Análisis de Algoritmos I

Profesora: Luz Gasca Soto *Ayudantes: Bernal Cedillo Enrique Antonio y García Flores Jorge Luis*

Práctica 4: Ordenamientos **Octubre, 2017**

1 Introducción

Esta práctica consiste en implementar los siguientes algoritmos de ordenamiento:

1. Selection Sort
2. Insertion Sort
3. Merge Sort
4. Quick Sort

2 Descripción

2.1 Entrada

El programa a implementar recibe como entrada en los argumentos de la linea de comandos (ejecutando desde la carpeta 'src'):

1. Nombre de la **imágen** a procesar (debe encontrarse en la carpeta **resource**).
2. **Velocidad**, se trata de un número entero que indica el numero de iteraciones que ocurriran antes de actualizar la interfaz grafica (Entre más grande sea el numero, menos actualizaciones de la interfaz. Y por lo tanto, mayor velocidad).
3. **Algoritmo** a utilizar para el ordenamiento. Las unicas opciones son: 'bubble', 'selection', 'insertion', 'merge', 'quick'

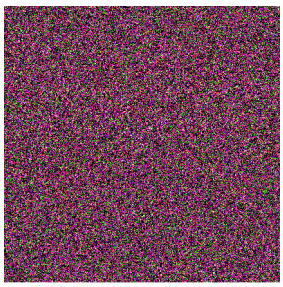
Por ejemplo, para ordenar la **imagen1** a una velocidad de **40**, utilizando **bubble** sort:

java sort.Main imagen1 40 bubble

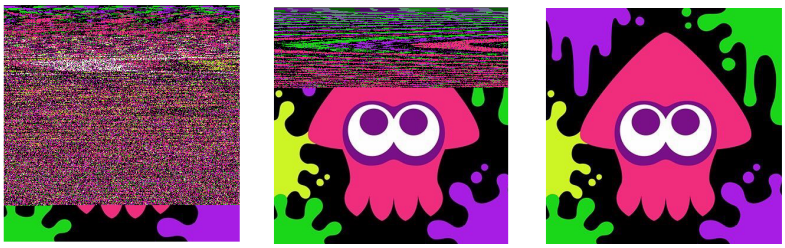
2.2 Salida

Una vez que se ha leido la imágen de entrada.

Los pixeles de dicha imágen deben ser intercambiados de manera aleatoria para generar una imágen con los mismos pixeles, pero en distinto orden.



Ya obtenida la imagen 'revuelta', debe entonces aplicarse el algoritmo de ordenamiento indicado en la linea de comandos para reconstruir la imagen y regresarla a su estado original.
Las iteraciones de los algoritmos de ordenamiento deben ser visibles en la interfaz grafica en tiempo real.
Esto con el fin de poder observar el comportamiento de los algoritmos de manera visual.



Debido al buen desempeño del grupo en la práctica anterior, se proporciona un código base con todo lo que se alcanzó a ver durante la ayudantía.
De manera que sólo queda pendiente implementar los algoritmos de ordenamiento en la clase **Sort.java**.
En dicha clase se proporciona (en la linea 171) un ejemplo con la implementación de Bubble Sort y el uso del método **update** en conjunto con la variable **framerate** para el manejo de las actualizaciones de la interfaz grafica.

3 Detalles adicionales

La práctica puede ser implementada en **Java** o **Python**.
El código base está escrito en Java.
(Pueden modificarlo o alterarlo a su gusto, o utilizar cualquier otra biblioteca para la interfaz grafica, siempre y cuando se indiquen en el readme las instrucciones de uso).

4 Fecha de Entrega

La fecha de entrega de la práctica es el día Jueves 19 de Octubre de 2017.
Al correo enrique_bernal@ciencias.unam.mx
Con asunto **'[Practica04]ApellidopaternoPrimernombre'**

4.1 Adjuntos

-) Readme.txt, con nombre completo e instrucciones para compilar y ejecutar el programa.
Indicando la **velocidad** ideal con cada uno de los algoritmos para alguna imagen (puede ser distinta a las imagenes proporcionadas).
-) Archivos necesarios para la ejecución de su programa en una carpeta llamada 'src'.

Todo lo anterior en un archivo .zip/.rar/.tar.gz con el mismo nombre del asunto del correo. (Pueden eliminar las imagenes de Resource para ahorrar espacio al enviar el correo)