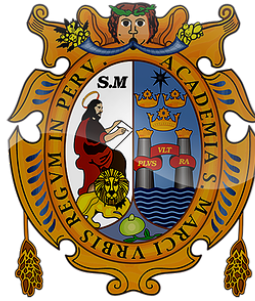


” Año del Fortalecimiento de la Soberanía Nacional”



Universidad Nacional Mayor de San Marcos
(Universidad del Perú, DECANA DE AMERICA)

FACULTAD DE INGENIERIA DE SISTEMAS E INFORMATICA Escuela
Profesional de Ingenieria de Software



Segundo Laboratorio: Laboratorio Python

Curso: Programación Paralela y Concurrente

Profesor: Edson Ticona Zegarra

Integrantes:

- Pichilingue Pimentel, Nathaly Nicole 19200247
- Torre Arteaga, Alexander 19200246
- Ricse Perez, Anthony Elias 19200276

Ciudad Universitaria 2022

1. Explicación del algoritmo implementado

- **Cuestiones generales** Las distintas versiones del algoritmo realizado fueron implementadas en Python, haciendo uso del Python nativo, y librerías como Cython y OpenMP. Para realizar la evaluación de la entrada, se creó una función que genera una lista de n círculos con sus respectivos datos como posición xy , radio y color rgb . Luego, en base a esos datos, se traza el círculo, se detecta los píxeles afectados y se realiza la modificación en el archivo PPM, lo que permite dibujar los círculos deseados. Consideraciones a tomar en nuestro código:
 - Nuestro algoritmo solo toma como valores de entrada válidos, números enteros, negativos o positivos.
- **Función principal** La función principal implementada en el proyecto llama a la función de entrada, donde ingresaremos los datos necesarios para dibujar el(los) círculo(s). Una vez se retorne la lista de círculos como respuesta, se llama a la función de creación de canvas en archivo PPM y luego a la función de dibujo, que nos retornará una matriz de píxeles. Con dicha matriz, se procede a llamar a la función de escritura de archivo, que modificará el canvas PPM creado inicialmente según la posición y el canal de colores rgb de los píxeles obtenidos, finalizando así la ejecución del programa.
- **Función de entrada** La función de entrada llamada `.entrada` inicia solicitando al usuario que ingrese el número de círculos a dibujar, una vez obtenido el número de círculos, se realiza una loop para solicitar los siguientes datos por cada círculo: coordenada (x,y) que representa el eje del círculo, radio del círculo, y valores para cada canal de color rgb del círculo, en el orden rojo, verde y azul. Luego, guarda dichos datos en una lista llamada círculos y retorna dicha lista.
- **Función de creación de canvas** La función de creación del canvas, llamada `create_image`, genera una imagen del tamaño 1024 por 960 preestablecido multiplicando con el valor de los canales de color negro, representado en rgb como 000 (respectivamente para cada canal de color), y finalmente guarda el resultado en una lista, que es el retorno de la función.

- **Función de dibujo** La función de dibujo de círculos, llamada `draw_circle`, se ejecuta por cada n círculo a dibujar de la siguiente manera: Primero, realiza la lectura de los datos de cada círculo, asignando los valores en cada lugar de un array. Luego empieza a realizar iteraciones entre la coordenada x e y, en un rango de -radio a + radio, y procede a realizar dos verificaciones, la primera si el punto actual está fuera de los límites de la imagen, y la segunda si el punto actual está dentro del círculo, con la ecuación de la circunferencia

$$\sqrt{x} + \sqrt{y} = \sqrt{r} \quad (1)$$

. Una vez realizada dichas validaciones, guarda los datos del píxel según canal de color y una función XOR, y los inserta en una posición. Una vez generado la lista de píxeles con su posición y respectivo color rgb, la función retorna dicha lista de píxeles a modificar en la imagen.

- **Función de escritura de archivo** La función de escritura de archivo, llamada `writePPM`, genera la imagen de salida final, iniciando con el header del archivo, luego genera una lista llamada `rgb` donde almacenará el color de cada píxel según posición. una vez realizado esto se guarda la imagen en un `array.array` y escribe el archivo PPM, retornando como resultado el archivo de salida.

2. Entorno de pruebas:

- Procesador: AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx 2.10 GHz
- Memoria RAM: 16 GB
- Número de procesadores: 8
- Sistema Operativo:
- IDE de desarrollo y pruebas: Visual Studio Code

3. Datos de entrada utilizados para las pruebas: Poner texto aquí

4. **Resultado de pruebas:** Poner texto aquí

5. **Análisis de resultados:** Poner texto aquí

6. **Anexo:**

Ray Tracer: <https://github.com/aysusayin/Ray-Tracer>