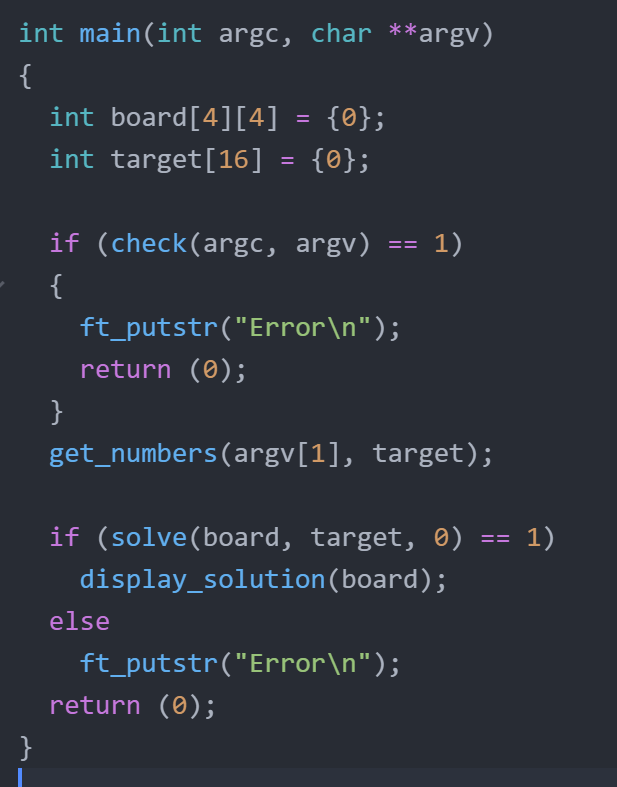
# Main Function

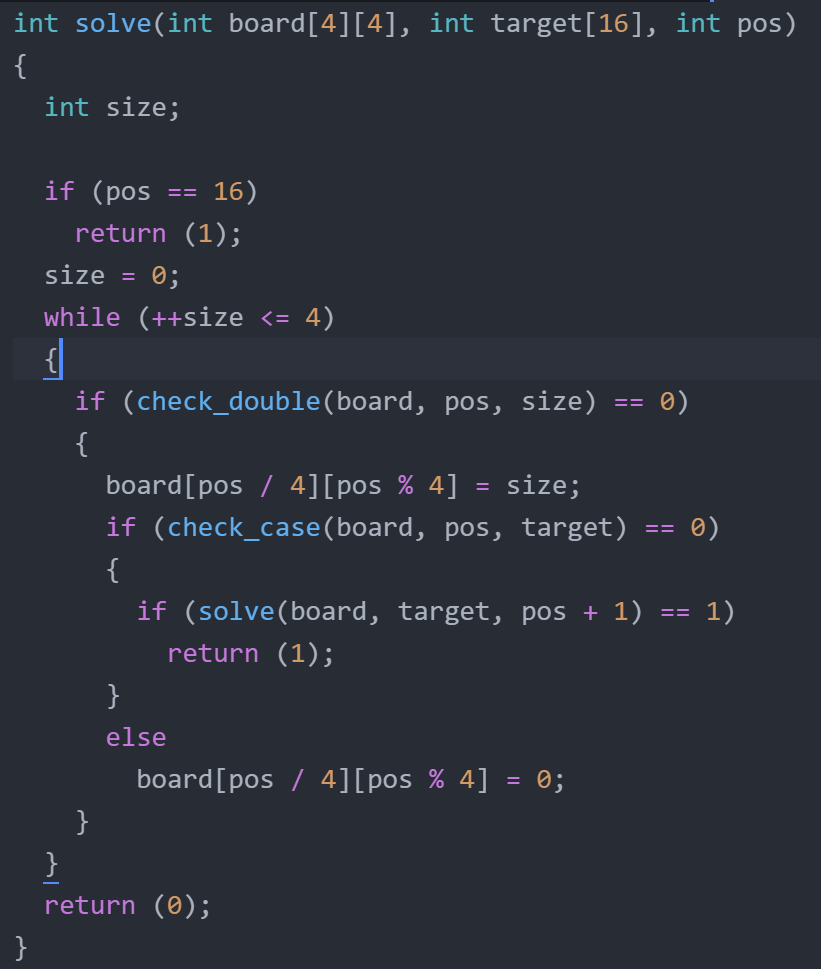


* Initialize the board to zeroes.
* The **board** consists of 16 positions from pos 0 to pos 15

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | col 0 | | col 1 | col 2 | col 3 |
| row 0 | | 0 | 1 | 2 | 3 |
| row 1 | | 4 | 5 | 6 | 7 |
| row 2 | | 8 | 9 | 10 | 11 |
| row 3 | | 12 | 13 | 14 | 15 |

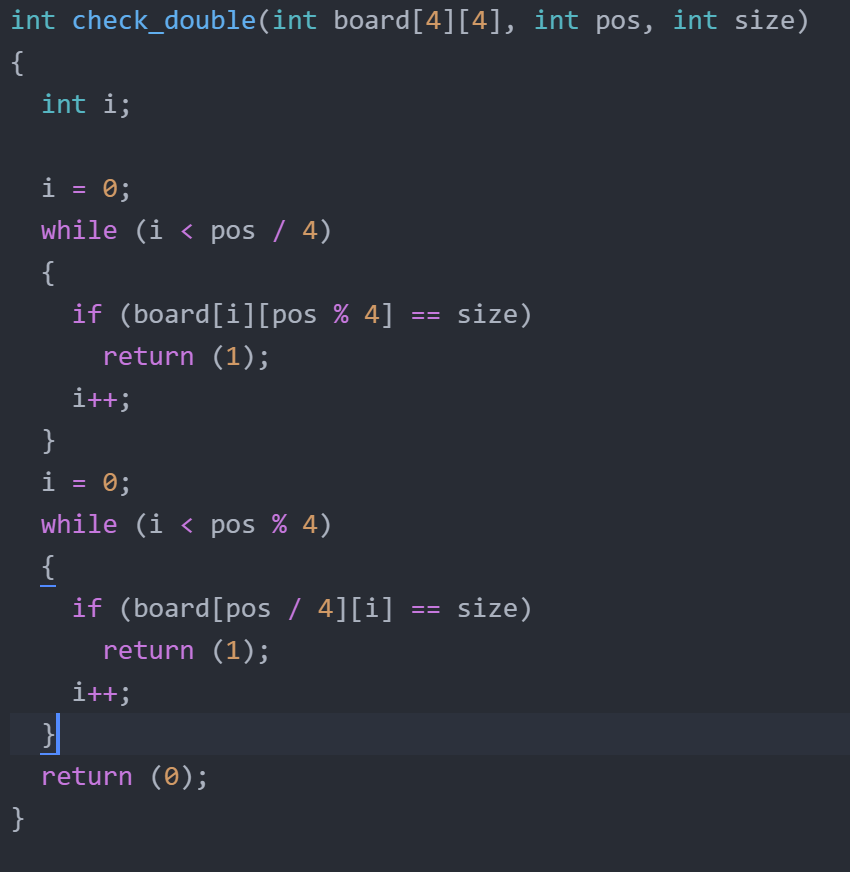
* We can find row number as pos / 4
* We can find column number as pos % 4
* **Target** is the visibility values (16 of them) received as input from the user

# Solve()



* Iterates from pos 0 to 15 trying to put numbers in each position
* For each position:
  + Iterate from 1 to 4 trying to select tower size 1 to 4
  + check\_double is to make sure that the number is not already used in that column and row
  + if it passes check\_double we use the size and go check if this number will give us the target visibility needed 🡪 check\_case()
  + if it fails the test, we try the next number
  + when we find a number, we try to solve the next position
  + if the next position is not solvable, we move to next size

# check\_double



Checks if the number we are trying to use as size of tower is already used in the same column/row

It checks only the previous positions since position > current position have not been filled yet and shouldn’t be considered

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | col 0 | col 1 | col 2 | col 3 |
| row 0 | 0 | 1 | 2 | 3 |
| row 1 | 4 | 5 | 6 | 7 |
| row 2 | 8 | 9 | **10** | 11 |
| row 3 | 12 | 13 | 14 | 15 |

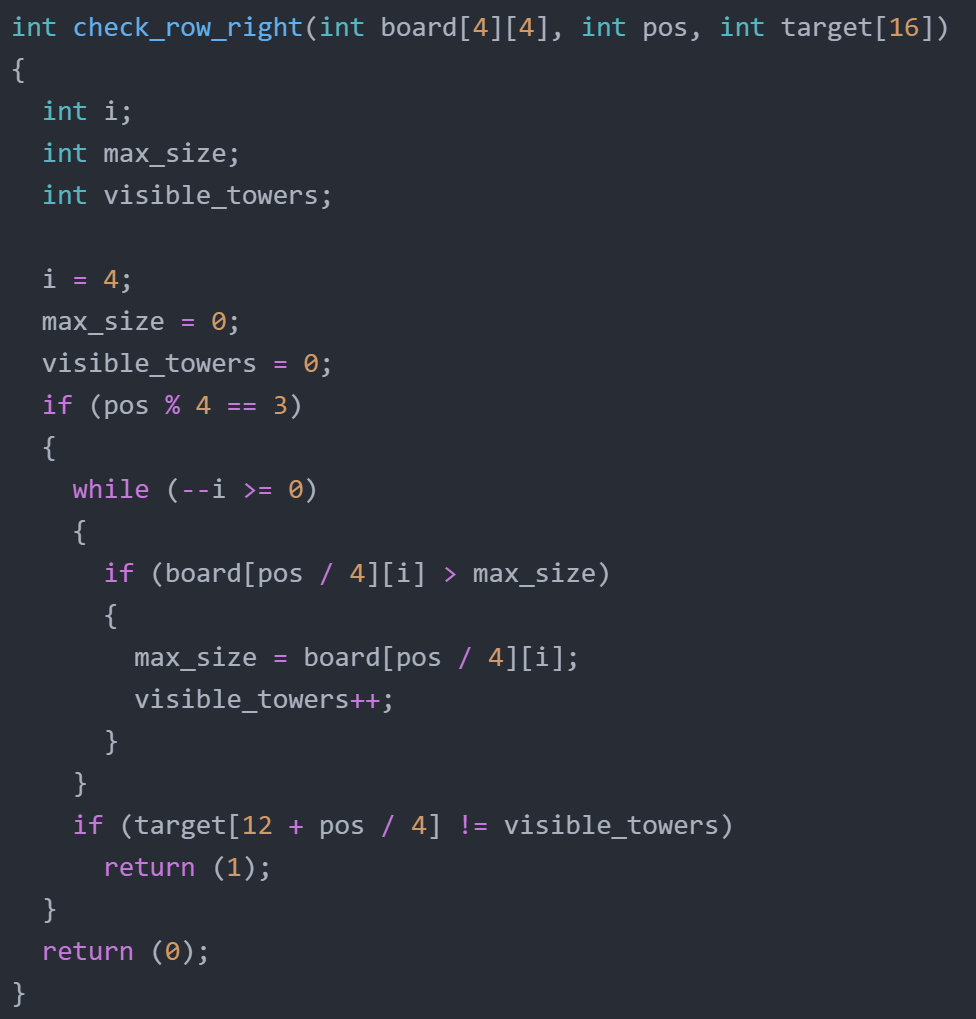
# Check\_case()

# 

Checks if the size chosen will fit in the restrictions put by the input/target

Return 1 if it disagrees with any column or row

# Check\_row and check\_column





* if (pos / 4 == 3) 🡪 checks only completed rows
* the while loop counts how many towers are visible with the proposed solution
* the if statement checks if the visible towers matches the target value