# Coin Change

Given a value N and an integer vector COINS representing coins of different denominations. Considering you have infinite supply of each coin, your task is to find total number of combinations of these coins that make a sum of N. If that amount of money cannot be made up by any combination of the coins, return 0.

**Input Format**

In the function an integer N is given, and a vector COINS consisting of coins.

**Output Format**

Return an integer representing total number of combinations.

**Constraint**

Total number of ways <= 10^12.

**Sample Input**

```
4 1 2 3
```

**Sample Output**

```
4
```

**Explanation**

{1, 1, 1, 1}, {1, 1, 2}, {1, 3}, {2, 2} are different combinations having a sum of 4.

**Solution**: coinChange.cpp

# Vacation

Taro's summer vacation starts tomorrow, and he has decided to make plans for it now.

The vacation consists of N days. For each i ($1 \leq i \leq N$), Taro will choose one of the following activities and do it on the i-th day:

- A: Swim in the sea. Gain $a_i$ points of happiness.

- B: Catch bugs in the mountains. Gain $b_i$ points of happiness.

- C: Do homework at home. Gain $c_i$ points of happiness.

As Taro gets bored easily, he cannot do the same activities for two or more consecutive days.

Find the maximum possible total points of happiness that Taro gains.

**Input Format**

In the function 3 vectors are passed.

**Output Format**

Print the maximum possible total points of happiness that Taro gains.

**Sample Input**

```
{10, 20, 30}{40, 50, 60}{70, 80, 90}
```

**Sample Output**

```
210
```

**Solution**: vacation.cpp

# Optimal Game Strategy

Oswald and Henry are playing the game of coins. They have a row of 'n' coins [C1,C2,C3...Cn] with values [V1,V2,V3...Vn] where Ci coin has Vi value. They take turns alternatively. In one turn the player can pick either the first or the last coin of the row. Given both Oswald and Henry are very smart players, you need to find the maximum possible value Oswald can earn if he plays first.

**Input Format:**
In the function you are given an integer N i.e. the number of coins and a vector V which represents the values of each coin in the row respectively.

**Output Format:** Return a single integer which is the maximum possible value as asked in the question.

**Constraints:** 1<=N<=15 1<=V[i]<=1000

**Sample Testcase:**

**Input: 4 1 2 3 4**
**Output: 6**

**Explanation:** Oswald will pick up coin with value 4, Henry will pick coin with value 3, Oswald will pick 2 and Henry will pick 1. Hence 4+2=6.

**Solution**: gameStrategy.cpp

# Minimum Partitioning

Given a set of integers, the task is to divide it into two sets S1 and S2 such that the absolute difference between their sums is minimum.
If there is a set S with n elements, then if we assume Subset1 has m elements, Subset2 must have n-m elements and the value of abs(sum(Subset1) – sum(Subset2)) should be minimum.

**Input Format**

In the function an integer vector is passed.

**Output Format**

Return an integer representing minimum difference.

**Sample Input**

```
{1, 6, 11, 5}
```

**Sample Output**

```
1
```

**Explanation**

```
Subset1 = {1, 5, 6}, sum of Subset1 = 12 Subset2 = {11}, sum of Subset2 = 11
```

**Solution**: minPartitioning.cpp