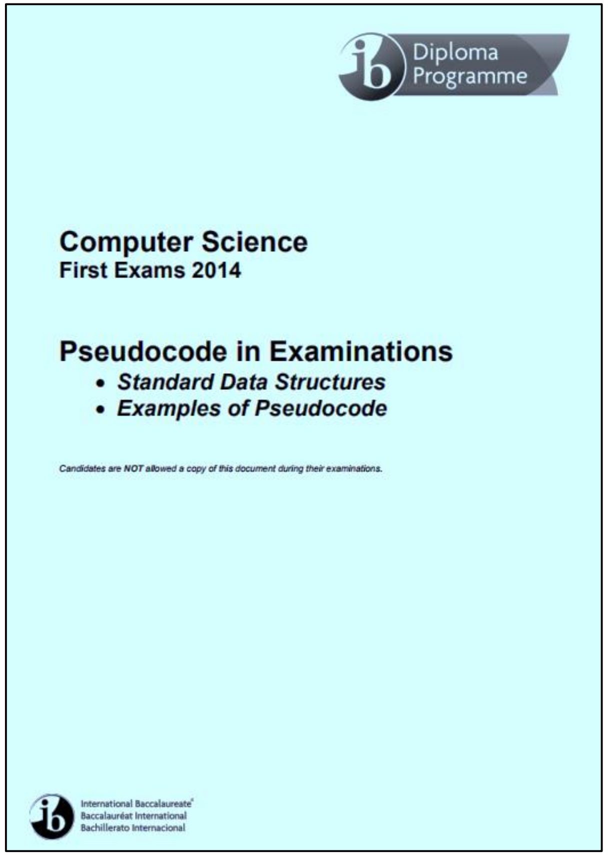
# Pseudo Code

PROGRAMMING ON PAPER FOR IB PAPER 1 EXAMS

SESSION 3

# 2 official pseudo code guides



Diploma Programme

**Computer Science**
**First Exams 2014**

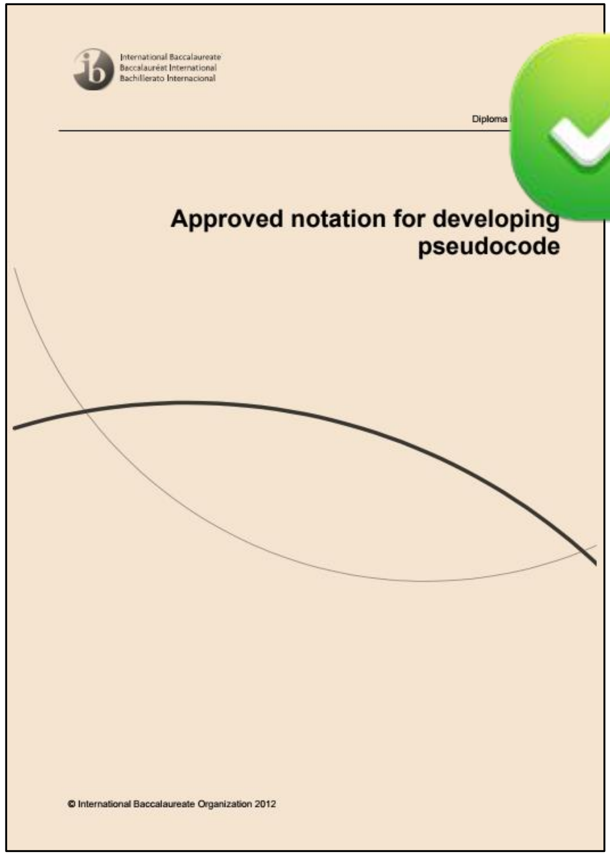
**Pseudocode in Examinations**

- ***Standard Data Structures***
- ***Examples of Pseudocode***

*Candidates are NOT allowed a copy of this document during their examinations.*

International Baccalaureate
Baccalauréat International
Bachillerato Internacional

8 pages
**NOT given in exam**



International Baccalaureate
Baccalauréat International
Bachillerato Internacional

Diploma

**Approved notation for developing pseudocode**

© International Baccalaureate Organization 2012

✓

3 pages
**Given in exam**

# Warning!

Pseudo code questions are *never* as overt or obvious as the examples we discuss in these sessions.

These examples are only there to teach you the *skills* you need to answer more complex problems.

*Topics 4, 5 and 7 can include pseudo code…*

# Top pseudo code tips

- ✓When possible, start answering a pseudo code question at the top of a page
- ✓Write pseudo code in pencil first and then copy into pen
- ✓Think about the data types and associated access methods BEFORE writing anything
- ✓Think about what control structures (especially loops) are associated with that data type
- ✓Be sure to return or output something at the end of the problem – even if they don’t ask for it!

# Basic structure of ALL pseudo code questions

```
Declarations / Initialisations
Control structures / Calculations
Output / Return
```

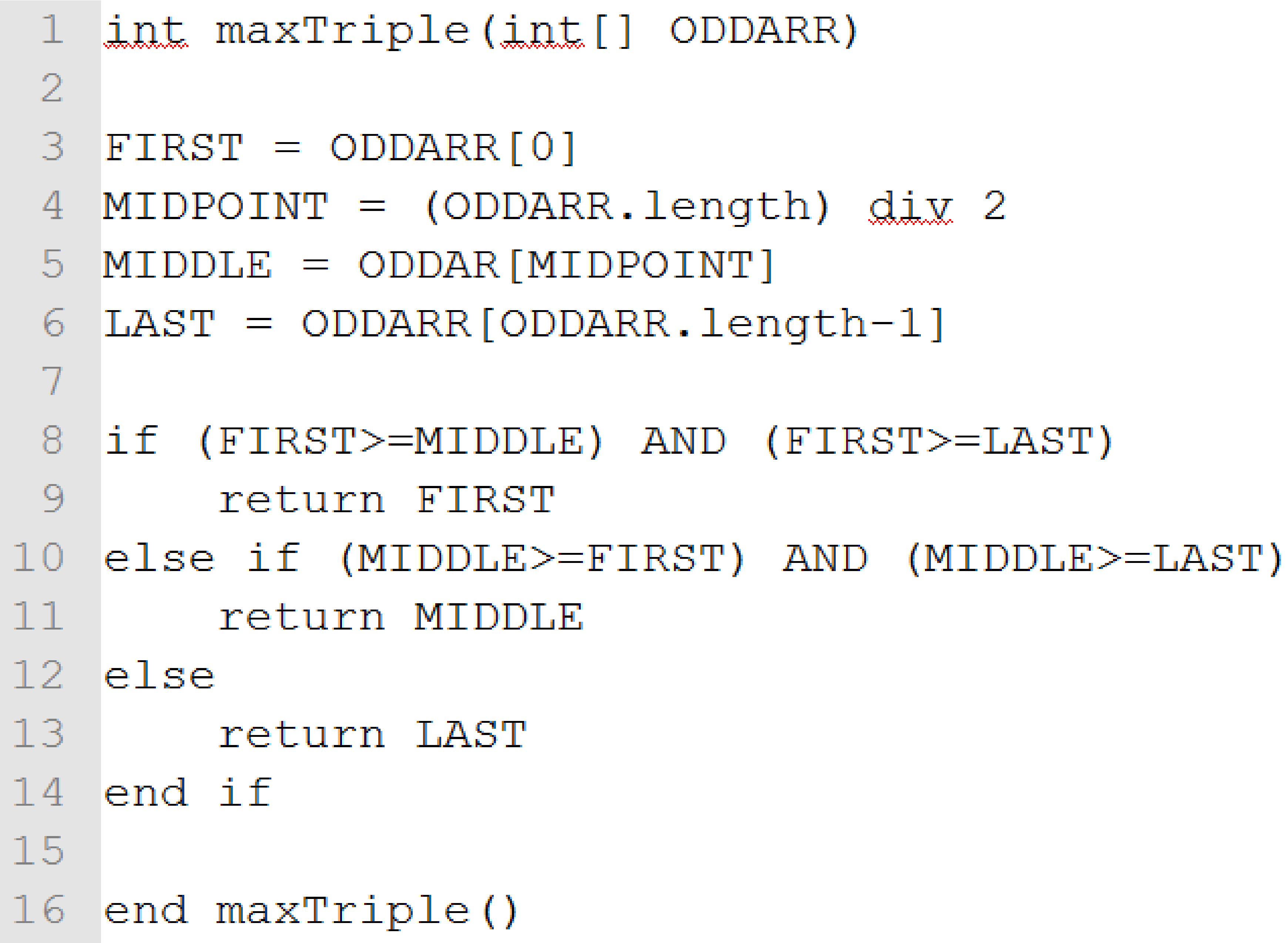
You get marks for individual sections, not the final output.

This means you could well get 7/8 even if your final output is not right.

# T1: Array based method

**Given an array of ints of odd length, look at the first, last, and middle values in the array and return the largest. The array length will be a least 1.**

```
maxTriple([3]) → 3
maxTriple([1, 5, 3]) → 5
maxTriple([1, 9, 5, 8, 3]) → 5
```

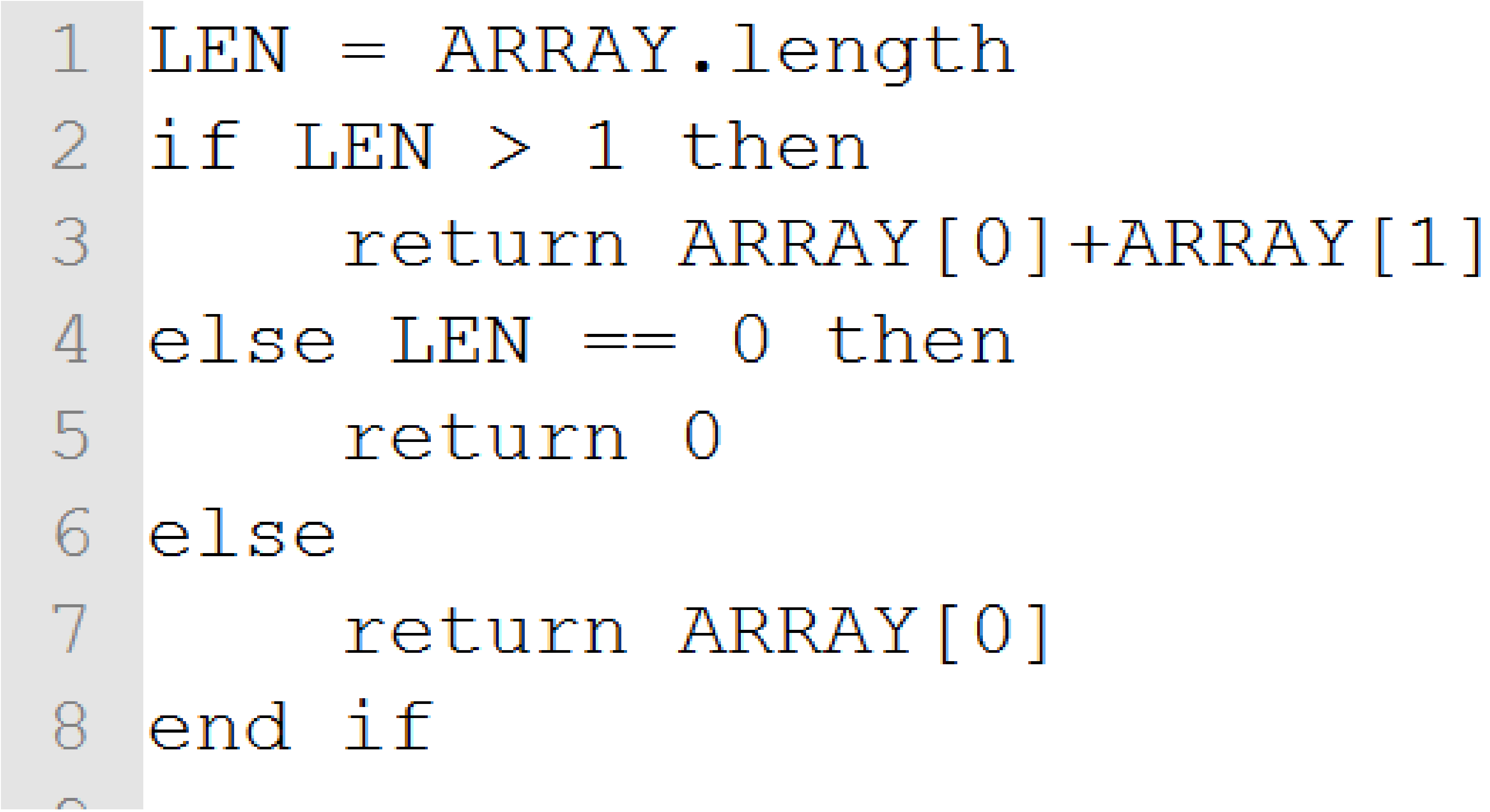


```
int maxTriple(int[] ODDARR)

FIRST = ODDARR[0]
MIDPOINT = (ODDARR.length) div 2
MIDDLE = ODDAR[MIDPOINT]
LAST = ODDARR[ODDARR.length-1]

if (FIRST>=MIDDLE) AND (FIRST>=LAST)
    return FIRST
else if (MIDDLE>=FIRST) AND (MIDDLE>=LAST)
    return MIDDLE
else
    return LAST
end if

end maxTriple()
```

# T2: Array counting

**Given an array of ints, return the sum of the first 2 elements in the array. If the array length is less than 2, just sum up the elements that exist, returning 0 if the array is length 0.**

```
sum2([]) → 0
sum2([1, 1]) → 2
sum2([1, 1, 1, 1]) → 2
```

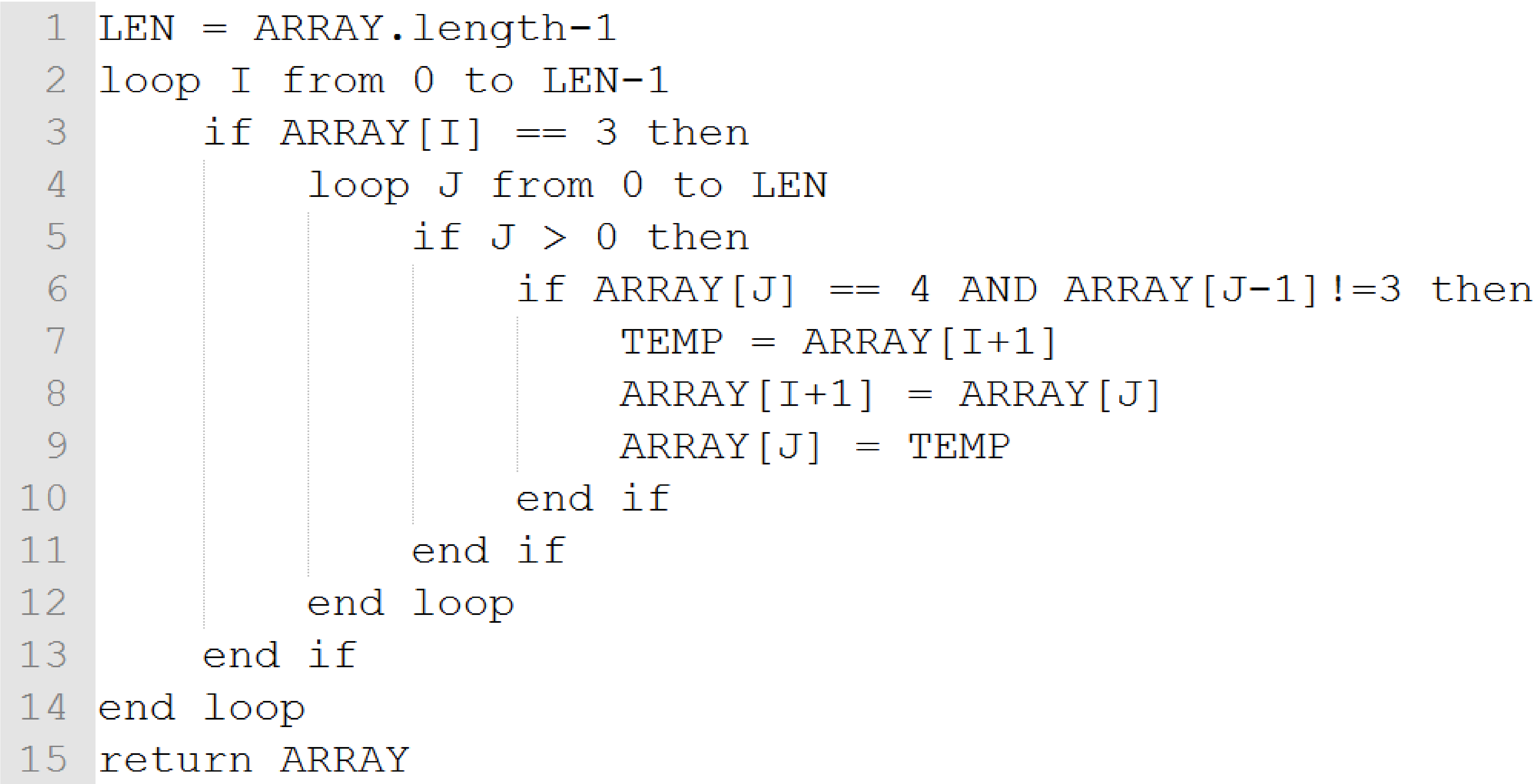


```
LEN = ARRAY.length
if LEN > 1 then
    return ARRAY[0]+ARRAY[1]
else LEN == 0 then
    return 0
else
    return ARRAY[0]
end if
```

# T3: Fix34 problem

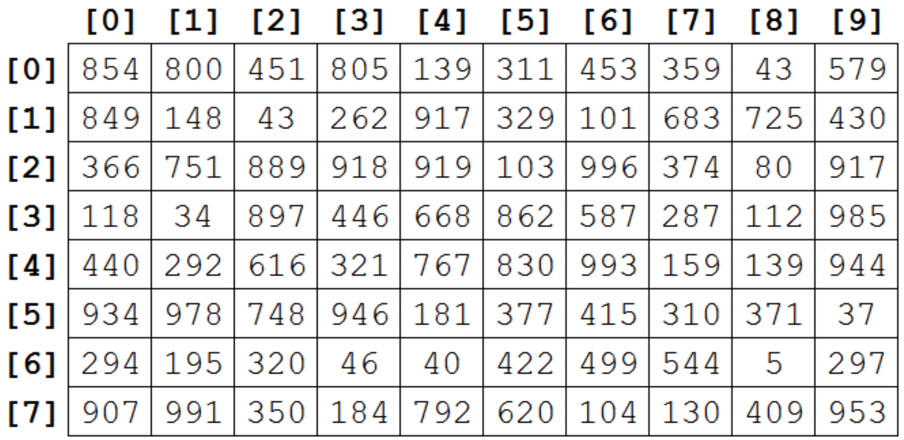
**Return an array that contains exactly the same numbers as the given array, but rearranged so that every 3 is immediately followed by a 4. Do not move the 3's, but every other number may move. The array contains the same number of 3's and 4's, every 3 has a number after it that is not a 3, and a 3 appears in the array before any 4.**

```
fix34([1, 3, 1, 4]) →          [1, 3, 4, 1]
fix34([1, 3, 1, 4, 4, 3, 1]) → [1, 3, 4, 1, 1, 3, 4]
fix34([3, 2, 2, 4]) →          [3, 4, 2, 2]
```



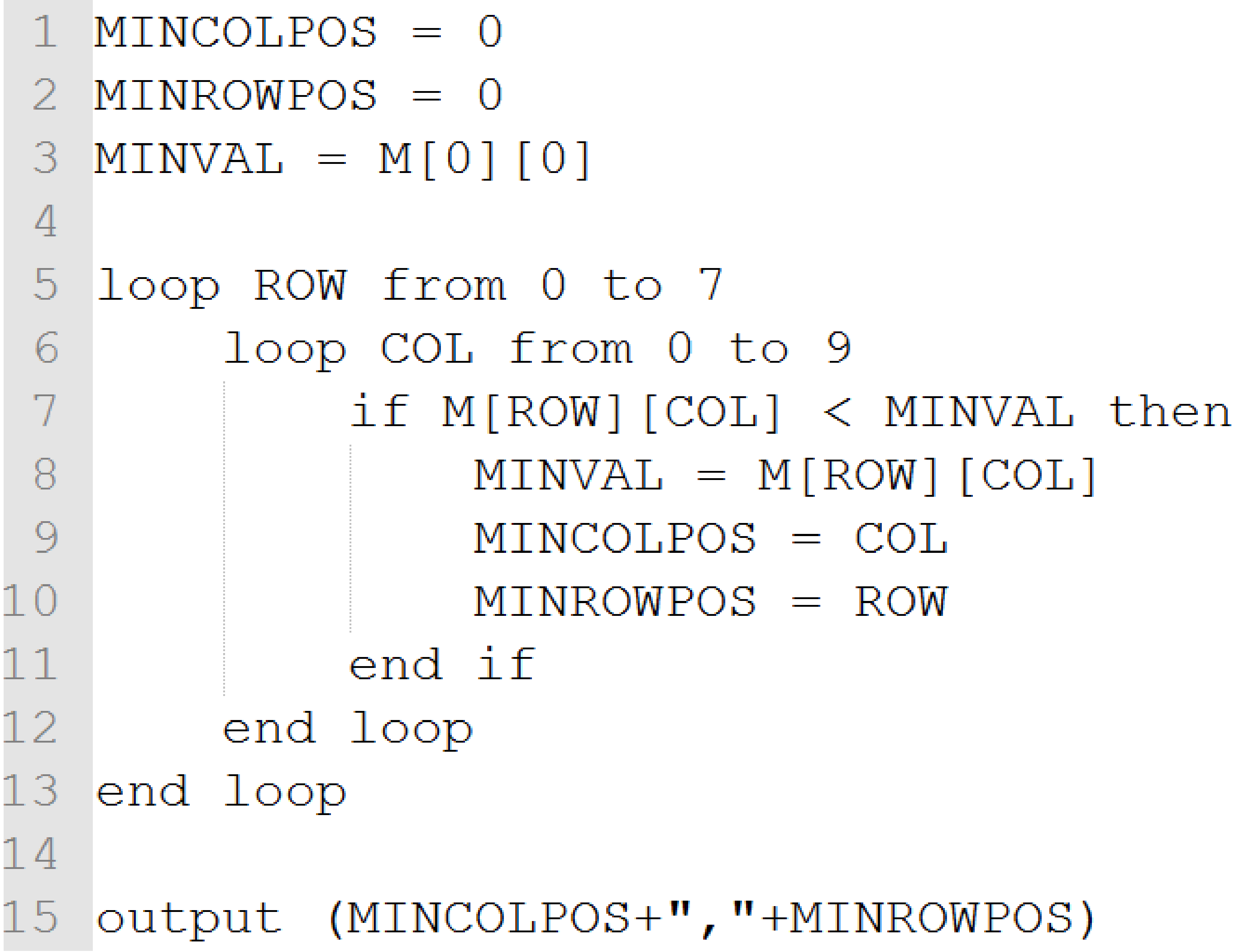
```
LEN = ARRAY.length-1
loop I from 0 to LEN-1
    if ARRAY[I] == 3 then
        loop J from 0 to LEN
            if J > 0 then
                if ARRAY[J] == 4 AND ARRAY[J-1]!=3 then
                    TEMP = ARRAY[I+1]
                    ARRAY[I+1] = ARRAY[J]
                    ARRAY[J] = TEMP
                end if
            end if
        end loop
    end if
end loop
return ARRAY
```

# T4: Position of smallest element in 2D array



| | [0] | [1] | [2] | [3] | [4] | [5] | [6] | [7] | [8] | [9] |
|---|---|---|---|---|---|---|---|---|---|---|
| [0] | 854 | 800 | 451 | 805 | 139 | 311 | 453 | 359 | 43 | 579 |
| [1] | 849 | 148 | 43 | 262 | 917 | 329 | 101 | 683 | 725 | 430 |
| [2] | 366 | 751 | 889 | 918 | 919 | 103 | 996 | 374 | 80 | 917 |
| [3] | 118 | 34 | 897 | 446 | 668 | 862 | 587 | 287 | 112 | 985 |
| [4] | 440 | 292 | 616 | 321 | 767 | 830 | 993 | 159 | 139 | 944 |
| [5] | 934 | 978 | 748 | 946 | 181 | 377 | 415 | 310 | 371 | 37 |
| [6] | 294 | 195 | 320 | 46 | 40 | 422 | 499 | 544 | 5 | 297 |
| [7] | 907 | 991 | 350 | 184 | 792 | 620 | 104 | 130 | 409 | 953 |

**A 2D array called M of 10 columns and 8 rows is created and filled with random unique integers. Return the location in the format COL, ROW of the smallest integer in the array.**



```
MINCOLPOS = 0
MINROWPOS = 0
MINVAL = M[0][0]

loop ROW from 0 to 7
    loop COL from 0 to 9
        if M[ROW][COL] < MINVAL then
            MINVAL = M[ROW][COL]
            MINCOLPOS = COL
            MINROWPOS = ROW
        end if
    end loop
end loop

output (MINCOLPOS+","+MINROWPOS)
```