# Exercise 1

Write a function *solution* that, given an integer N returns the máximum possible value (in absolute value) obtained by inserting one ‘5’ digit inside the decimal representation of integer N.

Examples:

- Given N=268, the function should return 5268
- Given N=670, the function should return 6750
- Given N=0, the function should return 50
- Given N=-999, the function should return -5999

You could asume that N is an integer within the range [-8000, 8000]

```
class Solution {
   public int solution(int N) {
       // write your code using any versión of C#, and indicate it
}
```

# Exercise 2

A non-empty array A of bits (1s and/or 0s) is given. The *maximal binary ones span* of A is the longest sequence of consecutive 1s.For example, for A such that:

A[0] = 0

A[1] = 1

A[2] = 1

A[3] = 1

A[4] = 0

A[5] = 1

A[6] = 1

A[7] = 1

A[8] = 0

A[9] = 1

the maximal binary ones span equals 3. There are two sequences of consecutives 1s of such length A[1..3] and A[5..7].

You are given an implementation of the function:

```
        class Solution { public int solution (int A[] ); }
```

that, given a non-empty array A consisting on N bits, finds the first sequence of consecutive 1s whose length equals the maximal binary ones span of A, and returns its starting position (index of the first 1 in the sequence).

It there are no bits set no 1, the function returns -1.

For example, given A as in the example above, the function should return 1, because the maximal binary ones span in A reaches 3 (which was already explained) and the first sequence of that length starts at index 1.

Given an array A consisting on two bits such that:

A[0] = 0

A[1] = 0

the function should return -1, as there are no 1s.

The attached code is still incorrect for some inputs. Despite the error(s), the code may produce a correct answer for the example test cases. The goal of the exercise is to find and fix the bug(s) in the implementation. You can modify, at most, two lines.

Assume that:

N is an integer in the range [1..1000];

Each element of array A is an integer that can have one of the following values 0:1

```
class Solution {
    public int solution(int[] A) {
        int n = A.Length;
        int i = n - 1;
        int result = -1;
        int k = 0, maximal = 0;
        while (i > 0) {
            if (A[i] == 1) {
                k = k + 1;
                if (k >= maximal) {
                    maximal = k;
                    result = i;
                }
            }
            else
                k = 0;
            i = i - 1;
        }
        if (A[i] == 1 && k + 1 > maximal)
            result = 0;
        return result;
    }
}
```

# Exercise 3

From the following table write a SQL query to find those distributors who purchased all types of item from the company. Return distributors ids.

*Input:*

*Table: items*

*Structur*e:

| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
|---|---|---|---|---|---|
| item_code | int(11) | No | PRI | | |
| item_name | varchar(255) | YES | | | |

**Example Data:**

| **item_code** | **item_name** |
|---|---|
| 10091 | juice |
| 10092 | chocolate |
| 10093 | cookies |
| 10094 | cake |

item_code is the primary key column for items table.

*Table: orders*

*Structure:*

| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
|---|---|---|---|---|---|
| order_id | int(11) | YES | | | |
| distributor_id | int(11) | YES | | | |
| item_ordered | int(11) | YES | MUL | | |
| item_quantity | int(11) | YES | | | |

*Example Data:*

| **order_id** | **distributor_id** | **item_ordered** | **item_quantity** |
|---|---|---|---|
| 1 | 501 | 10091 | 250 |
| 2 | 502 | 10093 | 100 |
| 3 | 503 | 10091 | 200 |
| 4 | 502 | 10091 | 150 |
| 5 | 502 | 10092 | 300 |
| 6 | 504 | 10094 | 200 |

|  |  |  |  |
|---|---|---|---|
| 7 | 503 | 10093 | 250 |
| 8 | 503 | 10092 | 250 |
| 9 | 501 | 10094 | 180 |
| 10 | 503 | 10094 | 350 |

item_ordered is a foreign key to items table.