Índice

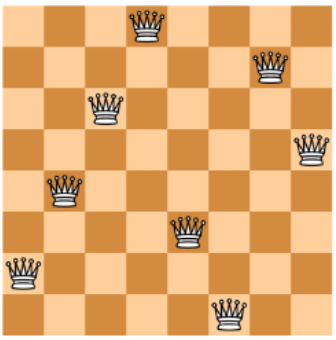
Ocho reinas 2
Números pentagonales 2
Cuadrado máximo 3
Símbolos simples 3
Sopa de letras 3
Camino correcto 4
Balanza equilibrada 4
Enemigo más cercano 4
Ciclos recurrentes 5
Primos cuadráticos 5
Diagonales de una espiral de números 6
Potencias distintas 6
Suma de monedas 6
Primos circulares 7
Palíndromos de doble base 7
Números triangulares codificados 7
Suma de primos consecutivos 7
¡Feliz no cumpleaños! 8

Ocho reinas

Cree una función la cual lea un parámetro que consista en un array de localizaciones de ocho reinas en un tablero estándar de ajedrez, sin que haya más piezas en el tablero. La estructura del parámetro deberá ser: [“(x,y)”, “(x,y)”,…] donde (x,y) representa la posición de cada una de las reinas en el tablero (“x” e “y” deben estar en el rango de 1 a 8, donde (1,1) es la casilla inferior izquierda del tablero, y (8,8) la casilla superior derecha). Su programa debe determinar si todas las reinas están colocadas de tal manera que ninguna de ellas está atacando a alguna otra. Si esto es correcto para el input, el programa deberá devolver el string “True”, en otro caso, deberá devolver la primera reina que está atacando a otra.

Ejemplos:

Input: {"(2,1)", "(4,2)", "(6,3)", "(8,4)", "(3,5)", "(1,6)", "(7,7)", "(5,8)"}



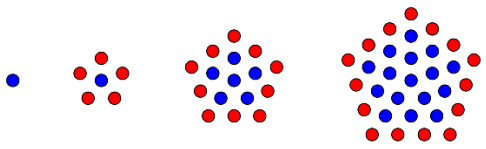
Output: “True”

Input: {"(2,1)", "(4,3)", "(6,3)", "(8,4)", "(3,4)", "(1,6)", "(7,7)", "(5,8)"}

Output: “(2,1)”

Números pentagonales

Cree una función la cual lea un parámetro que sea un integer positivo y determine cuantos puntos existen en un pentágono con longitud de lado igual a dicho integer y alrededor de un punto central, en la enésima iteración. Por ejemplo, en la imagen inferior puede ver que la primera iteración es un único punto, en la segunda hay 6 puntos, en la tercera hay 16 puntos y en la cuarta hay 31 puntos.



Su programa deberá devolver el número de puntos que existen en el pentágono completo en la enésima iteración (la cual vendrá dada por el parámetro pasado a la función).

Ejemplos:

Input: 2

Output: 6

Input: 5

Output: 51

Cuadrado máximo

Cree una función la cual lea un parámetro que sea una matriz 2D de ceros y unos, y que determine el área de la mayor submatriz cuadrada que contenga solo unos. Una submatriz cuadrada es una en la que tenga igual altura y anchura, y su programa deberá devolver el área de la mayor submatriz cuadrada que solo contenga unos.

Ejemplos:

Input: {"10100", "10111", "11111", "10010"}

1	0	1	0	0
1	0	1	**1**	**1**
1	1	1	**1**	**1**
1	0	0	1	0

Output: 4

Input: {"0111", "1101", "0111"}

Output: 1

Símbolos simples

Cree una función la cual lea un parámetro que determine si es una secuencia aceptable, devolviendo True o False. El parámetro estará compuesto de símbolos "+" e "=" con algunas letras y/o números entre ellos (por ejemplo: ++d+===+c++==a) y para que el string sea aceptable cada letra (no los números) deberá cumplir una de las siguientes condiciones:

- Antes de la letra debe haber los símbolos "=+" (consecutivos) y después el símbolo "+". **Ejemplo**: =+a+
- Antes y después de la letra debe haber el símbolo "=", pero sin tener el símbolo "+" consecutivo. **Ejemplo**: =a= -> True, pero +=a= o =a=+ -> False

El string no puede estar vacío o contener menos de 3 caracteres.

Ejemplos:

Input: "=d=3=+s++"

Output: True

Input: "f++d+"

Output: False

Sopa de letras

Cree una función la cual lea un parámetro string y devuelva el string con las letras ordenadas alfabéticamente, y los números ordenados del 0 al 9 al final de dicho string. Asuma que los símbolos de puntuación no estarán incluidos en el string.

Ejemplos:

Input: "informatica"

Output: "aacfiimnort"

Input: "dia16112018"

Output: "adi01111268"

Camino correcto

Cree una función la cual lea un parámetro que representa los movimientos realizados en un tablero 5x5 empezando en la posición superior izquierda. El parámetro será un string compuesto únicamente por los caracteres “n”, “s”, “e”, “o”, “?”. Cada uno de los caracteres representa un movimiento posible en el tablero, siendo n = norte, s = sur, e = este, o = oeste. El objetivo es determinar que caracteres (“n”, “s”, “e”, “o”) deberían ser los signos de interrogación (“?”) del string para formar un camino que llegue desde la esquina superior izquierda del tablero a la esquina inferior derecha, sin pasar por ninguna posición recorrida anteriormente.

Por ejemplo, si el parámetro es “e?s?sess” su programa deberá devolver el string correcto que permita formar un camino desde la esquina superior derecha a la inferior izquierda del tablero 5x5. Para este parámetro, su programa devolverá el string “eesesess”. El parámetro deberá tener únicamente una solución correcta y debe tener al menos un símbolo de interrogación. En el caso de no tener solución, su programa deberá devolver el string “Sin solución”.

Ejemplos:

Input: “???eenese?”

Output: “ssseeneses”

Input: “sese??eesss?”

Output: “sesenneessss”

Balanza equilibrada

Cree una función la cual lea dos parámetros, el primero siendo un array de dos integer positivos que representen el peso en una balanza (lados izquierdo y derecho), y el segundo siendo un array de pesos disponibles (integer) para poner en la balanza. El objetivo es determinar si puedes equilibrar la balanza usando al menos uno de los pesos disponibles, y como mucho dos. Por ejemplo: si los parámetros son {5, 9} y {1, 2, 6, 7}, entonces la balanza se podrá equilibrar añadiendo en el lado izquierdo el peso de 6, y el peso de 2 en el lado derecho. Su programa deberá devolver String con los pesos usados, ordenados de menos a mayor, y separados por una coma. En este ejemplo, su programa deberá devolver “2,6”.

Solo debe haber una única solución posible y la lista de pesos a añadir no debe estar vacía. Es posible también, añadir dos pesos en el mismo lado de la balanza. Si no fuera posible equilibrar la balanza, su programa deberá devolver el string “Imposible”.

Ejemplos:

Input: {3, 4} {1, 2, 5, 7}

Output: “1”

Input: {13, 4} {1, 2, 3, 6, 14}

Output: “3,6”

Enemigo más cercano

Cree una función la cual lea un parámetro que represente una matriz cuadrada 2D de números. Dicha matriz solo contendrá los números 1, 0 o 2. El objetivo será, desde la posición de la matriz donde haya un 1, devolver el número de movimientos (arriba, abajo, izquierda o derecha) que deberá hacer para alcanzar a un enemigo, representado por un 2. Es posible ir de un lado a otro de la matriz, es decir, desde el lado izquierdo de la matriz se puede llegar al derecho con un movimiento a la izquierda. Por ejemplo, para la matriz:

0	0	0	0
1	0	0	0
0	0	0	2
0	0	0	2

Su programa deberá devolver 2, ya que el enemigo más cercano (representado por un 2) está a dos espacios del 1, moviéndose a la izquierda y luego hacia abajo. El parámetro podrá contener cualquier número de 0 y 2, pero solo un 1. Si no contiene ningún 2, su programa deberá devolver un 0.

Ejemplos:

Input: [“000”, “100”, “200”]

Output: 1

Input: [“0000”, “2010”, “0000”, “2002”]

Output: 2

Ciclos recurrentes

Una fracción unitaria contiene un 1 en el numerador. La representación decimal de las fracciones unitarias con denominadores del 2 al 10 vienen dadas por:

1/2 = 0,5

1/3 = 0,(3)

1/4 = 0,25

1/5 = 0,2

1/6 = 0,1(6)

1/7 = 0,(142857)

1/8 = 0,125

1/9 = 0,(1)

1/10 = 0,1

Donde 0,1(6) significa 0,16666…, que tiene un ciclo recurrente de 1 dígito. Podemos ver que 1/7 tiene un ciclo recurrente de 6 dígitos.

Cree una función que encuentre el valor para d < 1000 (el denominador) que contenga el mayor ciclo recurrente en su parte decimal.

Primos cuadráticos

Euler descubrió la notable fórmula cuadrática:

$$n^2 + n + 41$$

La fórmula produce los 40 primos para los valores enteros consecutivos $0 \leq n \leq 39$. Sin embargo, cuando n = 40, $40^2 + 40 + 41 = 40(40+1) + 41$ es divisible por 41, y ciertamente cuando n = 41, $41^2 + 41 + 41$ es claramente divisible por 41.

La increíble fórmula $n^2 -79n + 1601$ fue descubierta, la cual produce 80 primos para los valores enteros consecutivos $0 \leq n \leq 79$. El producto de los coeficientes, -79 y 1601, es -126479.

Considerando cuadráticas de la forma:

$$n^2 + an + b, \text{ donde } |a| < 1000 \text{ y } |b| \leq 1000$$

donde |n| es el valor absoluto de n (por ejemplo, |11| = 11 y |-4| = 4)

Cree una función que encuentre el producto de coeficientes, "a" y "b", para la expresión cuadrática que produzca el máximo número de primos para los valores consecutivos de n, empezando con n = 0.

Diagonales de una espiral de números

Empezando por el número 1 y moviéndose a la derecha en el sentido de las agujas del reloj, se forma una espiral 5x5 como la siguiente:

21	22	23	24	25
20	7	8	9	10
19	6	1	2	11
18	5	4	3	12
17	16	15	14	13

Se puede verificar que la suma de los números en las diagonales es 101.

Cree una función que calcule la suma de los números de las diagonales en una espiral 1001x1001 formada de la misma manera.

Potencias distintas

Considere todas las combinaciones de enteros de a^b para $2 \leq a \leq 5$ y $2 \leq b \leq 5$:

$2^2 = 4, 2^3 = 8, 2^4 = 16, 2^5 = 32$

$3^2 = 9, 3^3 = 27, 3^4 = 81, 3^5 = 243$

$4^2 = 16, 4^3 = 64, 4^4 = 256, 4^5 = 1024$

$5^2 = 25, 5^3 = 125, 5^4 = 625, 5^5 = 3125$

Si se ordenan en orden numérico, eliminando las repeticiones, obtenemos la siguiente secuencia de 15 términos distintos:

4, 8, 9, 16, 25, 27, 32, 64, 81, 125, 243, 256, 625, 1024, 3125

Cree una función que devuelva cuántos términos distintos hay en la secuencia generada por a^b donde $2 \leq a \leq 100$ y $2 \leq b \leq 100$.

Suma de monedas

En España la moneda se compone de euros, €, y céntimos, ¢, y hay ocho tipos distintos de monedas en circulación:

1¢, 2¢, 5¢, 10¢, 20¢, 50¢, 1€ (100¢) y 2€ (200¢)

Es posible obtener 2€ de la siguiente forma:

1*1€ + 1*50¢ + 2*20¢ + 1*5¢ + 1*2¢ + 3*1¢

Cree una función que calcule cuántas formas distintas hay de obtener 2€ usando cualquier número de monedas.

Primos circulares

El número 197 es llamado un primo circular ya que todas las rotaciones de sus dígitos (197, 971 y 719) son igualmente primos.

Hay trece primos circulares por debajo del 100: 2, 3, 5, 7, 11, 13, 17, 31, 37, 71, 73, 79 y 97.

Cree una función que calcule cuántos primos circulares hay por debajo del millón.

Palíndromos de doble base

El número decimal 585 = 1001001001 (binario) es un palíndromo (se lee igual de izquierda a derecha que de derecha a izquierda) en ambas bases.

Cree una función que encuentre todos los números, por debajo de un millón, que sean palíndromos en base 10 y base 2.

Números triangulares codificados

El enésimo término de la secuencia de números triangulares viene dado por:

$$t_n = 1/2*n*(n+1)$$

Por tanto, los diez primeros números triangulares son:

1, 3, 6, 10, 15, 21, 28, 36, 45, 55, …

Convirtiendo cada letra de una palabra en un número que corresponda a su posición alfabética y sumando dichos valores se obtiene el "valor de la palabra". Por ejemplo, el "valor de la palabra" de SKY es 19 + 11 + 25 = 55 = t_{10}. Si el "valor de la palabra" es un número triangular, podemos decir que la palabra es una palabra triangular.

Cree una función que lea un fichero que contiene cerca de dos mil palabras típicas del inglés y devuelva el número de palabras triangulares que contiene dicho fichero.

Suma de primos consecutivos

El número primo 41 se puede escribir como la suma de seis números primos consecutivos:

$$41 = 2 + 3 + 5 + 7 + 11 + 13$$

Esta es la suma más larga de primos consecutivos que es igual a un primo, por debajo del 100.

La suma más larga de primos consecutivos, por debajo del 1000, que es igual a un primo contiene 21 términos, y es igual a 953.

Cree una función que calcule el número primo, por debajo del millón, que es el resultado de la suma más larga de primos consecutivos.

¡Feliz no cumpleaños!

Cuando Alicia se encontró con el Sombrerero Loco, la Liebre de Marzo y el Lirón, los tres estaban en mitad de la celebración de una fiesta de no cumpleaños. Al principio Alicia no entendía nada, pero, gracias a las explicaciones del Sombrerero Loco, finalmente entendió que un no cumpleaños es una fecha que no coincide con la del cumpleaños. Alicia cayó en la cuenta de que también ese día era su no cumpleaños. ¡Qué pequeño es este mundo!

La dificultad del no cumpleaños es ¡saber cuántos se cumplen! Por eso, para no complicarse, en el País de las Maravillas las tartas solían tener siempre una sola vela. Pero a la Reina de Corazones no le gustaba, y, cuando decidió cortar la cabeza a cualquiera que no pusiera el número correcto de velas, se dejaron de celebrar fiestas de no cumpleaños.

Cree una función la cual reciba dos parámetros, el primero deberá ser un string con la fecha de nacimiento de alguno de los habitantes del País de las Maravillas, y el segundo, otro string con la fecha actual que siempre será posterior. Cada fecha estará compuesta del día, mes y año, separados por un espacio. La segunda fecha (la actual) no tendrá por qué ser un día de no cumpleaños. Tenga en cuenta, que el anciano del País de las Maravillas nació en 1862, no hay ninguna consulta que supere el año 9999, y en el País de las Maravillas no existen los años bisiestos.

Su función deberá devolver cuántos no cumpleaños cumple el personaje en la fecha actual. Si la fecha actual no es un no cumpleaños, deberá devolver un 0.

Ejemplos:

Input: “4 7 1862”, “24 5 1865”

Output: 1052

Input: “17 2 2014”, “17 2 2015”

Output: 0