UNIVERSIDAD DEL VALLE DE GUATEMALA

Guatemala 07 de febrero de 2024

Jonathan Díaz - 23837
Anggie Quezada - 23643

Hoja de trabajo #3

Resultados gráficas

LINK GIT: https://github.com/Jonialen/HDT-3---Estructura-de-datos.git

Bubble Sort:

- Profiler utilizado: JProfiler

Cómo se empleó:

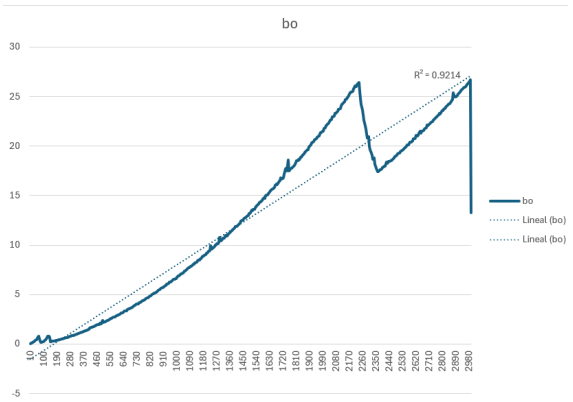
1. Configuramos el proyecto en JProfiler y ejecutamos el programa desde el entorno de JProfiler.

2. Seleccionamos los métodos relevantes para ser perfilados, en este caso, los métodos de Bubble Sort.

3. Analizamos los resultados obtenidos del profiling, que incluyen información detallada sobre el tiempo de ejecución de cada método.

Resultados obtenidos: 6399.053 milisegundos

Gráfica obtenida:



Gnome Sort:

- Profiler utilizado: JProfiler

Cómo se empleó:

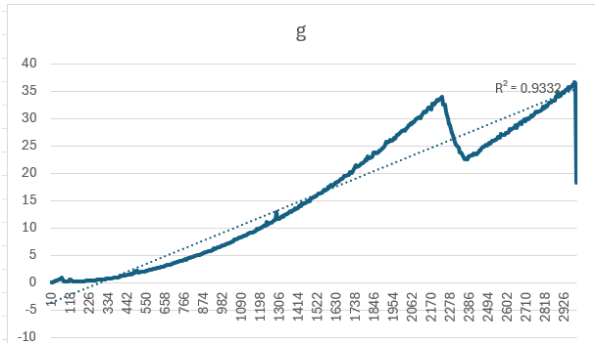
1. Configuramos el proyecto en JProfiler y ejecutamos el programa desde el entorno de JProfiler.

2. Seleccionamos los métodos relevantes para ser perfilados, en este caso, los métodos de Gnome Sort.

3. Analizamos los resultados obtenidos del profiling, que incluyen información detallada sobre el tiempo de ejecución de cada método.

Resultados obtenidos: 7958.515 milisegundo

Gráfica obtenida:



Merge Sort:

- Profiler utilizado: JProfiler

Cómo se empleó:

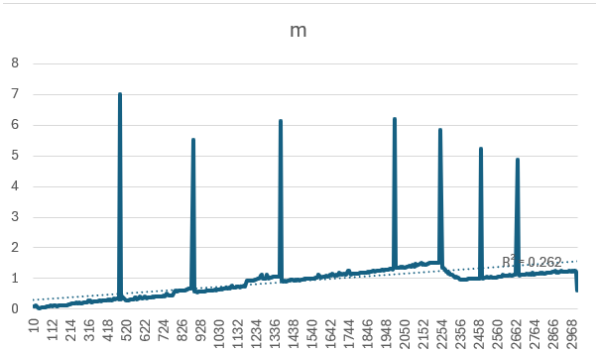
1. Configuramos el proyecto en JProfiler y ejecutamos el programa desde el entorno de JProfiler.

2. Seleccionamos los métodos relevantes para ser perfilados, en este caso, los métodos de Merge Sort.

3. Analizamos los resultados obtenidos del profiling, que incluyen información detallada sobre el tiempo de ejecución de cada método.

Resultados obtenidos: 460.181 milisegudnos

Gráfica obtenida:



Quick Sort:

- Profiler utilizado: JProfiler

Cómo se empleó:

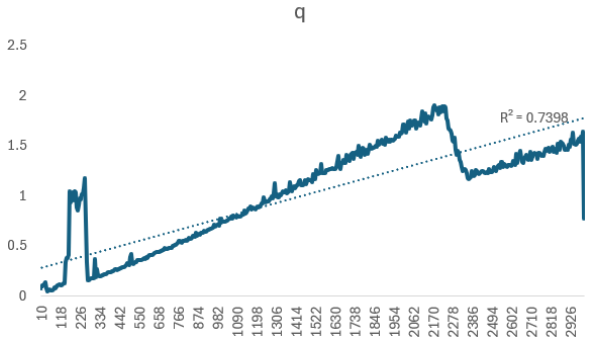
1. Configuramos el proyecto en JProfiler y ejecutamos el programa desde el entorno de JProfiler.

2. Seleccionamos los métodos relevantes para ser perfilados, en este caso, los métodos de Quick Sort.

3. Analizamos los resultados obtenidos del profiling, que incluyen información detallada sobre el tiempo de ejecución de cada método.

Resultados obtenidos: 516.799 milisegundos

Gráfica obtenida:



Radix Sort:

- Profiler utilizado: JProfiler

Cómo se empleó:

1. Configuramos el proyecto en JProfiler y ejecutamos el programa desde el entorno de JProfiler.

2. Seleccionamos los métodos relevantes para ser perfilados, en este caso, los métodos de Radix Sort.

3. Analizamos los resultados obtenidos del profiling, que incluyen información detallada sobre el tiempo de ejecución de cada método.

Resultados obtenidos: 249.339 milisegundos

Gráfica obtenida: