**#2 Resolución de Ejercicio OIA o similar (B)**

Resolución de ejercicio “Números Escondidos” con sus lotes de prueba en el archivo comprimido 35378581-CordobaFabianAriel.zip.

1. Diagrama de clases y responsabilidades
2. Documentación de 3 casos de prueba significativos:

(Los lotes de pruebas son han sido utilizados para testear el algoritmo)

|  |  |
| --- | --- |
| Caso 01: cifrasBuscadasEnDiferentesOrdenes | |
| Verifica que se considere cualquier orden de las cifras del número a ser buscado. | |
| cifrasBuscadasEnDiferentesOrdenes.in | cifrasBuscadasEnDiferentesOrdenes.out |
| 9  123231132  3  123 | SI 3  1 4 7 |

|  |  |
| --- | --- |
| Caso 02: mismasCifras | |
| Verifica que se considere como acierto cuando ambos números tengan exactamente las mismas cifras. | |
| mismasCifras.in | mismasCifras.out |
| 10  0123456789  10  0123456789 | SI 1  1 |

|  |  |
| --- | --- |
| Caso 03: todasLasPosiciones | |
| Verifica que se ubiquen todas las ocurrencias del número a ser ubicado. | |
| todasLasPosiciones.in | todasLasPosiciones.out |
| 10  1111111111  1  1 | SI 10  1 2 3 4 5 6 7 8 9 10 |

1. Complejidad computacional del algoritmo desarrollado:

Consideraciones:

N = cantidad de cifras del número dentro del cual se desea ubicar a la siguiente secuencia de cifras. (1<=N<=250)

M = cantidad de cifras de la segunda secuencia cifras (el número que se desea ubicar). (1<=M<=N)

Para obtener la complejidad computacional, obtenemos de cada subrutina del algoritmo su complejidad, aplicando las reglas para calcular cada Big O. Hacemos esto desde la lectura del archivo con los datos de entrada hasta la obtención del String con la salida correspondiente a dicha entrada. De cada subrutina solo expresaremos su complejidad más relevante.

Por el momento consideraremos el peor caso posible para cada subrutina, pero para una misma entrada, todas las condiciones no llevan a todas las posibles peores situaciones. Eso lo analizaremos luego de obtener a priori la complejidad resultante, para luego analizar qué sucede internamente, para así descubrir cual es el peor de los escenarios.

Los valores en cada tabla corresponden al orden de Big O.

El resultado de cada celda se obtiene como la suma o producto (según se indique) de las celdas debajo.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Complejidad del algoritmo (a priori):  (N-M)M2 | | | | | | | |
| suma | | | | | | | |
| Lectura de Archivo:  N + M | | Procesamiento de datos (cargar lista de posiciones de ocurrencias):  (N-M)\*M2 | | | | | Escribir String con la salida requerida:  N |
| suma | | producto | | | | |
| Cargar cifras del primer número:  N | Cargar cifras del segundo número:  M | Recorrido de secuencia donde se busca el numero:  N-M | Comparar partición de primer número con número buscado:  M2 | | | |
| suma | | | |
| Tomar M cifras del primer número:  M | Comparar las M cifras tomadas con las cifras del número buscado:  M2 | | |
| suma | | |
| Generar lista enlazada:  M | Recorrer M verificando si la lista enlazada la contiene:  M2 | |
| producto | |
| Recorrer cifras:  M | Evaluar si la lista enlazada contiene la cifra:  M |

Con esta tabla, observaremos casos extremos para quedarnos con la complejidad del peor escenario y así estimar mejor:

Caso A: 1 = M = N

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Complejidad del algoritmo:  1 | | | | | | | |
| suma | | | | | | | |
| Lectura de Archivo:  1 | | Procesamiento de datos (cargar lista de posiciones de ocurrencias):  1 | | | | | Escribir String con la salida requerida:  N = 1 |
| suma | | producto | | | | |
| Cargar cifras del primer número:  N =1 | Cargar cifras del segundo número:  M = 1 | Recorrido de secuencia donde se busca el numero:  1 | Comparar partición de primer número con número buscado:  1 | | | |
| suma | | | |
| Tomar M cifras del primer número:  M = 1 | Comparar las M cifras tomadas con las cifras del número buscado:  1 | | |
| suma | | |
| Generar lista enlazada:  M = 1 | Recorrer M verificando si la lista enlazada la contiene:  1 | |
| producto | |
| Recorrer cifras:  M=1 | Evaluar si la lista enlazada contiene la cifra:  M=1 |

Caso B: 1 < M = N

El tiempo empleado en procesamiento es prácticamente despreciable.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Complejidad del algoritmo:  N2 | | | | | | | |
| suma | | | | | | | |
| Lectura de Archivo:  N | | Procesamiento de datos (cargar lista de posiciones de ocurrencias):  N2 | | | | | Escribir String con la salida requerida:  N |
| suma | | producto | | | | |
| Cargar cifras del primer número:  N | Cargar cifras del segundo número:  M = N | Recorrido de secuencia donde se busca el numero:  1 | Comparar partición de primer número con número buscado:  N2 | | | |
| suma | | | |
| Tomar M cifras del primer número:  M=N | Comparar las M cifras tomadas con las cifras del número buscado:  N2 | | |
| suma | | |
| Generar lista enlazada:  M=N | Recorrer M verificando si la lista enlazada la contiene:  N2 | |
| producto | |
| Recorrer cifras:  M=N | Evaluar si la lista enlazada contiene la cifra:  M=N |

Caso C: 1 = M y 1<<N

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Complejidad del algoritmo:  N | | | | | | | |
| suma | | | | | | | |
| Lectura de Archivo:  N | | Procesamiento de datos (cargar lista de posiciones de ocurrencias):  N | | | | | Escribir String con la salida requerida:  N |
| suma | | producto | | | | |
| Cargar cifras del primer número:  N | Cargar cifras del segundo número:  1 | Recorrido de secuencia donde se busca el numero:  N | Comparar partición de primer número con número buscado:  1 | | | |
| suma | | | |
| Tomar M cifras del primer número:  1 | Comparar las M cifras tomadas con las cifras del número buscado:  1 | | |
| suma | | |
| Generar lista enlazada:  1 | Recorrer M verificando si la lista enlazada la contiene:  1 | |
| producto | |
| Recorrer cifras:  1 | Evaluar si la lista enlazada contiene la cifra:  1 |

Observamos que el caso B es donde se obtiene la complejidad de mayor orden, por lo tanto, el peor caso.

Diremos entonces que la complejidad computacional de nuestro algoritmo es la obtenida en el caso B.

**Complejidad del algoritmo: O(N2**), con N = la cantidad de cifras del numero en el cual se desea ubicar a la segunda secuencia de cifras. Es decir, N es la cantidad de cifras del primer número en el archivo de entrada.

El tamaño máximo de entrada para un algoritmo con O(N2), en un tiempo razonable, es del orden de 104. Es decir que este algoritmo podrá resolver una entrada de dicho orden en un tiempo razonable.

1. Explicar estrategia general y estructuras de datos
2. Dada una entrada provista por la cátedra, se ha obtenido la salida de forma teórica y comprobado la misma con el algoritmo desarrollado.

Entrada:

|  |
| --- |
| 23  12341324213423143214312  3  123 |

Salida:

|  |
| --- |
| SI 6  1 5 9 13 17 21 |