Corrección de examen

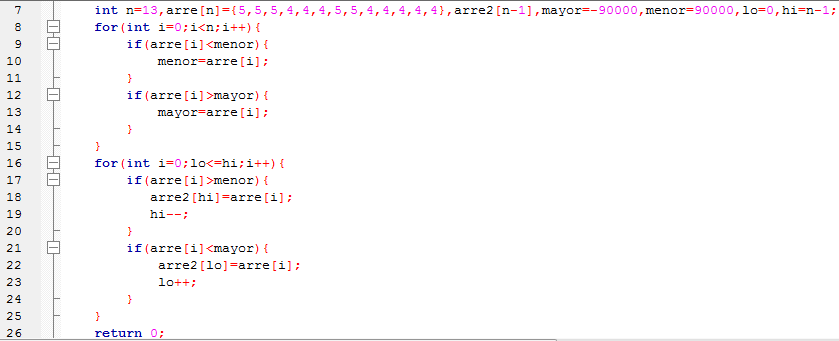
Análisis de algoritmos

Esteban Xelhuantzi Perla

324249

Profesor: Luis Carlos Gonzales Gurrola

1.



Lo que hace el primer ciclo “for” que se compone de las líneas 8 a 15 es determinar el menor y mayor de ambos números con condiciones,

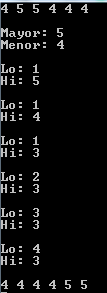
1. De la línea 9 a la 11 lo que se hace en ese condicional es determinar el numero menor de ambos números, al estar iterando el ciclo siempre va a estar comparando los valores en la posición actual con un valor muy grande, así cuando encuentre un valor más pequeño, se actualiza y garantizaremos tener el valor menor del arreglo
2. De la línea 12 a la 14 lo que se hace es determinar el número mayor de ambos números, al estar iterando el ciclo siempre va a estar comparando los valores en la posición actual con un valor muy pequeño el cual va a ser llamado “mayor”, así cuando encuentre un valor más grande, lo actualiza, y así garantizaremos tener el valor mayor del arreglo

El segundo ciclo “for” que se compone de las líneas 16 a 25 es el ordenamiento de los números, al estar iterando pueden entrar cualquiera de las siguientes 2 condiciones:

1. La primera condición la cual está entre las líneas 17 y 20 va comparando los números con la variable “menor”, si esta condición se cumple, es decir, si el numero del arreglo en la posición “i” es mayor a la variable “menor”, este número se colocara en un segundo arreglo el cual es del mismo tamaño, después de colocarlo se modifica el valor de la variable “hi”, la cual va a ir disminuyendo (cada que la condición se cumple) de uno en uno
2. La segunda condición la cual está entre las líneas 21 y 24 va comparando los números con la variable “mayor”, si esta condición se cumple, es decir, si el numero del arreglo en la posición “i” es menor a la variable “mayor” este numero se colocará en el mismo segundo arreglo creado, después de colocarlo se modificara el valor de la variable “lo”, la cual va a ir aumentando (cada que la condición se cumple) de uno en uno

Al hacer análisis de tiempo de este algoritmo determinamos que es ya que la línea 8 se va a ejecutar n+1 veces, y las condicionales se van a ejecutar O(1), y la cabecera del ciclo de la línea 16 se ejecutara veces, las cabeceras se ejecutaran n veces, y lo que está dentro será O(1), por lo tanto es O(n).

EJEMPLO DEL ALGORITMO DE ORDENAMIENTO:



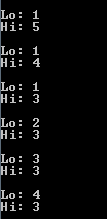
Como podemos ver en la parte superior de la imagen aparece el arreglo desordenado:



Después podemos ver como ya fueron determinados los valores mayores y menores, como se propuso en el primer ciclo “for”:



En las siguientes líneas podemos observar como se van cambiando los índices que tenemos, esto quiere decir que ya se esta empezando a ordenar el arreglo, como se propuso en el segundo ciclo “for”

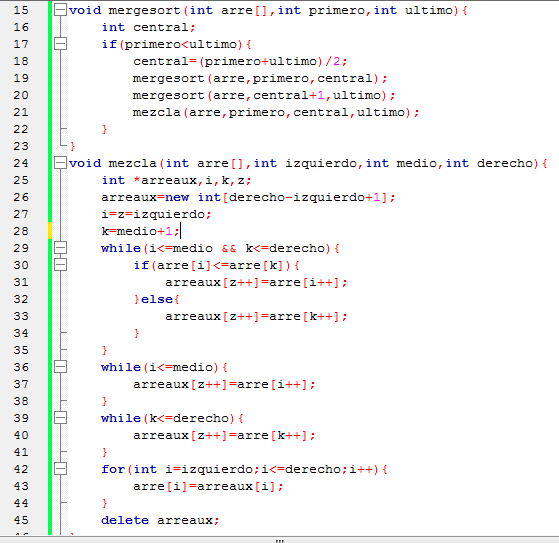


En la parte inferior de la imagen, se puede observar el arreglo ordenado



1.1 El algoritmo lineal es más rápido comparado con el nlogn porque el lineal esta hecho totalmente para este problema en particular, el algoritmo lineal crece mas lento que el nlogn ya que este otro algoritmo que tarda nlogn está diseñado para ordenar cualquier tipo de números

2.



Línea 16: variable para guardar la posición central del arreglo

Línea 17: Condicional para comenzar a dividir, si primero es menor a ultimo, aun se puede dividir el arreglo

Línea 18: Determinamos central, que es primero más ultimo sobre 2

Línea 19: Al volver a llamar a sí misma, la función mergesort va a comenzar a ordenar la sub lista izquierda, para limitar la sub lista izquierda, este va a estar delimitado por que central va a ser nuestro nuevo ultimo.

Línea 20: Al volver a llamar a sí misma, la función mergesort va a comenzar a ordenar la sub lista derecha, para limitar la sub lista derecha, central +1 va a ser nuestro nuevo primero, y ultimo seguirá igual.

Línea 21: Se llama a la función mezcla, esta función será la encargada de ordenar y unir las sub listas, la cual va a recibir el arreglo, el índice izquierdo, medio y derecho, los cuales nos ayudaran a ir posicionándonos.

Línea 25: Creamos una variable puntero llamada “arreaux”, y las variables i,k,z, “i” controla la sub lista izquierda y “k” controla la sub lista derecha.

Línea 26: Creamos un arreglo usando la variable “arreaux” que sea del tamaño necesario para los elementos.

Línea 27: Le damos el valor de “izquierdo” a las variables “i” y “z”.

Línea 28: Le damos el valor de medio + 1 a “k”.

Línea 29: Este ciclo comenzara a ordenar los elementos de nuestro arreglo, este va a funcionar mientras “i” sea menor o igual a medio y “k” sea menor o igual a derecha.

Línea 30: Este condicional nos dice que si el elemento de la sub lista izquierda es menor o igual al elemento de la sub lista derecha, colocaremos ese elemento en el arreglo auxiliar como aparece en la línea 31.

Línea 32: Este “else” nos dice que caso contrario (Que el elemento de la sub lista derecha es el menor) es el que se va a copiar

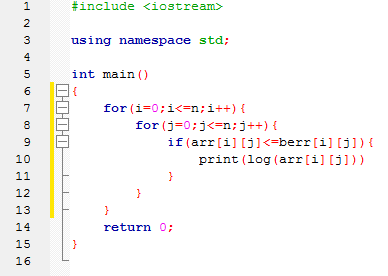
Línea 36: Este “while” nos sirve para cuando los elementos de la sub lista derecha ya se terminó y aún quedan elementos de la sub lista izquierda, solamente copiaremos los elementos restantes como se representa en la línea 37

Línea 39: Este “while” nos sirve para cuando los elementos de la sub lista izquierda ya se terminaron y aún quedan elementos de la sub lista derecha, solamente copiaremos los elementos restantes como se representa en la línea 40.

Al llegar a este punto, nuestro arreglo auxiliar ya esta ordenado, por lo tanto, vamos a reemplazar los valores en nuestro arreglo original como se representa en las líneas 42, 43 y 44

Por último, en la línea 45 borramos nuestro arreglo auxiliar.

3.



La cabecera de la linea numero 7 se va a ejecutar n+1 veces.

La cabecera de la linea numero 8 se va a ejecutar n(n+1) veces (.

La cabecera de la linea numero 9 se va a ejecutar n veces

La linea numero 10 se ejecutara dependiendo de si entra la condicion del “if” de la linea 9, por lo tanto es un , .

a)El mejor tiempo seria cuando la cabecera del “if” en la linea numero 9 nunca se cumpla, por lo tanto seria un =

b) el peor tiempo es cuando siempre se cumpla la condicion del “if” de la linea numero 9, por lo tanto seria un =

4.

Ejercicios notación Big O

1: =

Paso 1:

Paso 2:

Paso 3:

Por lo tanto, c1=36 pero no se va a cumplir hasta que

= , si se cumple.

2: 6

Paso 1: 6

Paso 2:

Paso 3: 6

Se determina que esta desigualdad se cumple a partir de y que C1 sea mayor o igual a 7 (En este ejemplo utilizaremos el 7).

Paso 4: 6

Paso 5: 6.525 7.125

Por lo tanto, si se cumple.

3:

Paso 1: C1

Paso 2:

Paso 3: 5C1

Por lo tanto, C1 puede valer 1, ya que siempre se va a cumplir.

5

4:

Paso 1:

Paso 2:

Paso 3:

Para que esta desigualdad se cumpla, C1 tiene que ser mayor o igual a 3(En este ejemplo usaremos él 3) y debe ser mayor o igual a 10.

Paso 4:

Por lo tanto, si se cumple

5:

Paso 1:

Paso 2:

Paso 4:

Por lo tanto, C1 debe ser mayor o igual a 7(En este ejemplo utilizaremos el 7), pero esta desigualdad se va a cumplir a partir de .

Paso 5 =

Pudimos comprobar que si se cumple.

6:

Paso 1:

Paso 2:

Paso 3:

Podemos observar que esta desigualdad no se va a cumplir nunca, ya que a partir de cierto valor de se va a superar el valor de la constante.

7:

Paso 1:

Paso 2:

Paso 3:

Por lo tanto, C1 puede valer 1, ya que este valor siempre va a cumplir la condición y .

Paso 4:

Demostramos que si se cumple.

8:

Paso 1: C1

Paso 2: C1

Paso 3:

Paso 4: C1

Podemos observar que esta condición se va a cumplir si C1 es mayor o igual a 4, para este ejemplo tomaremos el 4, y se empezara a cumplir a partir de .

Paso 5: 4 = 3.3703

Demostramos que si se cumple.

9:

Paso 1:

Paso 2:

Paso 3:

Dada esta condición, podemos determinar que nunca se cumplirá, ya que mientras mas suba el valor de n, esta terminará rebasando a la constante.

10:

Paso 1:

Paso 2:

Paso 3:

Dada esta condición, determinamos que nunca va a cumplirse para ninguna constante, ya que entre mas suba el valor de n, este terminara superando el valor de cualquier constante.

Ejercicios notación Ω

1:

Paso 1:

Paso 2:

Paso 3:

Dadas las condiciones, podemos observar que nunca va a ser cierta, si a C1 le damos el valor de 1, este valor no va a ser superado cada vez que n crezca

Comprobamos que no se cumple.

2:

Paso 1:

Paso 2:

Paso 3:

Podemos observar que las condiciones pueden ser cumplidas si C1 vale 1, y que n es igual o mayor a 2

Comprobación:

Se demostró que si cumple la condición.

3:

Paso 1:

Paso 2:

Paso 3:

Al observar las condiciones, podemos observar que C1 puede valer 1, ya que este valor siempre va a cumplir las condiciones dadas y que

Comprobación:

Se demostró que se cumple la condición.

4:

Paso 1:

Paso 2:

Paso 3:

Observando las condiciones, podemos determinar que C1 puede tomar varios valores menores o iguales a 1, en este caso usaremos , y nos damos cuenta de que

=

Podemos ver que se cumple la condición.

5:

Paso 1:

Paso 2:

Paso 3:

Dada esta condición, podemos determinar que C1 debe valer para que esta condición sea verdadera, y también determinamos que para que esta condición sea verdadera

Comprobación: n = 2, c =

=

Comprobamos que si se cumple.

=

6.

Paso 1:

Paso 2:

Paso 3:

Por las condiciones que se nos presentan, podemos determinar que C1 puede valer para poder cumplir, pero ya que, a partir de ese valor, es cuando la desigualdad comienza a ser verdadera.

Comprobación:

=

Comprobamos que si se cumple.

7.

Paso 1:

Paso 2:

Paso 3:

Dadas las condiciones, podemos observar que nunca se va a poder cumplir.

8.

Paso 1:

A partir de este punto, podemos fijarnos en que para que se cumpla la condición, debemos hacer que la constante valga y que

Paso 2:

Comprobación: n=2

4

Comprobamos que la condición si se cumple.

9.

Paso 1:

A partir de este punto, nos podemos dar cuenta de que C1 puede valer 1, ya que siempre va a cumplir la condición y n podrá tomar cualquier valor.

Comprobación: n=2

(

Se comprobó que la condición si se cumple.

10.

Paso 1:

A partir de este punto, nos podemos dar cuenta de que la condición nunca será verdadera, ya que independientemente de que valor le demos a C1, al ser una siempre terminará superando a la función

Ejercicios Notación Theta

1:

Paso 1:

Paso 2 (comprobar omega):

A partir de este punto podemos observar que independientemente de que valor le demos a C1, tarde o temprano la desigualdad no se va a cumplir conforme n crezca, por lo tanto, no es necesario demostrar Big O y demostramos que no pertenece a la familia de .

2.

Paso 1: (

Paso 2 (comprobar omega):

Al observar la desigualdad podemos comprobar que no existe una constante que satisfaga a la desigualdad, ya que mientras mas crezca n, tarde o temprano la desigualdad no se cumplirá, por lo tanto no pertenece a la familia de

3.

Paso 1: (

Paso 2 (comprobar omega):

,

Al ver la desigualdad, podemos determinar que C1 puede valer 1 y se cumplirá para

Paso 3 (Comprobar Big O): (

Al llegar a este punto podemos determinar que no importa que valor le demos a la constante, si n crece, llegará un punto en el que la desigualdad no se cumplirá, así que no pertenece a la familia de

4.

Paso 1: (

Paso 2 (comprobar omega):

Al ver la desigualdad, podemos determinar que C1 puede valer 1, ya que se va a cumplir para cualquier valor de n.

Paso 3 (Comprobar Big O): (

Al ver esta desigualdad, podemos determinar que C2 puede tomar el valor de 7, para satisfacer la desigualdad para cualquier valor de n

Comprobación: n=2

(=

Comprobamos que si pertenece a la familia de

5.

Paso 1: (

Paso 2(Comprobar omega):

Al ver esta condición, podemos determinar que C1 vale para poder satisfacer en cualquier valor de n

Paso 3(Comprobar Big O): (

Al ver esta ultima desigualdad, podemos determinar que no importa que valor le demos a C2, este valor tarde o temprano será superado cuando n crezca, por lo tanto no pertenece a la familia de

6.

Paso 1: (

Paso 2 (Comprobar Omega):

Al ver esta condición, podemos determinar que C1 puede valer , ya que este valor siempre va a servirá para cualquier valor de n, incluido

Paso 3 (Comprobar Big O): (

Al ver esta condición, podemos determinar que C2 puede valer 2, ya que este valor va a satisfacer la condición para cualquier valor de n, incluido

Comprobación: n=9

(= 364.5 810 1458

Se comprobó que si pertenece a la familia de .

7.

Paso 1: (

Paso 2(Comprobar Omega):

Dada esta condición, podemos determinar que C1 puede valer , ya que este valor funciona para cualquier valor de n.

Paso 3 (Comprobar Big O): (

(

Dada esta condición, podemos determinar que C2 puede valer 3 y satisfacer exitosamente cualquier valor de n.

Comprobación: n=2

( =

Demostramos que si pertenece a la familia de .

8.

Paso 1: (

Paso 2 (Comprobar Omega):

Dada la condición, podemos determinar que C1 puede valer para cumplir cualquier valor de n

Paso 3(Comprobar Big O): (

Al ver esta desigualdad podemos determinar que C2 puede valer 1 ya que se va a cumplir para cualquier valor de n.

Comprobación: n=5

( =

9.

Paso 1: (

Paso 2 (Comprobar Omega):

Al observar la condición, podemos determinar que C1 puede valer para cumplir la condición para cualquier valor de n.

Paso 3 (Comprobar Big O): (

Al observar la condición, nos podemos dar cuenta de que C2 puede valer 2 para cumplir la condición, pero esta se va a cumplir a partir de valor de = 2.

Comprobación: n=2

(

Comprobamos que si pertenece a la familia de .

10.

Paso 1: (

Paso 2 (Comprobar Omega):

Al observar esta condición, podemos observar que nunca va a ser verdadera independientemente del valor que le demos a C1, ya que al estar elevado a la 4ta potencia este terminara rebasando al exponente al cuadrado en algún punto, por lo tanto, no se cumple y podemos determinar que no pertenece a la familia de .