# Vending Machine Change

A vending machine has products that can be paid with coins. A zero indexed array C consisting of N integers representing the different coins that the machine accepts is given.

The goal is to find the minimum amount of coins that represent the change for a purchase given:

* M: the money that the customer inserted in the machine
* P: the product’s price
* C: The list of coins. This array will always be ordered and the first member will always be 1

The coins in C as well as M and P are integers, think of them as being in US Cents.

For example, you are given integer M = 1000, P = 499 and array C such that:

C[0] = 1

C[1] = 5

C[2] = 10

C[3] = 25

C[4] = 50

The result should be the following array:

R[0] = 1 //1 coin of 1 cent

R[1] = 0 //0 coins of 5 cents

R[2] = 0 //0 coins of 10 cents

R[3] = 0 //0 coins of 25 cents

R[4] = 10 //10 coins of 50 cents

Write a function in C#:

public int[] solution (int M, int P, int[] C)

that, given a positive integer M, a positive integer P and a non-empty zero-indexed array C consisting of N integers, returns an integer array with the optimal coin count representing the change.

If there’s not enough money to buy the product, the function should return null.

# Card Decks

One day in a casino a large amount of playing cards fell on the floor and got scrambled. To make things worse, some cards went missing in the event. A zero indexed array C consisting of N strings representing the different cards that fell on the floor is given.

The goal is to find the amount of complete playing card decks that can be recovered from the floor given:

* C: The array of unordered playing cards that fell on the floor.

A complete set consists of 52 cards of 4 suits {♣, ♦, ♥, ♠} and 13 ranks {A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K} (4x13=52).

For example, if you are given the following array:

{Q♥, J♥, K♣, 2♣, 8♠, Q♣, 6♦, 7♥, 2♠, 8♥, 3♣, 3♥, 7♦, J♠, A♦, J♦, 10♠, 9♣, A♥, 5♠, K♥, J♥, 10♣, 3♠, A♣, J♦, 8♦, 9♦, 2♥, 10♣, 7♣, 2♠, 10♦, Q♦, 2♦, A♦, A♣, K♠, 3♠, A♠, 10♠, 4♥, 6♠, J♣, 7♣, 2♥, 9♣, 10♥, 6♥, 8♠, 4♦, K♥, 4♣, 10♥, 9♥, Q♣, K♦, 3♦, 8♣, 6♥, 4♥, 7♠, 5♦, A♥, 6♣, Q♥, 8♦, 3♦, 9♥, J♣, 4♠, Q♦, 5♣, 9♦, Q♠, 4♣, 2♦, 9♠, 2♣, 6♠, 8♥, 6♦, 3♣, 8♣, 9♠, 7♠, K♣, 7♦, 5♦, Q♠, A♠, K♦, 10♦, 7♥, 5♥, K♠, 4♦, 5♠, 6♣, 3♥, J♠, 4♠, 5♥}

The result should be 1 because each card is repeated twice except for the 5 of clubs (5♣) which is present only once.

Write a function in C#:

public int solution (string[] C)

that, given a non-empty zero-indexed array C consisting of N strings, returns an integer with the number of complete playing card decks.

If it’s not possible to assemble a single deck, the function should return -1.

# Input

new string[]{"Q♥", "J♥", "K♣", "2♣", "8♠", "Q♣", "6♦", "7♥", "2♠", "8♥", "3♣", "3♥", "7♦", "J♠", "A♦", "J♦", "10♠", "9♣", "A♥", "5♠", "K♥", "J♥", "10♣", "3♠", "A♣", "J♦", "8♦", "9♦", "2♥", "10♣", "7♣", "2♠", "10♦", "Q♦", "2♦", "A♦", "A♣", "K♠", "3♠", "A♠", "10♠", "4♥", "6♠", "J♣", "7♣", "2♥", "9♣", "10♥", "6♥", "8♠", "4♦", "K♥", "4♣", "10♥", "9♥", "Q♣", "K♦", "3♦", "8♣", "6♥", "4♥", "7♠", "5♦", "A♥", "6♣", "Q♥", "8♦", "3♦", "9♥", "J♣", "4♠", "Q♦", "5♣", "9♦", "Q♠", "4♣", "2♦", "9♠", "2♣", "6♠", "8♥", "6♦", "3♣", "8♣", "9♠", "7♠", "K♣", "7♦", "5♦", "Q♠", "A♠", "K♦", "10♦", "7♥", "5♥", "K♠", "4♦", "5♠", "6♣", "3♥", "J♠", "4♠", "5♥"};