
#include <stdio.h>
#include <climits>

using namespace std;

// dp[i][j] 表示 i cents 使用第j种硬币的解决方案

// dp[i][j] = dp[i - s[j]][j] + dp[i][j - 1]
// dp[i - s[j]][j] 表示 使用第 j 种硬币后 i - s[j] cents的解决方案
// dp[i][j - 1] 表示i cents不使用第j种硬币的解决方案
int countWays(vector<int>& coin, int m, int n) {
    vector<vector<int>> dp(n + 1, vector<int>(m));
    for (int i = 0; i < m; ++i)
        dp[0][i] = 1;
    // 枚举每种cents的方案
    for (int i = 1; i < n + 1; ++i) {
        // 枚举每一种硬币
        for (int j = 0; j < m; ++j) {
            int s1 = i - coin[j] >= 0 ? dp[i - coin[j]][j] : 0;
            int s2 = j >= 1 ? dp[i][j - 1] : 0;
            dp[i][j] = s1 + s2;
        }
    }
    return dp[n][m - 1];
}

int main() {
    int T;
    scanf("%d", &T);
    while (T--) {
        unsigned int coinSize;
        int N;
        scanf("%d", &coinSize);
        vector<int> coin;
        for (int i = 0; i < coinSize; ++i) {
            int coinType;
            scanf("%d", &coinType);
            coin.push_back(coinType);
        }
        scanf("%d", &N);
        printf("%d\n", countWays(coin, coinSize, N));
    }
    return 0;
}
```