**OBSERVACIONES DEL LA PRACTICA**

María José Amorocho, 202220179

Jacobo Zarruk, 202223913

# **Preguntas de análisis**

1. ¿Qué instrucción se usa para cambiar el límite de recursión de Python?

La instrucción que se usa en el código para cambiar el límite de recursión de Python es sys.setrecursionlimit(). En este caso, el valor del límite se establece como 2 \*\*20, o 1048576 recursiones.

1. ¿Por qué considera que se debe hacer este cambio?

Este cambio debe hacerse dado a que el problema exige manejar una recursión profunda. Si el límite no se extiende, la recursión podría bloquear el programa o generar un desbordamiento de pila.

1. ¿Cuál es el valor inicial que tiene Python cómo límite de recursión?

El valor inicial que tiene Python como límite de recursión es de 1000.

1. ¿Qué relación existe entre el tiempo que toma la operación 4 con el número de vértices y arcos del grafo?

Toma de datos de la función 4:

|  |  |  |  |
| --- | --- | --- | --- |
| Tamaño entrada | Tiempo de ejecución (ms) | Vértices | Arcos |
| 50 | 36.62 | 74 | 73 |
| 150 | 49.04 | 146 | 146 |
| 300 | 49.47 | 295 | 382 |
| 1000 | 190.03 | 984 | 1633 |
| 2000 | 597.03 | 1954 | 3560 |
| 3000 | 1145.50 | 2922 | 5773 |
| 7000 | 2304.11 | 6829 | 15334 |
| 10000 | 10470.45 | 9767 | 22758 |
| 14000 | 16310.15 | 13535 | 32270 |

Entre mayor cantidad de nodos, mas arcos pueden formarse, de manera que existirían más rutas posibles. La función 4 al calcular la ruta más corta desde la estación indicada a todas las otras estaciones (los otros vértices del grafo), tardará más en ejecutarse si tiene mayor número de rutas que analizar.

Toma de datos de la función 6:

|  |  |  |  |
| --- | --- | --- | --- |
| Tamaño entrada | Tiempo de ejecución (ms) | Vértices | Arcos |
| 50 | 1.35 | 74 | 73 |
| 150 | 1.75 | 146 | 146 |
| 300 | 1.97 | 295 | 382 |
| 1000 | 2.00 | 984 | 1633 |
| 2000 | 2.01 | 1954 | 3560 |
| 3000 | 2.02 | 2922 | 5773 |
| 7000 | 2.29 | 6829 | 15334 |
| 10000 | 2.45 | 9767 | 22758 |
| 14000 | 3.10 | 13535 | 32270 |

1. ¿El grafo definido es denso o disperso?

Teniendo en cuenta que el grafo es dirigido, aplicando la fórmula para obtener su densidad, se obtiene que es un grafo disperso (para el archivo con el mayor tamaño de entrada).

1. ¿Qué adaptación debería hacerse a la fórmula en caso de que el grafo fuera NO dirigido?

Debería multiplicarse por dos la cantidad de arcos posibles, de manera que la fórmula para obtener su densidad sea 2m/n(n-1)

1. ¿El grafo es dirigido o no dirigido? Para cada tamaño de los datos incluya el resultado de la densidad en el documento de observaciones

Se trata de un grafo dirigido, ya que las rutas tienen una dirección específica entre las

Estaciones.

1. ¿El grafo es conexo?

Teniendo en cuenta que el grafo tiene una menor cantidad de arcos con relación a la cantidad de vértices, y el número de componentes conectados, se puede decir que es un grafo no conexo.

1. ¿Cuál es el tamaño inicial del grafo?

El tamaño inicial del grafo consiste en 13535 vértices y 32270 arcos (para el archivo con el mayor tamaño de entrada).

1. ¿Cuál es la Estructura de datos utilizada?

La estructura de datos usada consiste en un diccionario nativo de Python con los índices ‘stops’, ‘connections’, ‘components‘ y ‘paths’. ‘stops’ tiene como valor una tabla de hash, mientras que ‘connections’ es una matriz de adyacencia.

1. ¿Cuál es la función de comparación utilizada?

La función usada en la matriz de adyacencia es nombrada ‘compareStopIds’.