**#1**

Given the string, check if it is a [palindrome](keyword://palindrome).

Example

* For inputString = "aabaa", the output should be  
  checkPalindrome(inputString) = true;
* For inputString = "abac", the output should be  
  checkPalindrome(inputString) = false;
* For inputString = "a", the output should be  
  checkPalindrome(inputString) = true.

Input/Output

* **[execution time limit] 3 seconds (java)**
* **[input] string inputString**

A non-empty string consisting of lowercase characters.

*Guaranteed constraints:*  
1 ≤ inputString.length ≤ 105.

* **[output] boolean**
  + true if inputString is a palindrome, false otherwise.

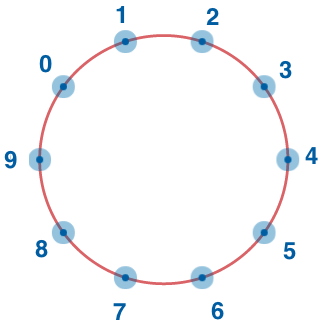
Consider integer numbers from 0 to n - 1 written down along the circle in such a way that the distance between any two neighboring numbers is equal (note that 0 and n - 1 are neighboring, too).

**#2**

Given n and firstNumber, find the number which is written in the radially opposite position to firstNumber.

Example

For n = 10 and firstNumber = 2, the output should be  
circleOfNumbers(n, firstNumber) = 7.



Input/Output

* **[execution time limit] 3 seconds (java)**
* **[input] integer n**

A positive **even** integer.

Guaranteed constraints:  
4 ≤ n ≤ 20.

* **[input] integer firstNumber**

Guaranteed constraints:  
0 ≤ firstNumber ≤ n - 1.

* **[output] integer**

**#3**

After they became famous, the CodeBots all decided to move to a new building and live together. The building is represented by a rectangular matrix of rooms. Each cell in the matrix contains an integer that represents the price of the room. Some rooms are *free* (their cost is 0), but that's probably because they are haunted, so all the bots are afraid of them. That is why any room that is *free* or is located **anywhere below** a *free* room in the same column is not considered suitable for the bots to live in.

Help the bots calculate the total price of all the rooms that are suitable for them.

**Example**

* For

matrix = [[0, 1, 1, 2],

[0, 5, 0, 0],

[2, 0, 3, 3]]

the output should be  
matrixElementsSum(matrix) = 9.

Here's the rooms matrix with unsuitable rooms marked with 'x':

[[x, 1, 1, 2],

[x, 5, x, x],

[x, x, x, x]]

Thus, the answer is 1 + 5 + 1 + 2 = 9.

* For

matrix = [[1, 1, 1, 0],

[0, 5, 0, 1],

[2, 1, 3, 10]]

the output should be  
matrixElementsSum(matrix) = 9.

Here's the rooms matrix with unsuitable rooms marked with 'x':

[[1, 1, 1, x],

[x, 5, x, x],

[x, 1, x, x]]

Note that the free room in the first row make the full column unsuitable for bots.

Thus, the answer is 1 + 1 + 1 + 5 + 1 = 9.

Input/Output

* **[execution time limit] 3 seconds (java)**
* **[input] array.array.integer matrix**

A 2-dimensional array of integers representing a rectangular matrix of the building.

*Guaranteed constraints:*  
1 ≤ matrix.length ≤ 5,  
1 ≤ matrix[i].length ≤ 5,  
0 ≤ matrix[i][j] ≤ 10.

* **[output] integer**
  + The total price of all the rooms that are suitable for the CodeBots to live in.