**Number of Coins**

**Medium**

Given a value **V** and array **coins[]** of size **M**, the task is to make the change for **V** cents, given that you have an infinite supply of each of coins{coins1, coins2, ..., coinsm} valued coins. Find the minimum number of coins to make the change. If not possible to make change then return -1.

**Example 1:**

**Input**: V = 30, M = 3, coins[] = {25, 10, 5}

**Output:** 2

**Explanation**: Use one 25 cent coin

and one 5 cent coin

**Example 2:**

**Input**: V = 11, M = 4,coins[] = {9, 6, 5, 1}

**Output:** 2

**Explanation**: Use one 6 cent coin

and one 5 cent coin

**Expected Time Complexity:** O(V\*M)  
**Expected Auxiliary Space:** O(V)  
  
**Constraints:**  
1 ≤ **V\*M** ≤ 106  
All array elements are distinct

**Company Tags**

[**Paytm**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Paytm) [**Morgan Stanley**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Morgan%20Stanley) [**Accolite**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Accolite) [**Amazon**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Amazon) [**Microsoft**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Microsoft) [**Samsung**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Samsung) [**Snapdeal**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Snapdeal) [**Oracle**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Oracle) [**Visa**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Visa) [**Google**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Google) [**Synopsys**](https://practice.geeksforgeeks.org/explore/?company%5b%5d=Synopsys)

**Topic Tags**

[**Dynamic Programming**](https://practice.geeksforgeeks.org/explore/?category%5b%5d=Dynamic%20Programming) [**Algorithms**](https://practice.geeksforgeeks.org/explore/?category%5b%5d=Algorithms)

//{ Driver Code Starts

//Initial Template for Java

import java.io.\*;

import java.util.\*;

class CodingMaxima

{

public static void main(String args[])

{

Scanner sc = new Scanner(System.in);

int t = sc.nextInt();

while(t-->0)

{

int v = sc.nextInt();

int m = sc.nextInt();

int coins[] = new int[m];

for(int i = 0;i<m;i++)

coins[i] = sc.nextInt();

Solution ob = new Solution();

System.out.println(ob.minCoins(coins, m, v));

}

}

}

// } Driver Code Ends

class Solution{

public int minCoins(int coins[], int M, int V)

{

int[] dp=new int[V+1];

Arrays.fill(dp, V+1);

dp[0]=0;

for(int coin : coins){

for(int i=coin ;i<=V;i++){

dp[i]=Math.min(dp[i], dp[i-coin]+1);

}

}

return dp[V]<=V ? dp[V]: -1;

}

}