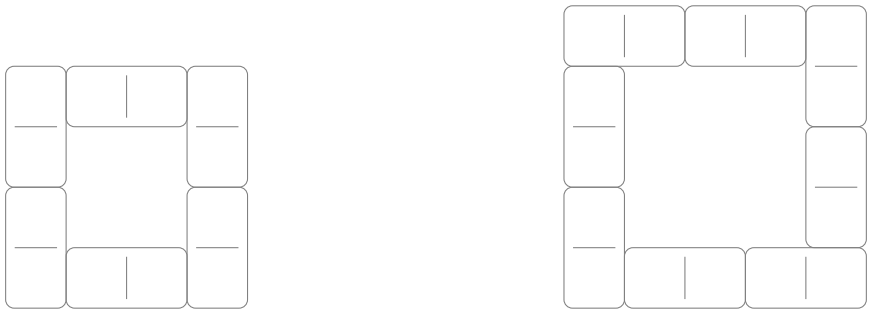
Problema
Dominó

Un juego de fichas de dominó consiste en un conjunto de piezas rectangulares de 2×1, donde cada mitad contiene un número entre 0 y $n-1$. Llamaremos un n-dominó al juego de fichas donde el valor correspondiente es n. Por ejemplo, el juego de dominó normal es un 7-dominó y sus piezas tienen valores entre 0 y 6. Además, en un n-dominó cada combinación de pares de números está presente exactamente una vez (1–3 y 3–1 son la misma pieza). Por ejemplo, un **7-dominó** consiste en **28 piezas**.

Un n-dominó es tan versátil como una baraja de cartas, en cuanto a que ambos permiten jugar una gran variedad de juegos. Uno de estos es el del cuadrado[1]. En el juego del cuadrado se te entrega un n-dominó y un valor k. El objetivo del juego es formar un cuadrado usando todas las piezas del n-dominó, de forma que para cada lado la suma de todos los valores en él sea k. A un cuadrado formado con un n-dominó de forma que los lados sumen k lo llamaremos un cuadrado (n, k).

Para formar un cuadrado válido la cantidad de pieza verticales y horizontales debe ser la misma. A continuación se muestra una imagen de un cuadrado inválido y uno válido. Notar que no para cualquier



(a) Cuadrado inválido (b) Cuadrado de lado 5

Figura 1: Un cuadrado válido y uno inválido.

valor de n es posible formar un cuadrado con un n-dominó. Por ejemplo, el 2-dominó está formado por 3 piezas (0–0, 0–1 y 1–1) y no es posible armar un cuadrado con ellas. Por otro lado, para un n-dominó donde si es posible formar un cuadrado no necesariamente es posible armar un cuadrado (n, k) para cualquier k. Por ejemplo, para el 7-dominó la suma de los lados siempre sera menor que 90, y por lo tanto no es posible formar un cuadrado $(7, 90)$.

Tu objetivo es, entonces, determinar una configuración de piezas para completar el cuadrado (n, k). Para esto, tendrás que implementar algunas funciones que simplificarán enormemente la resolución del problema.

[1] Inspirado en *Matemáticas Recreativas* de Y. I. Perelman

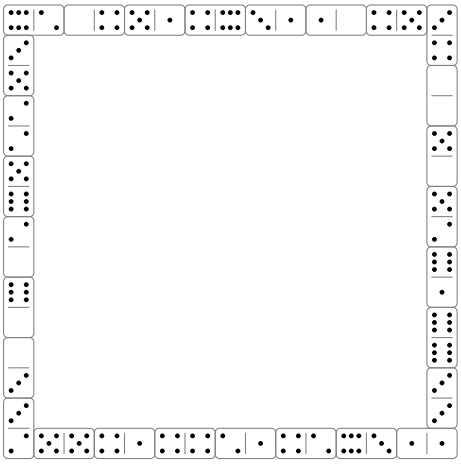


Figura 2: Un cuadrado $(7, 45)$. Notar que cada lado suma 45.

Subtarea 1

La primera subtarea consiste en implementar la función `cuadrado` que determina para un n si es posible formar un cuadrado con un n-dominó. En esta subtarea no importa la suma de los valores de cada lado, sólo se pide determinar si es posible formar un cuadrado válido usando todas las piezas del n-dominó.

- `cuadrado(n)`
  - `n`: el juego de n-dominó.
  - *return*: un booleano con valor `true` si se puede formar un cuadrado, y `false` si no.

Subtarea 2

La subtarea 2 consiste en implementar la `validar` que dado un cuadrado debe validar que este cumple con la restricción de que todos los lados sumen lo mismo. Esto también implica verificar que las fichas correspondan a las del juego completo de n-dominó. Sin embargo, puedes asumir que el juego completo de $n-$dominó sí permite formar un cuadrado.

- `validar(n, d)`
  - `n`: el juego de n-dominó.
  - `d`: vector de $2f$ enteros, donde se guarda el orden de las fichas que forma el cuadrado.
  - *return*: un booleano con valor `true` si los cuatro lados suman lo mismo, y `false` si no.

Subtarea 3, 4 y 5

En las subtareas 3, 4 y 5, debes implementar la función `construir` que para n y k construye, si es posible, un cuadrado (n, k), y retorna `true` si es posible o `false` si no.

- `construir(n, k, d)`
  - `n, k`: valores para n, k, que indican el cuadrado (n, k) que se quiere construir.
  - `d`: vector de $2f$ enteros, donde usted debe guardar los valores de las fichas de un cuadrado (n, k), partiendo desde alguna esquina. Observa que f es el número de piezas de dominó.
  - *return*: un booleano con valor `true` si se puede formar un cuadrado (n, k), y `false` si no.

Convención sobre el formato del arreglo *d* Para las funciones `validar` y `construir`, respectivamente al revisar y llenar el arreglo `d`, considera siempre que la posición 0 es una esquina, y que el resto se enumera en un sentido tal que los pares de valores de una misma ficha siempre quedan en posiciones adyacentes en el arreglo. El orden es fundamental para que los códigos puedan ser evaluados correctamente.

Por ejemplo, la figura 2 puede ser guardada en el arreglo de la siguiente manera (partiendo de la esquina superior izquierda):

$$6, 2, 0, 4, 5, 1, 4, 6, 3, 1, 1, 0, 4, 5, 3, 4, 0, 0, \ldots, 1, 4, 5, 5, 2, 3, 3, 0, 0, 6, 0, 2, 6, 5, 2, 2, 5, 3$$

Subtareas y Puntaje

25 puntos Implementar la función cuadrado, para $n \leq 1000$.

20 puntos Implementar la función validar, para $n \leq 1000$.

10 puntos Implementar la función construir, con $n \leq 7$.

20 puntos Implementar la función construir, para $n \leq 16$.

25 puntos Implementar la función construir, para $n \leq 1000$.