Informe
Trabajo Práctico Final
Programación Concurrente - 2C2023

Integrantes:

Apellido, Nombre	*Legajo*	*Email*
Baron, Elias	58362	eliasnbaron@gmail.com
Ferreyra, Valentin	55437	valentineferreyra@gmail.com
Kippes, Diego	38365	kippes.diego@gmail.com

Profesores:

Terlisky, Pablo
Mastropasqua, Nicolas

Fecha de entrega:

07/11/2023

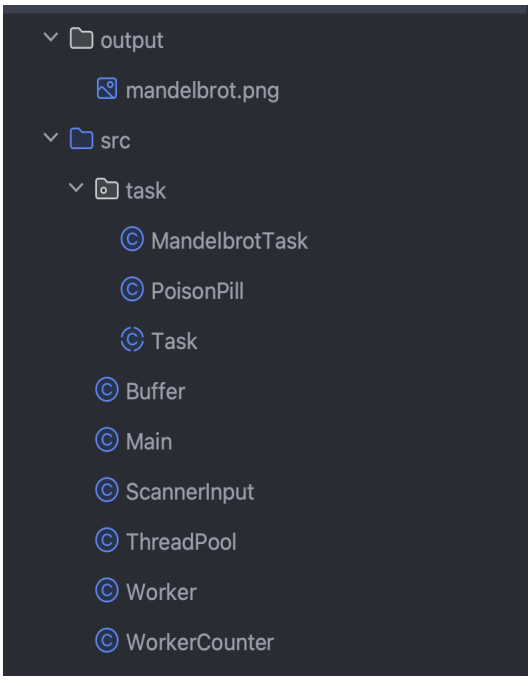
Introducción

Se pidió implementar en Java un programa que grafique el conjunto de Mandelbrot, y este informe surge como resultado de las diversas pruebas realizadas sobre el programa desarrollado.
El objetivo del programa fue dibujar *N* filas, siendo *N* la altura de la imagen a testear, pixel por pixel, el color representado por el número complejo *c* donde la posición del eje x representa la parte real de *c* y la posición del eje y la parte imaginaria.

Diseño del código:
Se divide el código en dos packages: *src* y *output*. src contiene el programa y su lógica, instanciando **Main** las diferentes instancias, controlando el tiempo de ejecución y generando la imagen. Se generará un **Buffer** con un tamaño fijo de 16, donde se irá llenando de la siguiente manera:

- primero se agregan *N* **MandelbrotTask**, siendo *N* la altura de la imagen a crear.
- Por último, se añaden *M* **PoisonPill**, siendo *M* la cantidad total de threads a generar.

Se tendrá un monitor **WorkerCounter**, encargado de controlar la cantidad de threads trabajando, evitando que Main termine de ejecutarse hasta que todos hayan finalizado sus tareas.
Una clase **Worker** que tomará tareas del Buffer y las ejecutará.



La imagen generada por la aplicación se mostrará como archivo *“mandelbrot.png”* en la carpeta *output* y se podrá ver en la terminal el tiempo que duró en ejecutarse el programa.

Evaluación

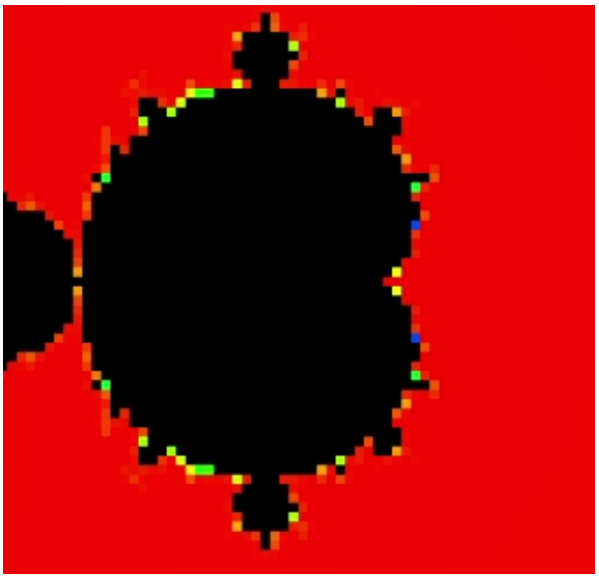
Las pruebas se realizaron en un equipo con las siguientes características:

- Microprocesador: Ryzen 5 3.7 de 6 núcleos.
- Memoria: 32GB de RAM.
- Sistema operativo: Windows 11.

Valores tomados para realizar las pruebas:

- El tamaño del Buffer, como se mencionó anteriormente, será de 16.
- x_inicial = -1; y_inicial = 1; x_rango = 2; y_rango = -2.

Imagen 1: 64x64



Para 1 thread:

Cant. iteraciones	Tiempo
10	26 ms
100	31 ms
500	31 ms
1000	36 ms

Para 2 threads:

Cant. iteraciones	Tiempo
10	24 ms
100	27 ms
500	34 ms
1000	35 ms

Para 4 threads:

Cant. iteraciones	Tiempo
10	22 ms
100	27 ms
500	33 ms
1000	34 ms

Para 8 threads:

Cant. iteraciones	Tiempo
10	22 ms
100	27 ms
500	35 ms
1000	40 ms

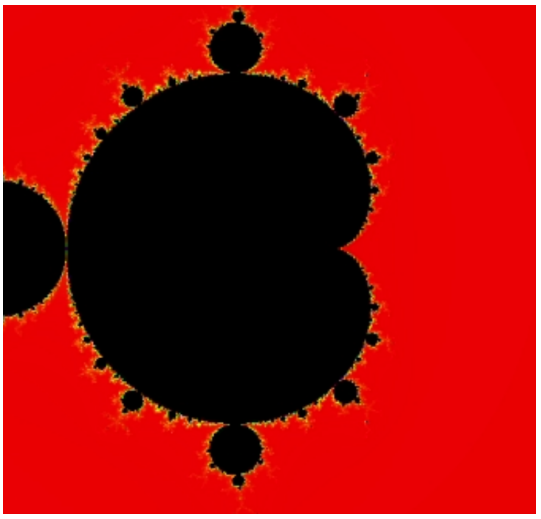
Para 16 threads:

Cant. iteraciones	Tiempo
10	23 ms
100	25 ms
500	38 ms
1000	53 ms

Imagen 2: 512x512

Para 1 thread:

Cant. iteraciones	Tiempo
10	74 ms
100	96 ms
500	177 ms
1000	288 ms



Para 2 threads:

Cant. iteraciones	Tiempo
10	77 ms
100	82 ms
500	127 ms
1000	185 ms

Para 4 threads:

Cant. iteraciones	Tiempo
10	87 ms
100	91 ms
500	106 ms
1000	134 ms

Para 8 threads:

Cant. iteraciones	Tiempo
10	89 ms
100	94 ms
500	102 ms
1000	113 ms

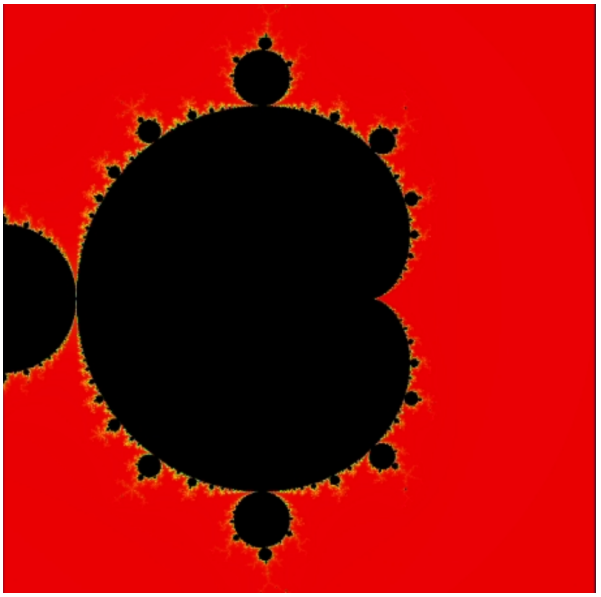
Para 16 threads:

Cant. iteraciones	Tiempo
10	95 ms
100	114 ms
500	142 ms
1000	137 ms

Imagen 3: 1024x1024

Para 1 thread:

Cant. iteraciones	Tiempo
10	140 ms
100	228 ms
500	554 ms
1000	953 ms



Para 2 threads:

Cant. iteraciones	Tiempo
10	138 ms
100	205 ms
500	340 ms
1000	543 ms

Para 4 threads:

Cant. iteraciones	Tiempo
10	125 ms
100	151 ms
500	238 ms
1000	344 ms

Para 8 threads:

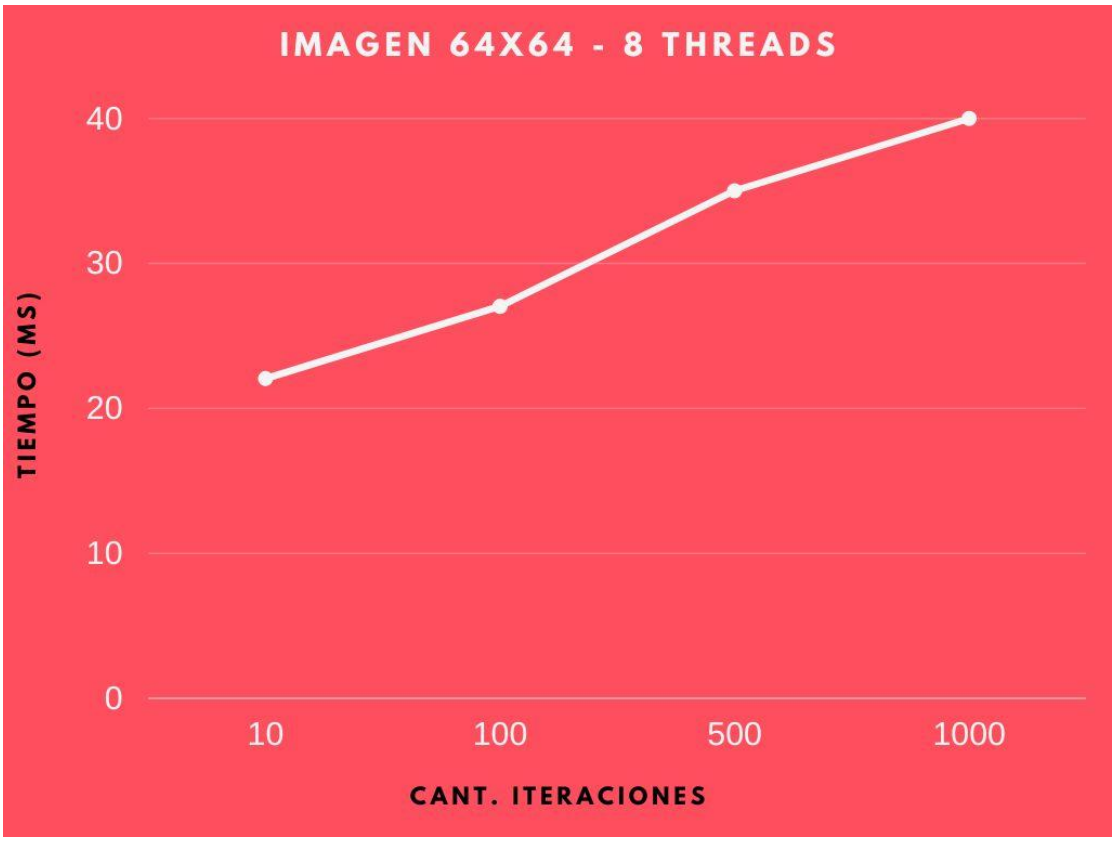
Cant. iteraciones	Tiempo
10	123 ms
100	139 ms
500	201 ms
1000	242 ms

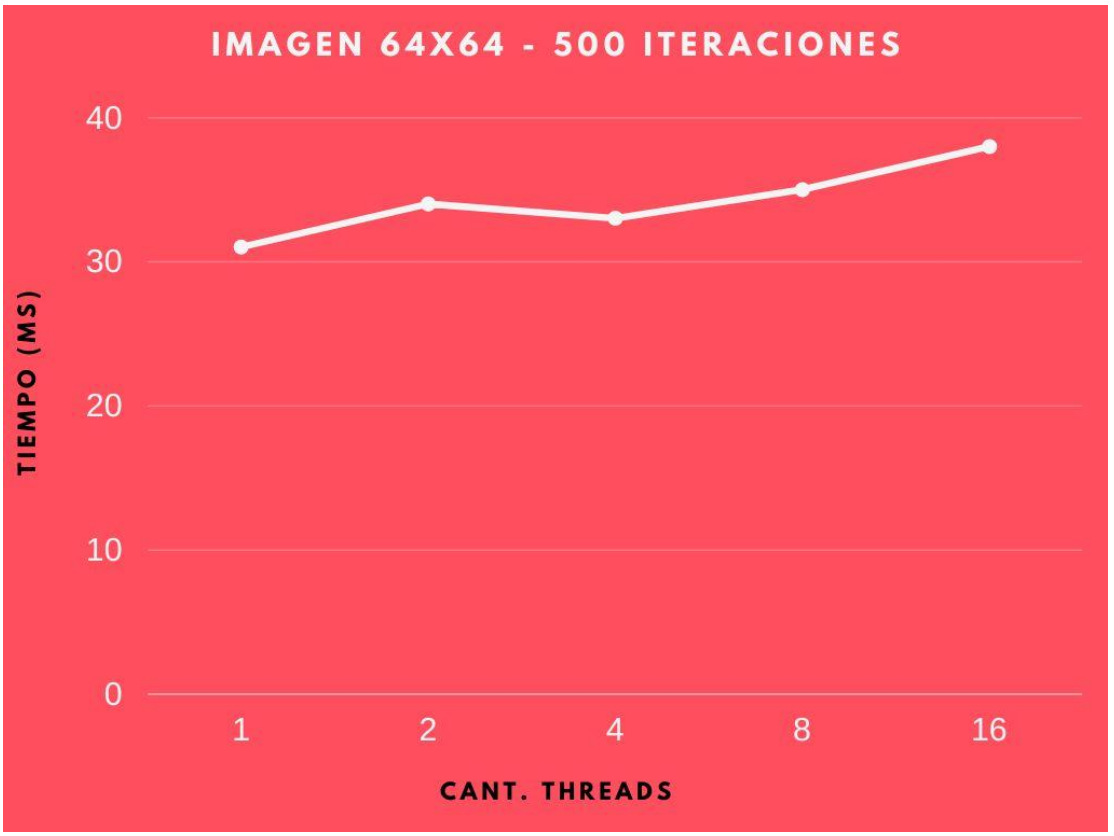
Para 16 threads:

Cant. iteraciones	Tiempo
10	239 ms
100	225 ms
500	242 ms
1000	267 ms

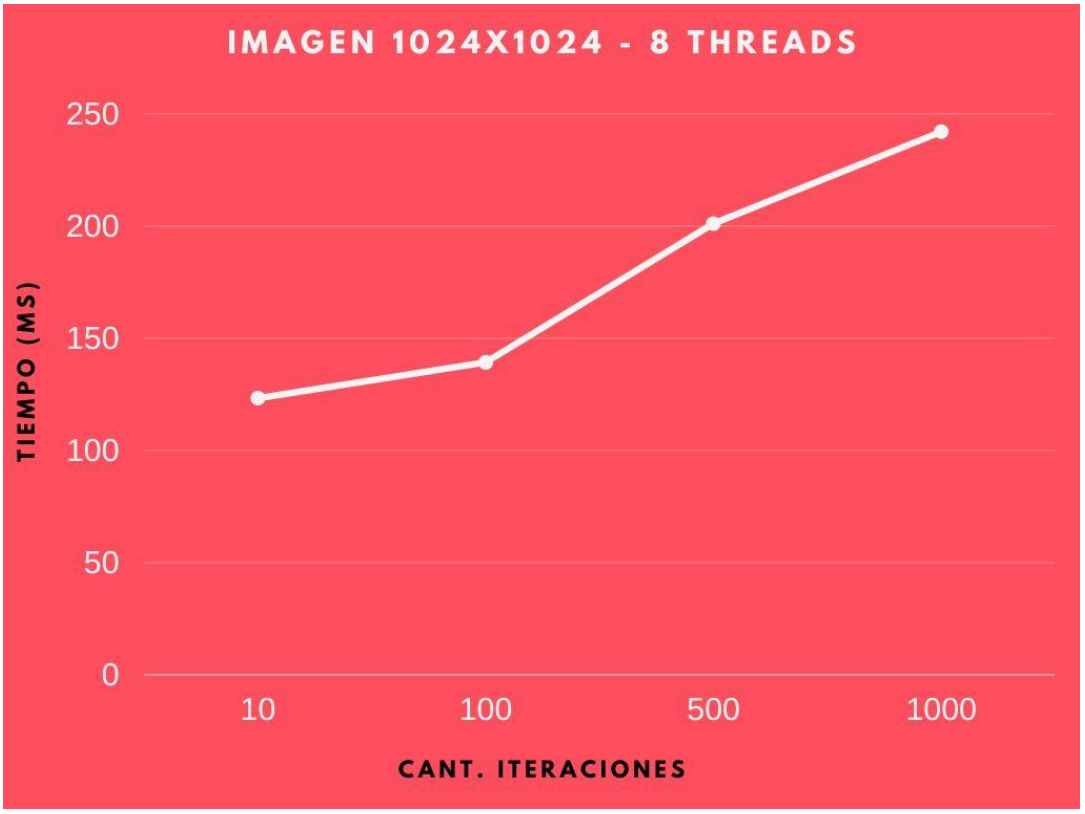
Para realizar los gráficos y poder llegar a conclusiones, tomamos las imágenes con menor y mayor resolución (64x64 y 1024x1024).

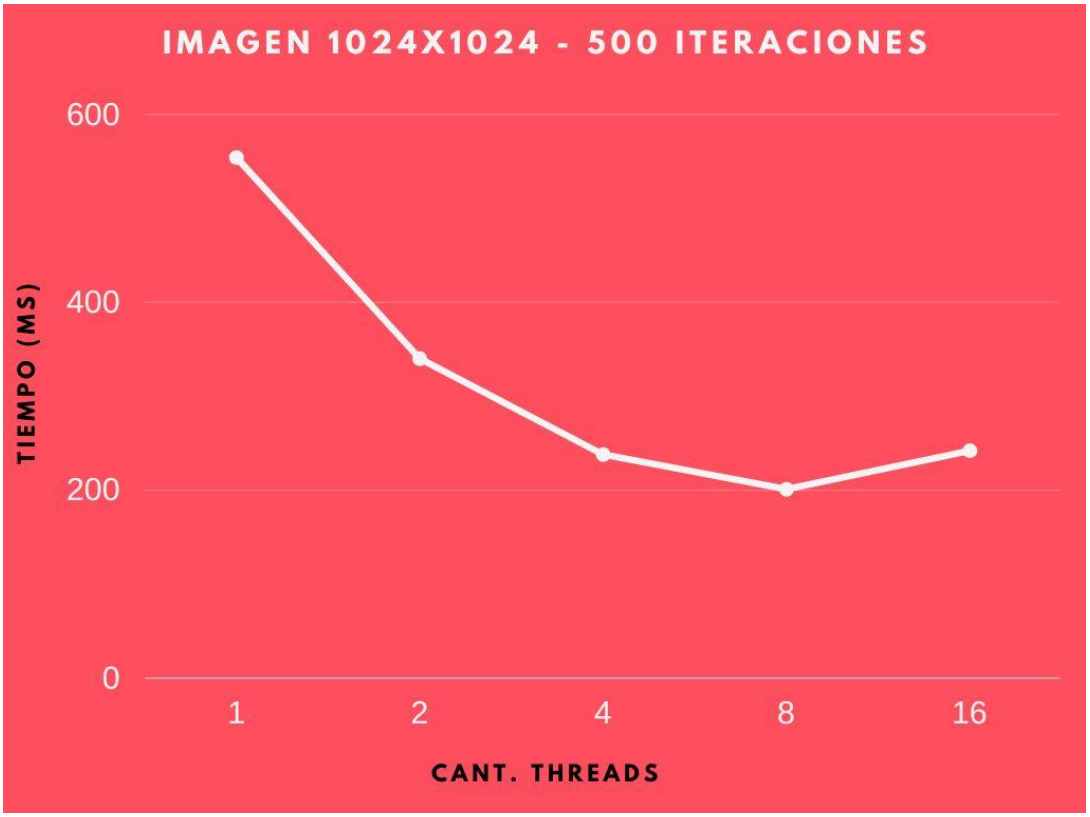
Gráficos obtenidos para imagen de tamaño 64x64:





Gráficos obtenidos para imagen de tamaño 1024x1024:





Análisis

A partir de los datos obtenidos, tanto en cuadros de tiempo de ejecución como los gráficos generados, el equipo llegó a las siguiente conclusiones:

- El tiempo transcurrido cambia dependiendo del tamaño de la imagen. Se puede concluir con los gráficos que: en 500 iteraciones, una imagen 1024x1024 tarda mucho más en generarse cuando solo hay un thread trabajando. Mientras que en una imagen 64x64, tarda ligeramente más cuando hay 16 threads trabajando.
- Se observa que el tiempo de ejecución disminuye al aumentar la cantidad de threads para tamaños de imagen más grandes. No obstante, existen ciertos puntos de inflexión en los que la adición de más hilos no mejora significativamente el rendimiento o incluso puede aumentar el tiempo de ejecución, debido a que si son más threads que tasks en el Buffer, estos no realizan ninguna acción más que terminar.
- La cantidad de iteraciones influye significativamente en el tiempo de ejecución del programa, especialmente en imágenes de mayor tamaño (se puede observar el incremento considerable en el gráfico de imagen 1024x1024). A medida que se aumenta el número de iteraciones, se incrementa la carga de trabajo del procesamiento del *conjunto de Mandelbrot*, lo que se refleja en un aumento en el tiempo de cálculo.