

Taller 7

Jose David Ramirez Beltran - 506222723

24 de septiembre de 2023

1. Introducción

Clase Nodo

La clase `Nodo` se utiliza para definir un nodo en una lista enlazada. Cada nodo consta de dos atributos: `valor`, que almacena el valor del nodo, y `siguiente`, un puntero al siguiente nodo en la lista.

Creación de Nodos y Ciclos

En este código, se crean cinco nodos denominados `nodo1` a `nodo5`. Luego, se establece un ciclo en la lista enlazada haciendo que el último nodo (`nodo5`) apunte al segundo nodo (`nodo2`), creando así una lista enlazada circular.

Función `detectar_ciclo`

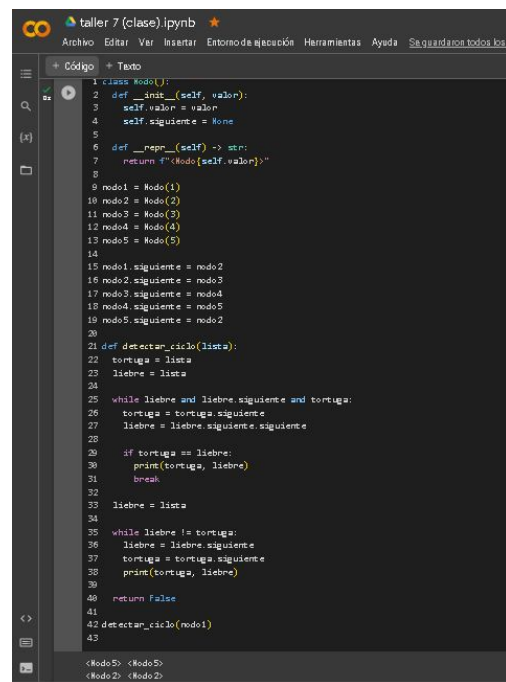
La función `detectar_ciclo` toma un nodo (generalmente el primer nodo de la lista enlazada) como entrada y utiliza el algoritmo de la tortuga y la liebre para detectar si hay un ciclo en la lista enlazada. La tortuga avanza un nodo a la vez, mientras que la liebre avanza dos nodos a la vez, osea que si ambos punteros se encuentran en el mismo nodo en algún momento, esto indica la existencia de un ciclo.

Detección e Impresión del Ciclo

Cuando se detecta un ciclo, la función imprime los nodos en los que se encuentran la tortuga y la liebre cuando se encuentran en el mismo punto. Luego, utiliza dos punteros adicionales para encontrar el nodo de inicio del ciclo.

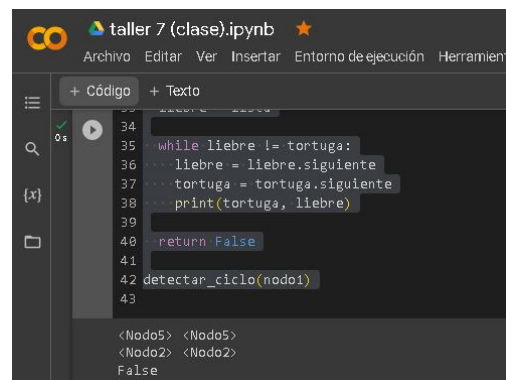
2. Desarrollo

Para llevar el desarrollo del algoritmo, se puede plantear de la siguiente manera:



```
1 class Nodo():
2     def __init__(self, valor):
3         self.valor = valor
4         self.siguiente = None
5
6     def __repr__(self) -> str:
7         return f"<Nodo {self.valor}>"
8
9 nodo1 = Nodo(1)
10 nodo2 = Nodo(2)
11 nodo3 = Nodo(3)
12 nodo4 = Nodo(4)
13 nodo5 = Nodo(5)
14
15 nodo1.siguiente = nodo2
16 nodo2.siguiente = nodo3
17 nodo3.siguiente = nodo4
18 nodo4.siguiente = nodo5
19 nodo5.siguiente = nodo2
20
21 def detectar_ciclo(lista):
22     tortuga = lista
23     liebre = lista
24
25     while liebre and liebre.siguiente and tortuga:
26         tortuga = tortuga.siguiente
27         liebre = liebre.siguiente.siguiente
28
29     if tortuga == liebre:
30         print(tortuga, liebre)
31         break
32
33     liebre = lista
34
35     while liebre != tortuga:
36         liebre = liebre.siguiente
37         tortuga = tortuga.siguiente
38         print(tortuga, liebre)
39
40     return False
41
42 detectar_ciclo(nodo1)
43
44 <Nodo5> <Nodo5>
45 <Nodo2> <Nodo2>
```

3. Funcionamiento



```
34
35 while liebre != tortuga:
36     liebre = liebre.siguiente
37     tortuga = tortuga.siguiente
38     print(tortuga, liebre)
39
40 return False
41
42 detectar_ciclo(nodo1)
43
44 <Nodo5> <Nodo5>
45 <Nodo2> <Nodo2>
False
```

Este código se usa para encontrar ciclos en listas enlazadas, lo hace mediante el mantenimiento de dos punteros; uno que avanza un nodo a la vez

(conocido como "tortuga") y otro que avanza dos nodos a la vez (conocido como "liebre"). La liebre eventualmente alcanzará a la tortuga si hay un ciclo en la lista, lo que indica la existencia de un ciclo, luego, uno de los punteros se reinicia para encontrar el nodo de inicio del ciclo y avanzan a la misma velocidad hasta que se encuentran nuevamente en el nodo de inicio del ciclo, por ende se puede decir que este algoritmo es útil para identificar ciclos en listas enlazadas porque funciona bien en tiempo lineal.

4. Conclusiones

En resumen, el código demuestra su capacidad para identificar ciclos en estructuras de datos lineales al crear una lista enlazada circular de cinco nodos y aplicar el algoritmo. En la gestión de listas enlazadas y otras estructuras similares, la detección de ciclos es un proceso crucial, y este algoritmo proporciona una solución efectiva para ese propósito. Además, las recomendaciones adicionales que se incluyen en la descripción ayudan a explorar y comprender mejor cómo funciona el algoritmo en diferentes situaciones y cómo se puede aplicar a listas enlazadas y otros tipos de estructuras de datos.

5. Bibliografía

<https://github.com>

techiedelight.com/es/detect-cycle-linked-list-floyds-cycle-detection-algorithm/

<https://colab.research.google.com/drive/1fdLJkA4wexEbJrEpOKQxcVc47YXxIZrp>