**OBSERVACIONES DEL LA PRACTICA**

Juan José Osorio 202021720

Thais Tamaio

# **Preguntas de análisis**

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

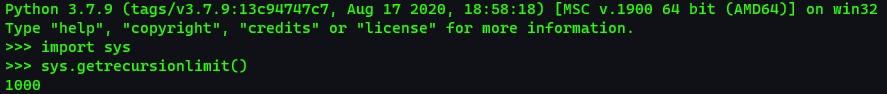
La instrucción que se utiliza para cambiar el límite de recursión de Pyhton es sys.setrecursionlimit(), en el laboratorio, se establece un valor de .

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

Teniendo en cuenta que por defecto python cuenta con un límite de recursión de 1000, una función podría llamarse a si misma más de 1000 veces, por lo que si el límite de recursión se sobrepasa se genera el siguiente error: *RuntimeError: maximum recursion depth exceeded*. Como el grafo que se construye en el laboratorio cuenta con 13535 vértices, el ejecutar las funciones recursivas, que permiten obtener información útil acerca de este grafo, pueden ser funciones que se tengan que llamarse a sí mismas más de 1000 veces. Por consiguiente, establecer un límite de recursión bastante alto evita que se genere un error al ejecutar operaciones recursvias sobre el grafo.

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

Como se mencionó en la pregunta b), el límite de recursión por defecto de python es de 1000, esto se puede confirmar por medio de la siguiente imagen:



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

|  |  |  |  |
| --- | --- | --- | --- |
| **Tamaño del archivo** | **Vértices** | **Arcos** | **Tiempo de ejecución [ms]** |
| 50 | 74 | 73 | 28.17907600000035 |
| 150 | 146 | 146 | 52.49211800000194 |
| 300 | 295 | 382 | 86.62632099999973 |
| 1000 | 984 | 1633 | 365.0449129999979 |
| 2000 | 1954 | 3560 | 1168.4759449999983 |
| 3000 | 2922 | 5773 | 1653.1278870000024 |
| 7000 | 6829 | 15334 | 6936.531637000007 |
| 10000 | 9767 | 22758 | 15192.810172999998 |
| 14000 | 13535 | 32270 | 27875.733144000005 |

Con base en la tabla, se puede observar que a medida que aumenta el número de vértices y de arcos, aumenta el tiempo de ejecución de la opción 4. Esto se debe, a que cuando hay más vértices y arcos, mayor es el número de datos que se tiene que procesar para completar la opción 4.

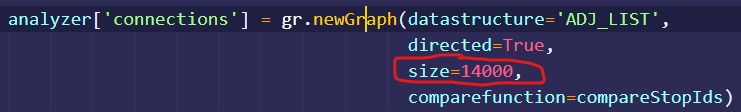
|  |  |  |  |
| --- | --- | --- | --- |
| **Tamaño del archivo** | **Vértices** | **Arcos** | **Tiempo de ejecución [ms]** |
| 50 | 74 | 73 | 1.4470619999992778 |
| 150 | 146 | 146 | 1.7816429999984393 |
| 300 | 295 | 382 | 1.5925460000034946 |
| 1000 | 984 | 1633 | 2.001575000002049 |
| 2000 | 1954 | 3560 | 2.9438560000053258 |
| 3000 | 2922 | 5773 | 3.52588299999843 |
| 7000 | 6829 | 15334 | 4.3458810000011 |
| 10000 | 9767 | 22758 | 4.977478000000701 |
| 14000 | 13535 | 32270 | 4.300545000005513 |

1. ¿Qué características tiene el grafo definido?

Las características del grafo definido son: Es un dígrafo (es decir, que no es bidireccional) y tiene una función de comparación que compara los ID de las estaciones. Su tamaño inicial es 14.000 vértices y el tipo del grafo es “ADJ\_LIST”, por lo que se busca almacenar únicamente la información relevante del grafo, es decir, solo se guardan los vértices y arcos presentes, no como en el caso de una matriz de adyacencias donde se consume una cantidad mayor de memoria.

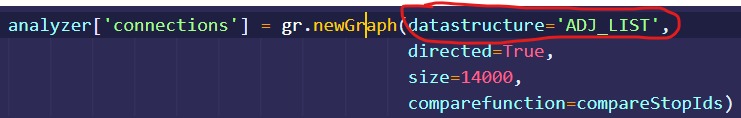
1. ¿Cuál es el tamaño inicial del grafo?

Como se mencionó en el punto e), el grafo tiene por defecto un size de 14.000. Esto se puede ver en la siguiente imagen:



1. ¿Cuál es la Estructura de datos utilizada?

Como se mencionó en el punto e), la estructura de datos que se utilizó para la creación del grafo es “ADJ\_LIST”. En esta estructura de datos, se utilizan mapas para almacenar toda la información relacionada con los vértices y los arcos de un grafo. Esto se puede ver en la siguiente imagen:



1. ¿Cuál es la función de comparación utilizada?

La función de comparación que se utiliza para la creación del grafo es: “compareStopIds”. Esta función, compara los códigos de dos estaciones diferentes. La función se puede ver en la siguiebte imagen:

