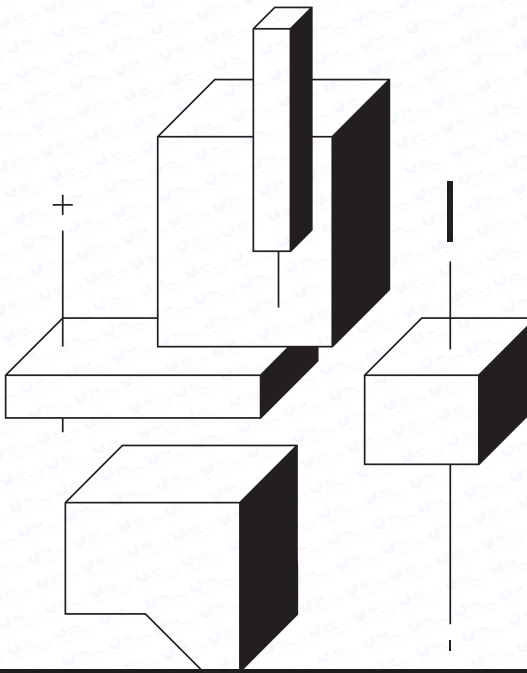
Jala University
Cursos de Admisión

ABSTRACCION Y **PROGRAMACION ORIENTADA A OBJETOS**



Jala University
Cursos de Admisión

Parte I

Paradigmas de programación

A continuación veremos unos conceptos introductorios

Que es un paradigma de programación?

Un paradigma de programación es un conjunto de conceptos y principios que definen la manera en que se estructuran y ejecutan los programas de computadora. Es como un marco o modelo que guía la forma en que los programadores piensan y escriben código. Cada paradigma tiene sus propias reglas, técnicas y estilos que determinan cómo se resuelven los problemas de programación. Los cuales se dividen en dos grandes grupos, paradigmas imperativos y paradigmas declarativos

Caracteristicas del paradigma imperativo

- En el paradigma imperativo, el énfasis está en cómo se realizan las operaciones.
- Se centra en los pasos específicos que debe seguir el programa para alcanzar un resultado.
- Los programas imperativos están compuestos por una secuencia de instrucciones que modifican el estado del programa.
- Se destacan los conceptos de variables, asignaciones, bucles y estructuras de control.

Caracteristicas de paradigma declarativo

- En el paradigma declarativo, el énfasis está en qué debe hacer el programa, no en cómo debe hacerlo.
- Se centra en la descripción de la lógica y las reglas que deben seguirse para alcanzar el resultado deseado.
- Los programas declarativos expresan la solución como una serie de afirmaciones o declaraciones sobre el problema.
- No se especifica el flujo de control detallado del programa, sino más bien las relaciones y restricciones entre los datos.

Jala University
Cursos de Admisión

Parte II

Abstracción

Jala University
Cursos de Admisión

Antes de ver que es programación orientada a objetos y entender su sintaxis en python, veremos un concepto muy importante. **Abstracción**

Que es la abstracción?

En términos simples, la abstracción permite crear modelos simplificados de objetos del mundo real en el software, centrándose en las características esenciales y funciones que son relevantes para el problema que se está resolviendo. Esto significa que los detalles internos de cómo funciona el objeto no son visibles desde el exterior, lo que facilita su uso y comprensión.

Por ejemplo, si estás desarrollando un sistema de gestión de biblioteca, podrías crear una clase "Libro" que tenga propiedades como título, autor y número de páginas. Estos detalles son relevantes para el funcionamiento de la biblioteca, pero los detalles de cómo se almacena la información internamente en la clase pueden estar ocultos para el usuario del sistema.

En resumen, la abstracción en la POO permite simplificar la complejidad al representar objetos y sus interacciones de una manera que sea fácil de entender y utilizar, sin necesidad de comprender todos los detalles internos de su implementación.

Consideración para realizar abstracción

- Antes de empezar a abstraer, se debe considerar el contexto del problema, para solo abstraer las caracteristicas principales y que seran utiles durante la resolucion del problema
- Identificar de forma clara las acciones vs las caracteristicas que componen al objeto en la vida real, las acciones siempre seran representados por verbos
- El model que vayamos a abstraer basado en el objeto de la vida real puede ser fisico como intangible, por ejemplo una cuenta bancaria no es algo fisico, pero es algo que existe en la vida real y podemos abstraer sus caracteristicas para solucionar un problema

Jala University
Cursos de Admisión

Parte II

Programacion orientada a objetos

Que es POO?

La Programación Orientada a Objetos (POO) es un enfoque de programación que se basa en el concepto de "objetos". En la POO, los objetos son entidades que pueden tener datos (llamados atributos) y comportamientos (llamados métodos).

Imagina que estás construyendo un sistema para una biblioteca. En lugar de tratar la biblioteca como una serie de funciones y variables independientes, en la POO la biblioteca se consideraría como un "objeto" que tiene características (como el nombre, la ubicación y el número de libros) y acciones (como prestar un libro, devolver un libro, etc.).

Que es un objeto?

Un objeto en POO es una instancia concreta y única de una clase que tiene sus propios atributos(propiedades) y comportamientos (métodos). Los objetos son la base de la POO y permiten modelar entidades del mundo real de manera más eficiente y organizada en el código de programación.

Que es un atributo?

Un atributo es una característica o propiedad que describe el estado de un objeto. También se le conoce como campo, propiedad o variable de instancia. Los atributos representan los datos asociados con un objeto y pueden ser de diferentes tipos, como números, cadenas de texto, booleanos, objetos u otros tipos de datos.

Que es un metodo?

Un método es una función asociada a una clase o a un objeto específico. Los métodos permiten que los objetos realicen acciones o ejecuten operaciones específicas.

Ya definimos varios conceptos, pero como escribimos todo eso en codigo? Necesitamos definiir lo que es una clase para poder hablar de su sintaxis y funcionamiento

Que es una clase?

Una clase es una plantilla o un modelo para crear objetos. Representa un conjunto de atributos y métodos que define el comportamiento y las características de los objetos que se pueden crear a partir de ella.

Piensa en una clase como un plano o un diseño para crear objetos. Define qué propiedades y qué acciones pueden tener esos objetos, pero no representa los objetos en sí mismos. Los objetos son instancias específicas de una clase.

Una clase puede incluir lo siguiente:

1. **Atributos**: Representan las características o datos que tienen los objetos de la clase. Por ejemplo, una clase "Coche" podría tener atributos como "color", "marca", "modelo", etc.
2. **Métodos**: Son las funciones que definen el comportamiento de los objetos de la clase. Los métodos permiten que los objetos realicen acciones específicas. Por ejemplo, una clase "Coche" podría tener métodos como "acelerar", "frenar", "girar", etc.

La clase actúa como un contenedor que encapsula tanto los datos como el comportamiento de los objetos que se crean a partir de ella. Proporciona una forma de organizar y estructurar el código, facilitando la reutilización y la mantenibilidad del software.

Como es la sintaxis para definir una clase

```python
class Auto:

  matricula = 'abc123'
  marca = 'Toyota'
  color = 'Negro'
  km = 150

  def encender(self):
    print('El auto esta encendido')

  def apagar(self):
    print('El auto esta apagada')

  def acelerar(self):
    print('Acelerando...')
```

Jala University
Cursos de Admisión

Como es la sintaxis para definir una clase

```
class Auto:

  matricula = 'abc123'
  marca = 'Toyota'
  color = 'Negro'
  km = 150

  def encender(self):
    print('El auto esta encendido')

  def apagar(self):
    print('El auto esta apagada')

  def acelerar(self):
    print('Acelerando...')
```

Ahora usaremos la clase para crear el objeto y poder llamar sus metodos

```
auto = Auto()
auto.encender()
auto.acelerar()
auto.apagar()
```

Hablemos del constructor, antes de ver su uso, veremos un concepto sencillo

Que es el constructor?

Los constructores se utilizan para crear instancias sobre los objetos de una clase. La función de los constructores es asignar valores a los atributos de clase cuando se crea un objeto de esa misma clase. A continuacion veremos su sintaxis

```
class Auto:

  def __init__(self):
    matricula = 'abc123'
    marca = 'Toyota'
    color = 'Negro'
    km = 150

  def encender(self):
    print('El auto esta encendido')

  def apagar(self):
    print('El auto esta apagada')

  def acelerar(self):
    print('Acelerando...')
```

Los siguientes son ejemplos de malas definiciones de clases, todos presentan errores de sintaxis o semantica, es decir fallar al compilar o al ejecutarse.

```python
class Auto:

  def __init__(self):
    matricula = 'abc123'
    marca = 'Toyota'
    color = 'Negro'
    km = 150

  def encender():
    print('El auto esta encendido')

  def apagar(self):
    print('El auto esta apagada')

  def acelerar(self):
    print('Acelerando...')

auto = Auto()
auto.encender()
auto.acelerar()
auto.apagar()
```

Esta definicion esta mal, ya que el metodo encender no tiene como parametro self, todos los metodos dentro de una clase deben tener el parametro self

```python
class Auto:

  def __init__(self):
    matricula = 'abc123'
    marca = 'Toyota'
    color = 'Negro'

  def imprimir_matricula(self):
    print(matricula)

  def encender(self):
    print('El auto esta encendido')

  def apagar(self):
    print('El auto esta apagada')

  def acelerar(self):
    print('Acelerando...')

auto = Auto()
auto.imprimir_matricula()
auto.acelerar()
auto.apagar()
```

El metodo **imprimir_matricula** imprime una variable llamada matricula, pero esta no esta definida en el metodo o en la clase

```python
class Auto:

  def __init__(self, color):
    matricula = 'abc123'
    marca = 'Toyota'
    color = color

  def encender(self):
    print('El auto esta encendido')

  def apagar(self):
    print('El auto esta apagada')

  def acelerar(self):
    print('Acelerando...')

auto = Auto()
auto.acelerar()
auto.apagar()
```

la clase Auto en su constructor necesita el color, por eso al momento de llamar al constructor vacío **Auto()** fallara