Resolución ejercicio "Espiral galáctica"

Abstracción procedimental

Luego de una lectura profunda y el entendimiento del enunciado, se analiza las entradas y salidas esperadas del algoritmo a confeccionar. Se aclara que se puede pedir que se analizen varias matrices y que para cada una se imprima el resultado correspondiente.

Respecto a la entrada, se puede decir que la primera línea es un número N que es impar y menor a 100, que nos indica el tamaño de la matriz; luego, aparecen línea por línea los valores que describen cada fila de la matriz, separados por un espacio cada elemento. Finalmente, cuando se ingrese un 0 indicando el tamaño de la matriz, el algoritmo finaliza y debe imprimir las salidas.

Respecto a la salida, se solicita que se imprima en pantalla la cantidad de estrellas de cada galaxia ingresada, en orden y un resultado por línea.

Subproblema 1: Crear matriz a partir de valor ingresado por usuario

Considerando que el valor N ingresado por el usuario es impar y mayor a 0 pero menor a 100, se debe crear una matriz donde se almacenen los valores a partir de eso.

Subproblema 2: Cargar valores a cada fila de la matriz

Luego del valor N que ingresa el usuario, inmediatamente ingresa fila tras fila que describe los valores de la matriz, considerando que los valores estarán separados por espacios entre ellos.

Por ejemplo:

3 → N
1 2 3 → Fila 1
4 5 6 → Fila 2
7 8 9 → Fila 3

Como se puede apreciar en el ejemplo, los valores están separados por espacios.

Subproblema 3: Encontrar en centro de la matriz

Por los ejemplos provistos, se puede ver que el movimiento de la galaxia siempre parte desde el centro de la matriz, o sea, del elemento central. De este modo, hay que encontrar una forma de conocer el centro de la matriz cuadrada de dimensión impar, sin importar su dimensión.

Subproblema 4: Calcular cantidad de estrellas de matriz

Una vez se tiene la matriz creada y cargada, se procede a hacer el análisis correspondiente para averiguar la cantidad de estrellas que hay en la misma.

Se identifica un patrón en el camino de la espiral galáctica, donde se hacen movimientos hacia arriba, a la derecha, hacia abajo y a la izquierda (en ese orden) hasta que hay un movimiento final en el que se llega a los límites de la galaxia y termina.

Patrón identificado:

1. Movimiento arriba,
2. Movimiento derecha,
3. Movimiento abajo,
4. Movimiento izquierda,
5. Se repiten los pasos 1 a 4, hasta que se encuentra el límite de matriz y termina.

Los movimientos van incrementando "el paso", donde siempre se comienza con un movimiento de 1 elemento hacia arriba, luego 2 hacia la derecha, 3 hacia abajo, 4 a la izquierda, 5 arriba nuevamente, y así sucesivamente. Vale aclarar, de todos modos, que el movimiento final puede no corresponder con el valor del paso, ya que si se respetará estaría por fuera de la matriz.

2	1	2	2	3	2	3
4	2	4	4	4	4	4
3	2	3	3	4	5	3
4	5	2	2	7	1	2
5	2	1	3	5	3	6
4	2	4	3	1	2	0
3	9	9	9	9	2	9

Si se analiza este ejemplo, se ve que los movimientos son:

- Arriba 1
- Derecha 2
- Abajo 3
- Izquierda 4
- Arriba 5
- Derecha 5

Si se pone atención al último movimiento, se ve que no es Derecha 6 debido a que si así fuera, se iría por fuera de la matriz, como se comentó. Por ende, eso es algo a tener en cuenta a la hora de analizar.

Subproblema 5: Generar salidas de varias matriz en formato esperado

Una vez se tiene el valor de cantidad de estrellas hay que almacenarlo de alguna forma para luego mostrarlo en la salida junto al resto de resultados de los otros análisis; todo esto, respetando el formato esperado: un resultado por línea y ordenados por matriz ingresada.

Resolución

Resolución subproblemas 1 y 2

Se arma un *ciclo mientras* donde se revisa la condición que el número N sea distinto a 0, si es distinto, se pide el input que indica el tamaño de la siguiente matriz a analizar.

Se eligió pedir el input de la fila con los números separados por espacios y a este input procesarlo con la función separarNumeros().

Explicación de la función separarNumeros(cadena):

- La función toma una cadena como parámetro.
- Luego, se inicializa una variable *nums* a 0 y se recorre la cadena, retornando la cantidad de números que posee la fila.
    - Este análisis lo hace mediante la búsqueda de *espacios en blanco y el último símbolo* de la cadena.
- Se crea un vector de dimensión igual a la variable *nums*.
- Se inicializa una variable local *indice*.
- Se recorre nuevamente la cadena en su totalidad, donde si el símbolo en posición *i* es distinto al espacio, se agrega el símbolo al vector en posición *indice*; caso contrario, si nos encontramos en el último símbolo de la cadena o encontramos un espacio, incrementamos el contador en uno.
- Se retorna el vector con los números cargados.

Luego de que se pidió el input de la fila, se pone el vector con números en la matriz con la función de Python *.append()*. Recomendamos no hacer uso de esta función en la resolución de problemas algorítmicos ya que esto puede ser resuelto con estructuras de datos. Fue resuelto de esta manera por cuestiones de rapidez simplemente.

Este proceso se realiza N veces, o sea, por N filas.

Resolución subproblema 3

Para el tipo de matrices con las que se va a trabajar, o sea, matrices cuadradas de dimensión impar, se puede encontrar el elemento del centro de diversas maneras:

- (Dimensión - 1) / 2
- (Dimensión / 2) + 0.5

De esta manera, con una matriz 3x3, el centro va a ser el elemento en fila 2 y columna 2. En una 5x5, el centro estará en fila 3 y columna 3, partiendo desde 1.

$$\frac{3}{2} + 0.5 = 2$$

$$\frac{5}{2} + 0.5 = 3$$

$$\frac{3-1}{2} = 2$$

$$\frac{5-1}{2} = 3$$

Esta lógica será utilizada en la función *contadorEstrellas()*, que será explicada a continuación.

Resolución subproblema 4

Como se comentó en un principio, hay un patrón identificado en el movimiento de la galaxia de acuerdo a la matriz y su dimensión. Lo que se hace es crear una función *contadorEstrellas()*.

- Se toma una matriz y su dimensión como parámetro.
- Se inicializan las variables correspondientes y se define el centro de la matriz, que será el punto inicial del recorrido que se hará para contar las estrellas.
- Se toman dos variables *fila* y *columna* que serán la referencia de dónde estamos parados en la matriz a medida que se va desarrollando el recorrido.
- Se toma una variable *contador* que indica la cantidad de elementos que recorre cada movimiento, es decir, comienza en 1 porque ese es primer movimiento, luego incrementa a 2 para el segundo, y así sucesivamente.

Se identifica lo siguiente respecto a los movimientos en la matriz:

➔ Arriba: Decremento la fila y mantengo la columna.

➔ Derecha: Mantengo la fila e incremento la columna.

➔ Abajo: Incremento la fila y mantengo la columna.

➔ Izquierda: Mantengo la fila y decremento la columna.

Se crea una porción de código que resuelve cada movimiento, y todo esto dentro de un *ciclo mientras*. La única manera que el algoritmo salga de este ciclo es que se encuentre con un elemento en "el borde" de la matriz, lo que hace que se ejecute la palabra reservada *break*, que hace que se salga inmediatamente del ciclo donde se está.

Previo a empezar a contar las estrellas de cada movimiento, se verifica una condición que permite saber si nos iríamos de la matriz en cada de que nos movamos la cantidad de elementos que nos dice la variable *contador*. Cada movimiento tiene una condición.

- ☐ Arriba: *valor de la fila donde nos encontramos - contador + 1 debe ser mayor o igual a 0*. De esta manera si es menor a 0, indica que nos moveríamos tan arriba que nos iríamos de la matriz, lo que generaría un error.

- ☐ Derecha: (*columna donde nos encontramos + contador - 1) debe ser menor o igual a (dimensión - 1).* De esta manera verificamos que el movimiento hacia la derecha no salga de la matriz por el valor de la columna.

- ☐ Abajo: *fila + contador - 1 debe ser menor a dimensión*. Así, podemos saber si nos salimos de la matriz por un movimiento hacia abajo.

- ☐ Izquierda: *columna - contador + 1 debe ser mayor a -1*. Se verifica, así, si en un movimiento a la izquierda nos saldríamos de la matriz.

Como se puede apreciar, las condiciones varían de acuerdo a los valores de *contador* y de *fila* o *columna*, o sea, depende de dónde estamos parados en la matriz y de cuánto nos deberíamos mover de acuerdo al patrón.

De todos modos, si la condición en el movimiento final no se cumple, lo que realizamos es movernos hasta el elemento que está en "el borde", y finalizamos el recuento de estrellas. Por ejemplo, si el contador dice que debemos movernos 6 elementos, pero la condición del movimiento no se cumple, nos movemos una cantidad n < 6 de elementos hasta llegar al elemento final previo a salirnos de la matriz.

2	1	2	2	3	2	3
4	2	4	4	4	4	4
3	2	3	3	4	5	3
4	5	2	2	7	1	2
5	2	1	3	5	3	6
4	2	4	3	1	2	0
3	9	9	9	9	2	9

En este caso de ejemplo, el último movimiento es hacia la derecha y tiene en cuenta 5 elementos. Si se analiza el valor de *contador* desde el principio, llegaríamos a la conclusión que en ese movimiento final *contador* vale 6, ya que en el movimiento anterior valía 5. De cualquier manera, se mueve 5 elementos porque es lo que puede, y finaliza la galaxia.

Poniendo números al último movimiento de ese caso, consideramos lo siguiente:

$$contador = 6$$

$$fila = 1$$

$$columna = 2$$

Si verificamos la condición del movimiento a la derecha: *columna donde nos encontramos + contador - 1) debe ser menor o igual a (dimensión - 1)*, y calculamos:

$$columna + contador - 1 <= dimensión - 1$$

$$2 + 6 - 1 <= 7 - 1$$

$$7 <= 6 \rightarrow \text{Falso}$$

Debido a que no se cumple la condición esperada, se mueve hasta el final, o sea, para este caso hasta la última columna y finaliza el recuento.

Si se analiza el código, mediante un *if* se verifica la condición de cada movimiento y si se cumple, se recorre la matriz normalmente como nos dicta el patrón y por cada elemento en el que vamos pasando, vamos incrementando el valor de la variable *estrellas*, que es una variable acumulador y es la respuesta a cada matriz. Si no se cumple la condición, como se dijo antes, nos movemos hasta el final, lo que significa cosas distintas para cada movimiento.

- Arriba: Ir desde la fila en la que estamos, hasta la fila 0.
- Derecha: Ir desde la columna en la que estamos, hasta la columna *dimensión - 1*.
- Abajo: Ir desde la fila en la que estamos, hasta la fila *dimensión - 1*.
- Izquierda: Ir desde la columna en la que estamos, hasta la columna 0.

Posteriormente, destacar que cuando terminamos el movimiento, tenemos que redefinir las variables *fila* y *columna* que utilizamos para saber dónde estamos en la matriz, por lo que se considerará lo siguiente en la solución propuesta:

2	1	2	2	3	2	3
4	2	4	4	4	4	4
3	2	3	3	4	5	3
4	5	2	2	7	1	2
5	2	1	3	5	3	6
4	2	4	3	1	2	0
3	9	9	9	9	2	9

Si ponemos como ejemplo el primer movimiento, del índice 33 al 23, o sea, del elemento que vale 2 al que vale 3 (arriba, donde contador = 1), contamos los elementos de ambos índices mediante un *ciclo para* y luego dejamos las variables *fila* y *columna* de modo que quedemos parados en el índice 24, o sea, el elemento que vale 4.

Si pusiéramos otro ejemplo, con el penúltimo movimiento que es hacia arriba, donde se parte del índice 51 y se finaliza en el índice 01, dejamos las variables *fila* y *columna* para que quedemos parados en el índice 02.

Teniendo en cuenta esto, se considera de forma general lo siguiente para setear las variables de posición luego de cada movimiento:

Movimiento ↓ \ Variable →	Fila	Columna
Arriba	fila - contador + 1	columna + 1
Derecha	fila + 1	columna + contador - 1
Abajo	fila + contador - 1	columna - 1
Izquierda	fila - 1	columna - contador + 1

Luego de haber hecho esto, debemos guardar el valor en alguna estructura para que persista en nuestro algoritmo y no se pierda en caso de que hayan varios casos de prueba. Cuando se completó la carga de la matriz (subproblema anterior), simplemente llamamos a la función *contadorEstrellas(matriz, dimensión)* y almacenamos el valor que devuelve (cantidad de estrellas de la matriz) en un vector llamado *vectorResultados*, que está sobredimensionado y posee 1000 elementos.

El proceso se repite por cada matriz ingresada por el usuario, donde una variable *indice* funciona como acumulador y va incrementando con cada matriz, para así guardar los resultados en el vector.

Resolución subproblema 5

Para finalizar, y una vez que se tienen los resultados que nos interesan en el vector *vectorResultados*, lo recorremos con un *ciclo para* y si el elemento en posición *i* es distinto a -1, se imprime en consola. Hacemos esta condición porque una matriz no puede tener una cantidad negativa de estrellas, y al crear el vector se lo crea con -1 en cada elemento. A medida que se cargan los resultados en el vector, se van pisando los -1 con los valores de cantidad de estrellas de cada matriz.

De esta manera, queda un total de líneas igual a la cantidad de casos de prueba, donde en cada una se muestra la cantidad de estrellas de cierta matriz. La impresión de resultados se hace respetando el orden de ingreso.