ESTRUCTURAS DE DATOS Y ALGORITMOS II **2021-1**

PRACTICA # 3: ALGORITMOS DE ORDENAMIENTO PARTE 3

OBJETIVO: El estudiante identificará la estructura de los algoritmos de ordenamiento Merge-sort, Counting-sort y Radix-sort

Objetivo de clase: EL alumno implementará casos particulares de estos algoritmos para entender mejor su funcionamiento a nivel algorítmico.

ACTIVIDADES PARA EL DESARROLLO DE LA PRÁCTICA

Ejercicio 0 Menú de Usuario
Crea un menú para los algoritmos de ordenamiento de la práctica del día

Ejercicio 1. COUNTING SORT
En el lenguaje de tu preferencia (C o Java), realiza la implementación de Counting sort aplicado a un caso específico. Toma en cuenta los siguientes requerimientos:

1) Utiliza un arreglo de 15 elementos (caracteres), los cuales serán solicitados al usuario (asume que el usuario los va a ingresar correctamente) Para ello considera un rango de [A-G] (mayúsculas)
2) Crea un segundo arreglo donde cada posición será asociada a uno de los posibles valores del rango indicado
3) En una primera pasada al arreglo (lista) que llenó el usuario, realiza la cuenta de las apariciones de cada uno de los valores. (Muestra en pantalla el arreglo de la cuenta al finalizar la primera pasada)
Realiza la cuenta del arreglo; en cada índice se considera la cantidad de elementos actuales y los anteriores. (Muestra en pantalla la suma del arreglo)
5) Crea un tercer arreglo para ingresar los elementos ordenados.
6) Realiza una segunda pasada al primer arreglo, partiendo desde el final y para cada elemento, analiza la posición que le corresponde en el segundo arreglo y establece su posición final en el tercero.
Para verificar el funcionamiento y los pasos del algoritmo, agrega impresiones en pantalla del arreglo de la cuenta, y del arreglo final conforme se van añadiendo los elementos

Ejercicio 2. Radix sort
En el lenguaje de tu preferencia (C o Java). Realiza la implementación de radix para el ejemplo visto en clase. Considera los requerimientos siguientes

1) Los datos de entrada serán números de longitud 4 , de entre los cuales solo aparecerán dígitos del 0 al 3
2) Deberás implementar una cola para cada uno de estos dígitos (0,1,2,3)
3) Se deberán solicitar al usuario los valores a ordenar (mínimo 10), los cuales podrás almacenar en un arreglo o una lista
4) Es necesario realizar varias pasadas sobre esta lista o arreglo, y en cada una de ellas utilizar las colas para almacenar los elementos de acuerdo con su posición (unidades, decenas, etc). El programa deberá mostrar la lista resultante en cada iteración
5) Al finalizar, el programa deberá mostrar la lista ordenada

CONTINÚA EN LA PARTE POSTERIOR…

Ejercicio 3. Merge Sort

Con ayuda de NetBeans, abre el proyecto proporcionado, el cual es una implementación de Merge Sort en Java

- Agrega las instrucciones en la clase Principal, en el método ***main(),*** para poder crear un arreglo de números de tipo double, posterioremente crear una instancia de la clase MergeSort, y a través de esa instancia realizar el ordenamiento por MergeSort
- Realiza un análisis del programa indicando cómo es que se realiza la división de la lista y posteriormente la combinación ya ordenada de las sub listas
- Modifica la clase Merge sort, de tal manera que se puedan imprimir en pantalla las iteraciones del algoritmo (las divisiones de la lista en 2)
- Prueba el algoritmo para arreglos aleatorios de diferentes tamaños 50,100,500,1000 y realiza tus análisis

4.- Escribe las conclusiones de tu práctica en la cual incluyas comentarios generales de todos los algoritmos de ordenamiento vistos en las primeras 3 prácticas del curso