UNIVERSIDAD DE SANTIAGO DE CHILE
FACULTAD DE CIENCIAS
DEPARTAMENTO DE MATEMÁTICA Y CIENCIA DE LA COMPUTACIÓN
LICENCIATURA EN CIENCIA DE LA COMPUTACIÓN
Profesor(es): Nicolas Thériault
Segundo Semestre de 2018

Programación Avanzada – Laboratorio 2

1. Objetivos

Objetivo principal: aplicar algoritmos de ordenamiento de datos para hacer un análisis estadístico de unas series de datos.

Objetivos específicos:

1. encontrar los valores mínimos y máximos;
2. encontrar la mediana y los cuartiles;
3. determinar los valores más repetidos y la cantidad de valores distintos
4. comparar la eficiencia de las diferentes estrategias.

El laboratorio se trabajará en grupos de 2 o 3 alumnos, entregando un resultado (informe y programas) por grupo.

2. Programación

La programación debe ser en C (no en C++), utilizando a lo más las librerías estándares siguientes:

- stdio.h
- stdlib.h
- math.h
- time.h

Además, para evitar cualquier problemas de compatibilidad, se recomienda utilizar solamente la función *clock()* para la medición de tiempo.

Archivos a analizar (simulados):

- Examenes médicos
- Notas de alumnos
- Puntos flotantes (sin más restricciones)

el formato de los datos será: archivos de tipo ".tex" con 10 entradas por linea, en formato decimales (ej: "3.6") o exponenciales (ej: "1.257e-4")

Crear funciones para:

1. Leer los archivos y liberar la memoria activa antes de cerrar el programa;
2. Devolver los resultados;
3. Dos funciones que ordenan el listado (arreglo o enlazado) directamente:
   - Ordenar por mezcla;
   - Quicksort;
4. Dos funciones que ordenan utilizando árboles, entre:
   - Ordenar por montículos;
   - Árboles AVL;
   - Árboles 2-3;
5. Una función que utiliza una estrategia "intercambio tiempo-memoria" (para utilizar en a lo menos un de los archivo), elegida entre:
   - Ordenar por conteo;
   - Listado de repeticiones de valores;

3. Se solicita

1. Programar los diferentes algoritmos en C, asegurando que hacen un uso apropiado de memoría (por ejemplo que se liberan corectamente todos los espacios de memoria utilizados al terminar las funciones).
2. Para cada archivo de datos, determinar si hay valores repetidos, y en el caso positivo encontrar el valor más repetido.
3. Si no está indicado en la programación (con el indicador *const*), cuales parámetros de las funciones no pueden ser modificados para las funciones. Además, documentar cuando algunos parámetros pueden ser repetidos en el mismo llamado de la función.
4. El informe debería detallar:
   - La estructura de datos utilizada para los diferentes algoritmos (con las razones generales de porque se eligió).
   - Las estrategias utilizadas en cada caso.
   - Como se hicieron las mediciones de tiempo (en que computador/procesador, etc).
   - Conclusiones obtenida de los análisis de tiempos.

4. Evaluación

La nota del laboratorio se calculará según la ponderación siguiente:

- Algoritmos [**30 %**]:
  El informe detalla las eleciones que se utilizaron durante la implementación.
- Análisis [**10 %**]:
  El análisis estadístico es correcto y completo.
- Informe [**10 %**]:
  El informe está escrito en lenguaje apropiado, sin faltas de ortografía o gramática..
- Implementación [**30 %**]
  El programa está escrito de forma que puede ser leído y/o re-utilizado fácilmente por otros programadores: la redacción es limpia (con espacios y divisiones claras) y bien documentada, las sub-funciones y las variables tienen nombres naturales (que indican a que sirven) o van acompañadas de comentarios aclarando a que sirven.
  Se libera el espacio antes de cerrar las funciones.
  Todos los algoritmos dan resultados correcto.
- Eficiencia [**20 %**]
  Los tiempos de calculo de ordenamiento comparados con una implementación "básica" (que satisface lo solicitado). Rangos de notas:
  - 1: no ordena correctamente.
  - 1.1–2.4: más de 20 veces más lenta
  - 2.5–3.9: 5 a 20 veces más lenta
  - 4–5.4: 2 a 5 veces más lenta
  - 5.5–6.9: a lo más 2 veces más lenta
  - 7: tiempos parecidos o mejores