ANÁLISIS Y DISEÑO DE ALGORITMOS
Ing. Román Martínez M.
Semestre Agosto-Diciembre 2016

TAREA #4

Esta tarea deberás resolverla programando en el lenguaje y ambiente de programación C++ los ejercicios que se indican. A través de estos ejercicios, seguiremos reforzando los conceptos de análisis de algoritmos, pero aplicando ahora la estrategia de divide y vencerás. Como resultado, deberás entregar una impresión de tus códigos en la clase del **viernes 9 de septiembre**. A esta sesión, acude con tu computadora y programas, pues aleatoriamente se pedirá a algunos alumnos exponer sus resultados.
Adicional al reporte impreso que entregarás en la sesión, deberás grabar y subir un video de no más de 3 minutos en el que muestres tus programas en ejecución y des tu testimonio de tu experiencia al realizar esta tarea y los aprendizajes logrados. La entrega del video será en el espacio de Facebook del curso, de manera pública al resto del grupo y podrá subirse durante el día de la entrega.
Para la sesión del martes 6 de septiembre, se espera que lleves un avance de al menos el primer problema, para compartir experiencias con el grupo.

PROBLEMA 1
Implementa los siguientes casos que mencionamos en clase como ejemplos iniciales para la técnica de divide y vencerás:

a) Obtener a^b con un algoritmo de orden logarítmico. El programa pedirá al usuario los valores de a y b para dar resultado.
b) Para una secuencia de a's seguidas de b's, obtener cuántas a's y b's tiene la secuencia por medio de un algoritmo de orden logarítmico. Los datos para probar este algoritmo se declararán como constante en el programa y tú mismo deberás generar algunas pruebas representativas con unos cuantos datos. En lo que programes, añade la funcionalidad de un contador de comparaciones, de tal manera que al finalizar el proceso también se de este resultado en pantalla.

PROBLEMA 2
En este ejercicio, deberás programar los algoritmos del Quick Sort y del Merge Sort como los describimos en clase. Trata de evitar copiar códigos que ya existen en Internet y desarrollar tú mismo tu programa basándote en lo que tenemos en el material del curso. Esto te permitirá razonar a fondo los algoritmos y comprenderlos, que es la intención principal en esta tarea. Los algoritmos asumirán que los datos que ordenarán serán números enteros cualesquiera, y que estos requieren ser ordenados descendentemente, es decir, de mayor a menor.
Una vez que los tengas programados, deberás preparar algunas pruebas con miles de datos enteros con las siguientes condiciones:

A. Los datos a ordenar están ya ordenados descendentemente.
B. Los datos a ordenar están ordenados ascendentemente.
C. Los datos a ordenar están en un orden aleatorio (utiliza la función estándar *rand* para generarlos).

Cada una de las 3 pruebas deberá ejecutarse con ambos algoritmos de ordenamiento, y deberás medir la eficiencia con contadores de comparaciones y de intercambios. Al finalizar tus pruebas, deberás poder llenar una tabla como la que sigue en la que captures los resultados:

Cantidad de datos en las pruebas = #	**MERGE SORT**	**QUICK SORT**
Prueba A: datos ordenados	*Indicadores de resultado de la prueba*	*Indicadores de resultado de la prueba*
Prueba B: datos con orden inverso	*Indicadores de resultado de la prueba*	*Indicadores de resultado de la prueba*
Prueba C: datos aleatorios	*Indicadores de resultado de la prueba*	*Indicadores de resultado de la prueba*

Identifica en esta tabla cuál es el mejor caso y el peor caso de las pruebas. Relaciona estos resultados con el orden de complejidad y el análisis de cada algoritmo. En tu video has referencia a estas reflexiones.

Esta tabla deberás también entregarla como resultado, junto con la impresión de los códigos implementados, y añadiendo la conclusión a la que llegas con respecto a estos dos algoritmos y su uso.

Para cualquier duda respecto a las condiciones de esta implementación, consulta al profesor.