
PROYECTO 2 – Aplicación móvil

202100019 – Angela María Esther Escobar Alvarez

Resumen

Como punto principal hablaremos sobre las clases, nodos y listas enlazadas dobles que utilicé en mi proyecto y quise abarcar la mayoría de programación orientada a objetos.

Las clases de Python proveen todas las características normales de la Programación Orientada a Objetos como es el mecanismo de la herencia de clases permite múltiples clases base, una clase derivada puede sobre escribir cualquier método de su(s) clase(s) base, y un método puede llamar al método de la clase base con el mismo nombre. Una clase es una plantilla para el objetivo de la creación de objetos de datos según un modelo predefinido. Las clases se utilizan para representar entidades o conceptos, como los sustantivos en el lenguaje. Al igual que los bloques de código, los nodos de Python son una interfaz de secuencias de comandos dentro de un entorno de programación visual. Las siguientes listas enlazadas se desglosan de la siguiente manera:

- * Es un tipo de lista enlazada que permite moverse hacia delante y hacia atrás.
- * Las Listas pueden crear, actualizar y eliminar elementos.
- * Las Listas tienen dos protocolos, uno secuencial y el otro directo.

Palabras clave

- **Nodo:** En informática y en telecomunicación, de forma muy general, un nodo es un punto de intersección, conexión o unión de varios elementos que confluyen en el mismo lugar.
- **Clase:** Es la descripción de un conjunto de objetos similares; consta de métodos y de datos que resumen las características comunes de dicho conjunto.
- **Listas enlazadas:** Una lista enlazada es una colección lineal de elementos llamados nodos. El orden entre ellos se establece mediante punteros; direcciones o referencias a otros nodos. tipo de los datos que se quiera almacenar en la lista.
- **Celdas:** En esta ocasión se define como las celdas que están infectadas y las que no.
- **POO:** La programación orientada a objetos se basa en el concepto de crear un modelo del problema de destino en sus programas. La programación orientada a objetos disminuye los errores y promociona la reutilización del código.

Abstract

As a main point we will talk about the classes, nodes and double linked lists that I used in my project and I wanted to cover the majority of object-oriented programming. Python classes provide all the normal features of Object Oriented Programming such as the

class inheritance mechanism allowing multiple base classes, a derived class can override any method of its base class(es), and a method can call the base class method with the same name. A class is a template for the purpose of creating data objects according to a predefined pattern. Classes are used to represent entities or concepts, like nouns in language. Like code blocks, Python nodes are a scripting interface within a visual programming environment. The following linked lists break down as follows:

- * It is a type of linked list that allows moving forwards and backwards.
- * Lists can create, update and delete items.
- * The Lists have two protocols, one sequential and the other direct.

Keywords

- *Node*: In computing and telecommunications, in a very general way, a node is a point of intersection, connection or union of several elements that come together in the same place.
- *Class*: It is the description of a set of similar objects; It consists of methods and data that summarize the common characteristics of said set.
- *Linked list*: Is a linear collection of elements called nodes. The order between them is established by pointers, addresses or references to other nodes. type of the data you want to store in the list.
- *Cells*: This time it is defined as the cells that are infected and those that are not.
- *POO*: Object-oriented programming is based on the concept of creating a model of the target problem in your programs. Object-oriented programming reduces bugs and promotes code reuse.

Introducción

La programación orientada a objetos es una evolución de la programación procedural basada en funciones. La POO nos permite agrupar secciones de código con funcionalidades comunes.

Una de las principales desventajas de la programación procedural basada en funciones es su construcción, cuando una aplicación bajo este tipo de programación crece, la modificación del código se hace muy trabajosa y difícil debido a que el cambio de una sola línea en una función puede acarrear la modificación de muchas otras líneas de código pertenecientes a otras funciones que estén relacionadas.

Con la programación orientada a objetos se pretende agrupar el código encapsulándolo y haciéndolo independiente, de manera que una modificación debida al crecimiento de la aplicación solo afecte a unas pocas líneas.

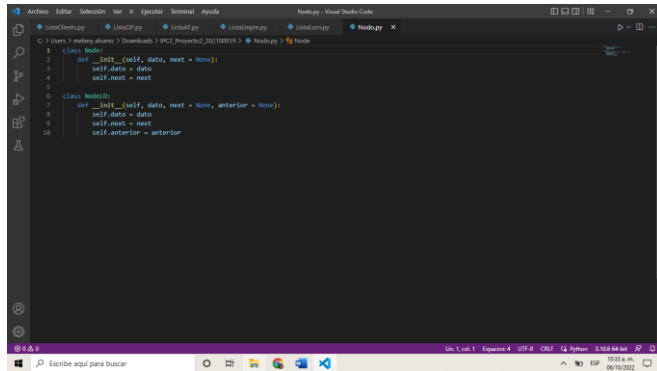
La organización de una aplicación en POO se realiza mediante estructuras de código, también llamados objetos. Estos objetos contienen una serie de procedimientos e información destinados a resolver un grupo de tareas con un denominador común. Un procedimiento que este situado en un objeto no podrá ser usado por otro procedimiento perteneciente a otro objeto, si no es bajo una serie de reglas. Los datos que mantenga el objeto permanecerán aislados del exterior y sólo se podrá acceder a ellos siguiendo ciertas normas.

El objetivo de POO es catalogar y diferenciar el código, en base a estructuras jerárquicas dependientes, al estilo de un árbol genealógico.

Desarrollo del tema

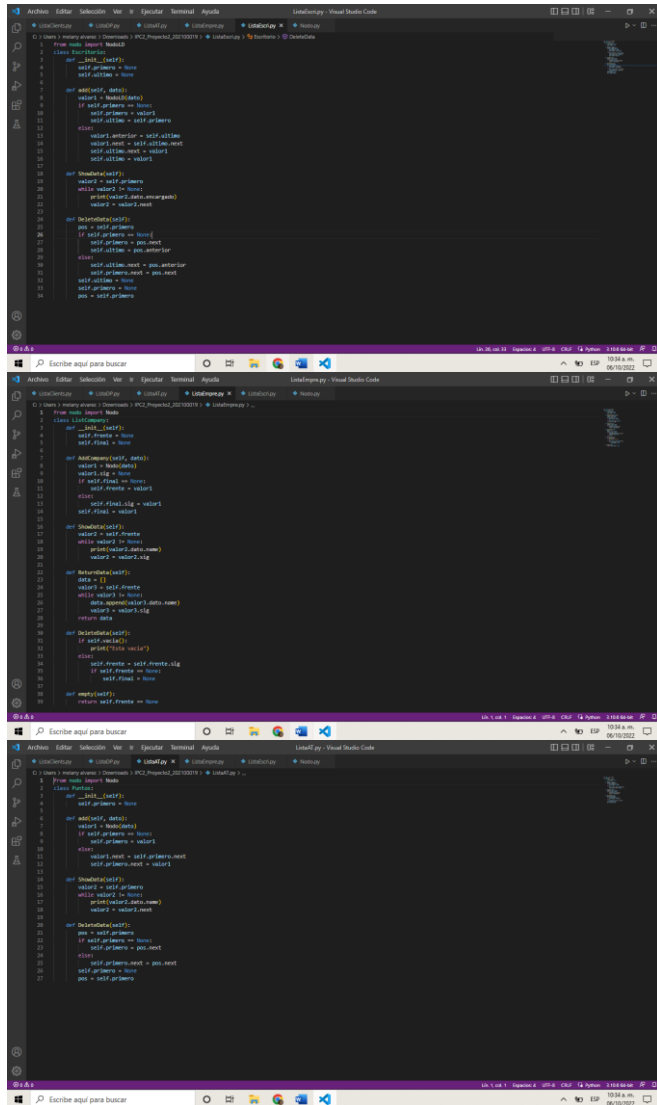
Listas y Nodos

-Nodo

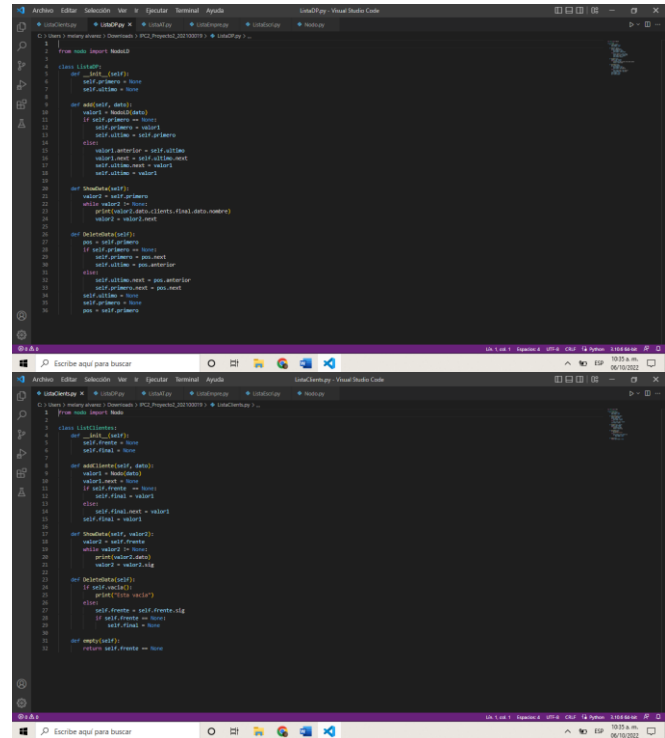


```
1 class Node:
2     def __init__(self, data, next = None):
3         self.data = data
4         self.next = next
5
6     def __str__(self):
7         return str(self.data)
8
9     def add_data(self, data):
10        self.data = data
11
12    def show_data(self):
13        print(self.data)
14
15    def delete_data(self):
16        self.data = None
```

-Listas



```
1 class List:
2     def __init__(self):
3         self.head = None
4         self.tail = None
5         self.size = 0
6
7     def __str__(self):
8         return str(self.head)
9
10    def add_data(self, data):
11        self.head = Node(data)
12        self.tail = self.head
13        self.size += 1
14
15    def show_data(self):
16        print(self.head)
17
18    def delete_data(self):
19        self.head = None
```



```
1 class LinkedList:
2     def __init__(self):
3         self.head = None
4         self.tail = None
5         self.size = 0
6
7     def __str__(self):
8         return str(self.head)
9
10    def add_data(self, data):
11        self.head = Node(data)
12        self.tail = self.head
13        self.size += 1
14
15    def show_data(self):
16        print(self.head)
17
18    def delete_data(self):
19        self.head = None
```

Conclusiones

- La Programación Orientada a Objetos es actualmente el paradigma que más se utiliza para diseñar aplicaciones y programas informáticos.
- Son muchas sus ventajas, principalmente cuando necesitas resolver desafíos de programación complejos.
- El modelo orientado a objetos (OO) es una colección de objetos o clases que permite que un programa pueda examinar y manipular elemento de su entorno.
- Una clase es el anteproyecto que ofrece la funcionalidad en ella definida, pero ésta queda implementada solo al crear una instancia de la clase, en la forma de un objeto.
- La Programación Orientada a Objetos (POO) es un paradigma de programación que busca que nuestra forma de programar sea más cercana a la forma como nos relacionamos en nuestro día a día.

Referencias bibliográficas

- <https://desarrolloweb.com/articulos/1330.php>
- http://sis324loo.blogspot.com/2008/09/conclusiones_6587.html
- <https://www.tusclasesparticulares.cl/blog/2018/3/primeros-pasos-aprender-programacion-orientada-objeto-poo.html>
- <https://villegas1.activoforo.com/t2-que-es-la-programacion-orientada-a-objetos-poo.html>
- <https://ejemplos.net/que-significa-poo-en-programacion/.html>