

# 989. Add to Array-Form of Integer

Solved

Easy Topics Companies

The **array-form** of an integer `num` is an array representing its digits in left to right order.

- For example, for `num = 1321`, the array form is `[1,3,2,1]`.

Given `num`, the **array-form** of an integer, and an integer `k`, return the **array-form** of the integer `num + k`.

## Example 1:

**Input:** `num = [1,2,0,0]`, `k = 34`

**Output:** `[1,2,3,4]`

**Explanation:** `1200 + 34 = 1234`

## Example 2:

**Input:** `num = [2,7,4]`, `k = 181`

**Output:** `[4,5,5]`

**Explanation:** `274 + 181 = 455`

## Example 3:

**Input:** `num = [2,1,5]`, `k = 806`

**Output:** `[1,0,2,1]`

**Explanation:** `215 + 806 = 1021`

Problem List < > Run Submit Auto Premium

Description Accepted Editorial Solutions Submissions

All Submissions

Accepted

Zhukovlya submitted at Jul 01, 2024 22:03

Editorial Solution

Runtime: 158 ms | Beats 66.67% | Memory: 60.38 MB | Beats 79.31%

Analyze Complexity

Code C#

```
public class Solution {
    public IList<int> AddToArrayForm(int[] num, int k) {
        List<int> result = new List<int>();
        int carry = 0;
        int n = num.Length;

        // Перебираем массив num с конца
        for (int i = n - 1; i >= 0 || k > 0; i--) {
            // Получаем текущую цифру из массива num, если индекс в пределах массива
            int currentDigit = (i >= 0) ? num[i] : 0;
            // Суммируем текущую цифру, значение k и перенос
```

Testcase Test Result

Accepted Runtime: 103 ms

Case 1 Case 2 Case 3

Input

num =

[1,2,0,0]

k =

34

Output

Код:

```
using System.Collections.Generic;

public class Solution
{
    public IList<int> AddToArrayForm(int[] num, int k)
    {
```

```

List<int> result = new List<int>();
int carry = 0;
int n = num.Length;

// Перебираем массив num с конца
for (int i = n - 1; i >= 0 || k > 0; i--)
{
    // Получаем текущую цифру из массива num, если индекс в пределах массива
    int currentDigit = (i >= 0) ? num[i] : 0;
    // Суммируем текущую цифру, значение k и перенос
    int sum = currentDigit + (k % 10) + carry;
    carry = sum / 10;
    result.Add(sum % 10);
    k /= 10;
}

// Если после обработки всех цифр остался перенос, добавляем его
if (carry > 0)
{
    result.Add(carry);
}

// Переворачиваем результат, так как мы добавляли цифры с конца
result.Reverse();

return result;
}
}

```