Taller 2

Fundamentos de análisis y diseño de algoritmos

Carlos Andres Delgado S, Ing [*]

Abril 2018

1. Reglas del taller

1. El taller debe ser entregado el **Sábado, 14 de Abril de 2018 08:00am** hora de Colombia, por el campus virtual. El enlace se cierra a esa hora y no le permitirá enviar después de eso.
2. El informe debe ir en formato PDF. Los formatos editables (docx, odt, etc) tienen problemas a la hora de ver el archivo en diferentes versiones. Si no entra en este formato se sanciona 0.5 en la nota del taller
3. Debe incluir en el informe los nombres completos y códigos de los integrantes del grupo. Si no incluye esto, se sanciona 0.5 nota del taller.
4. El código debe ir comentado explicando brevemente cada función y apartes importantes del código. Se debe indicar para que se utiliza en el algoritmo. Si no incluye esto, se sanciona 0.5 nota del taller.
5. No se permite división de grupos ni cambios de integrantes de última hora. No se recibirá el taller si esto va producir conflictos entre los integrantes.
6. Debe entregar el código fuente organizado en carpetas dentro del primer nivel del archivo comprimido, no cree una jerarquía compleja difícil de revisar. Si no realiza este paso se sanciona 0.2 en la nota. Lo importante es que eso sea claro para la revisión.
7. No se permite copiar código de Internet ni de sus compañeros. Si se encuentra código copiado de alguna parte el taller será anulado por completo.
8. Entregue un sólo archivo comprimido. No entregue archivos comprimidos dentro de archivos comprimidos, ya que esto dificulta la revisión enormemente. Si hace esta mala práctica se aplica una sanción con 0.5 en la nota del taller.
9. El informe debe tener buena presentación, esto hace parte de la nota de los puntos del taller
10. No deje que el enlace del campus se cierre. No se reciben trabajos por correo, no insista. Si el campus falla, esto será verificado con la DINTEV. Recuerde estar autenticado ya que el curso permite acceso a invitados
11. Las primeras líneas de cada archivo de código fuente, deben tener los integrantes del grupo con sus nombres y código completos. Si no cumple esto, será sancionado con 0.5 en la nota del taller.

[*] carlos.andres.delgado@correounivalle.edu.co

1. **(35 %) Análisis algoritmo Divide y vencerás.**

   El siguiente algoritmo de ordenamiento es conocido como `StoogeSort`.

   ```
   STOOGE-SORT(A,i,j)
   1 if A[i] > A[j]
   2     then exchange A[i] ↔ A[j]
   3 if i + 1 ≥ j
   4     then return
   5 k ← ⌊(j − i + 1)/3⌋          ▷ Round down.
   6 STOOGE-SORT(A,i,j − k)       ▷ First two-thirds.
   7 STOOGE-SORT(A,i + k,j)       ▷ Last two-thirds.
   8 STOOGE-SORT(A,i,j − k)       ▷ First two-thirds again.
   ```

   a) (5 %) Explique la estrategia: Dividir, conquistar y combinar

   b) (10 %) Implemente el algoritmo

   c) (5 %) Realice pruebas con arreglos aleatorios de tamaño 10, 100, 1000, 10000.

   d) (15 %) Calcule la complejidad teórica del algoritmo y compárela con la complejidad práctica encontrada. Analice lo que encuentra.

2. **(35 %) Diseño algoritmo Divide y vencerás.**
   Diseñe un algoritmo bajo la estrategia divide y vencerás para encontrar la moda de un vector. La moda de un vector es el elemento que más se repite, si existe más de una moda, el algoritmo retornará todos los elementos que son moda. Ejemplo

   - (1,2,2,3,4) La moda es 2
   - (1,2,2,3,3,5) La moda es 2 y 3

   a) (5 %) Explique la estrategia: Dividir, conquistar y combinar

   b) (10 %) Implemente el algoritmo

   c) (5 %) Realice pruebas con arreglos aleatorios de tamaño 10, 100, 1000, 10000.

   d) (15 %) Calcule la complejidad teórica del algoritmo y compárela con la complejidad práctica encontrada. Analice lo que encuentra.

3. **(30 %)** Comparación ejecución algoritmos

   Implemente los algoritmos Insertion-Sort y Merge-Sort y realice una comparación entre los utilizando una tabla para entradas aleatorias de tamaño 10, 50, 100, 500, 1000, 2000, 5000 y 10000.

   El análisis debe estar contenido en una tabla donde, para cada algoritmo, se tomen muestras de diferentes tamaños (unas 2 o 3 muestras por cada n, por lo que debe tomar el tiempo promedio), se de el tiempo real de ordenamiento, la complejidad temporal del algoritmo, y se saque el factor constante[1].
   Un ejemplo de la tabla (para un algoritmo de complejidad $\mathcal{O}(n^2)$) es el siguiente:

Entrada (n)	Tiempo Real (seg.)	Complejidad ($\mathcal{O}(n^2)$)	Constantes
5	0.11	25	0.0044
5	0.11	25	0.0044
5	0.12	25	0.0048
10	0.38	100	0.0038
10	0.40	100	0.0040
10	0.41	100	0.0041
⋮			
		Constante:	0.0042

[1] Recuerde que $Tiempo\ Real = Constante * Complejidad$