Juego de Saltos

NOTA: Si usted está leyendo este documento sin haber extraído el compactado que descargó del sitio, ciérrelo ahora, extraiga todos los archivos en el escritorio, y siga trabajando desde ahí. Es un error común trabajar en la solución dentro del compactado, lo cual provoca que los cambios **no se guarden**. Si usted comete este error y entrega una solución vacía, no tendrá oportunidad de reclamar.

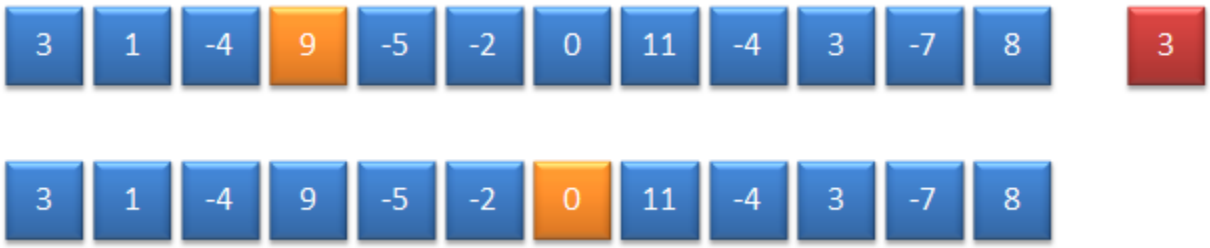
Se tiene un *array* de N posiciones, donde cada posición tiene un valor entero. El jugador tiene una ficha ubicada en la primera posición del *array,* y un dado con valores entre 1 y K, que puede lanzar una cierta cantidad de veces. En cada turno se tira el dado y se desplaza la ficha del jugador tantas posiciones como indique el dado. Posteriormente, se avanza la ficha hacia adelante (aumentando el índice en el *array*) o hacia atrás, una cantidad de pasos igual al valor del *array* en dicha posición, según si el valor es positivo o negativo. En ese punto, termina el turno, y se vuelve a lanzar el dado. El juego termina cuando se han realizado todos los lanzamientos, o cuando se cae en una posición fuera del *array*.

El problema consiste en, dado el *array* del juego, y el valor obtenido en cada lanzamiento del dado, determinar la posición en la que termina el jugador. Esta posición puede ser un valor entre 0 y N-1 si el jugador termina dentro del *array*, el valor -1 si el jugador se sale del *array* por la izquierda, o N si se sale del *array* por la derecha.

Por ejemplo, supongamos un juego donde se lanzó el dado 4 veces, con los valores 5, 3, 4, 1, dondese emplea el siguiente *array*:

El jugador comienza en la posición 0. En La primera tirada del dado se obtiene el valor 5. El jugador avanza hasta la posición 5, que tiene valor -2, y por tanto retrocede 2 espacios, terminando en la posición 3:

En este punto, se vuelve a lanzar el dado, y se obtiene el valor 3. Se avanza hasta la posición 6, cuyo valor es 0, por tanto, no se realiza ningún desplazamiento a partir de esta posición:

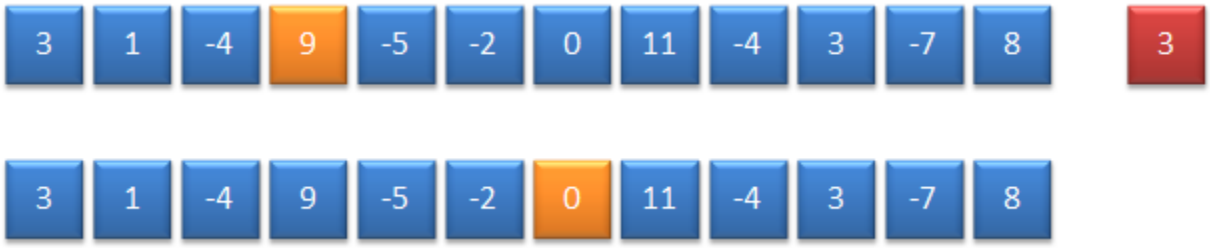


En este punto, se vuelve a lanzar el dado, y se obtiene el valor 4, por lo que se avanza a la posición 10, y de ahí se retrocede hasta la posición 3:

En este punto se lanza el dado por última vez, y se obtiene el valor 1. El jugador se desplaza hacia la posición que tiene valor -5, por lo que se retroceden 5 posiciones, saliendo del *array* por la izquierda:

En este punto se acaba el juego, independientemente de si quedan tiradas de dado disponibles o no, debido a que se salió por el borde del *array*. La respuesta para este caso es -1.

Si en cambio en la última tirada se hubiera obtenido, por ejemplo, el valor 3 en vez del valor 1, el jugador se hubiera desplazado a la posición 6, con valor 0, y no se hubiera movido. En ese momento, como se agotaron la cantidad de tiradas de dado, se termina el juego y la respuesta hubiera sido 6:



Por otro lado, si la última tirada hubiera sido 4, en vez de 1, se hubiera caído en la posición 7, con valor 11, y se hubiera salido por el extremo derecho del *array*:

En ese caso la respuesta hubiera sido 12, que es la cantidad de elementos del *array,* aun cuando la posición en que se debería haber caído es la 18. Es decir, si se sale por alguno de los bordes, se devuelve -1 o N, sin importar cuan "lejos" se caiga del borde del *array* correspondiente.

Por último, en caso de que el valor de la última tirada hubiera sido mayor que 8, se hubiera salido directamente por el extremo derecho del *array*, devolviendo también como resultado 12.

Usted debe implementar el método `PosicionFinal` dentro de la clase `Juego`.

```
public class Juego
{
    public static int PosicionFinal(int[] array, int[] tiradas)
    {
        // Borre esta línea e implemente su código aquí
        throw new NotImplementedException();
    }
}
```

Junto a este documento usted debe haber recibido una solución de Visual Studio 2010, con dos proyectos. El primero consiste en una biblioteca de clases que tiene el fragmento de código anterior. Usted debe programar su solución en ese proyecto, borrando la línea señalada, y escribiendo su código dentro del método `PosicionFinal`. Si lo necesita, usted puede adicionar tantos métodos como desee, pero garantice que la solución del ejercicio se obtenga llamando al método `PosicionFinal`. Además encontrará un proyecto de consola, con algunos casos de ejemplo, y los resultados esperados. La aplicación de consola brinda un método `Probar(int[] array, int[] tiradas, int correcto)`, que ejecuta su implementación, e imprime en consola si el resultado es el correcto, o no. Los ejemplos mostrados en este documento se brindan en la aplicación de consola, junto al valor de resultado correcto:

```
// Casos del documento
Probar(new[] { 3, 1, -4, 9, -5, -2, 0, 11, -4, 3, -7, 8 }, new[] { 5, 3, 4, 1 }, -1);
Probar(new[] { 3, 1, -4, 9, -5, -2, 0, 11, -4, 3, -7, 8 }, new[] { 5, 3, 4, 3 }, 6);
Probar(new[] { 3, 1, -4, 9, -5, -2, 0, 11, -4, 3, -7, 8 }, new[] { 5, 3, 4, 4 }, 12);
Probar(new[] { 3, 1, -4, 9, -5, -2, 0, 11, -4, 3, -7, 8 }, new[] { 5, 3, 4, 9 }, 12);
```

Es su responsabilidad adicionar tantos ejemplos como considere necesario, para garantizar la validez de su solución.

Se garantiza que ninguno de los argumentos del método `PosicionFinal` es `null` ni está vacío. Usted no necesita validar estos parámetros.

Recuerde guardar a menudo para evitar, entre otras complicaciones, pérdidas de información por causa de fallos en el fluido eléctrico.