República Bolivariana de Venezuela

Ministerio del Poder Popular para la Educación Superior

Universidad Rafael Urdaneta

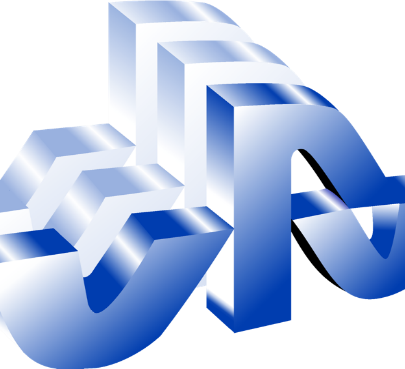
Facultad de Ingeniería

Profesor: Eli Ramon Mora.

Cátedra: programación orientada a objetos.

Sección: A

**Programación orientada a objetos**



Estefany Carolina Puerta Puertas

29.811.120

**Programación estructurada:** es un paradigma de programación basado en utilizar [funciones](https://entrenamiento-python-basico.readthedocs.io/es/latest/leccion5/funciones.html#python-funciones) o subrutinas, y únicamente tres estructuras de control: Secuencia que es la ejecución de una sentencia tras otra, Selección o condicional se refiere a la ejecución de una sentencia o conjunto de sentencias, según el valor de una variable booleana e Iteración (ciclo o bucle), ejecución de una sentencia o conjunto de sentencias, mientras una variable booleana sea verdadera. Este paradigma se fundamente en el teorema correspondiente, que establece que toda función computable puede ser implementada en un lenguaje de programación que combine sólo estas tres estructuras lógicas o de control.

La estructura de secuencia es la que se da naturalmente en el lenguaje, ya que por defecto las sentencias son ejecutadas en el orden en que aparecen escritas en el programa. Para las estructuras condicionales o de selección, Python dispone de la sentencia [if](https://entrenamiento-python-basico.readthedocs.io/es/latest/leccion4/condicional_if.html" \l "python-sent-if), que puede combinarse con sentencias [elif](https://entrenamiento-python-basico.readthedocs.io/es/latest/leccion4/condicional_if.html#python-sent-elif) y/o [else](https://entrenamiento-python-basico.readthedocs.io/es/latest/leccion4/condicional_if.html" \l "python-sent-else), para los bucles o iteraciones existen las estructuras [while](https://entrenamiento-python-basico.readthedocs.io/es/latest/leccion4/bucle_while.html" \l "python-bucle-while) y [for](https://entrenamiento-python-basico.readthedocs.io/es/latest/leccion4/bucle_for.html#python-bucle-for).

**Programación orientada a objetos:** se basa en crear un modelo de programación informática que organiza el diseño de software en torno a datos u objetos, en lugar de funciones y lógica. Un objeto se puede definir como un campo de datos que tiene atributos y comportamiento únicos. La programación orientada a objetos se centra en los objetos que los desarrolladores quieren manipular en lugar de enfocarse en la lógica necesaria para manipularlos. La organización de un programa orientado a objetos también hace que el método sea beneficioso para el desarrollo colaborativo, donde los proyectos se dividen en grupos.

**El método constructor** es un método especial para crear e inicializar un objeto creado a partir de una clase. Sólo puede haber un método especial con el nombre de "constructor" en una clase. Un error de sintaxis será lanzado, si la clase contiene más de una ocurrencia de un método constructor.

Cuando se crea un objeto en Java se realizan las siguientes operaciones de forma automática:

1.Se asigna memoria para el objeto.

2. Se inicializan los atributos de ese objeto con los valores predeterminados por el sistema.

3. Se llama al constructor de la clase que puede ser uno entre varios.

El constructor de una clase tiene las siguientes características:

Tiene el mismo nombre que la clase a la que pertenece.

En una clase puede haber varios constructores con el mismo nombre y distinto número de argumentos (se puede sobrecargar), no se hereda y no puede devolver ningún valor (incluyendo void).

Debe declararse público (salvo casos excepcionales) para que pueda ser invocado desde cualquier parte donde se desee crear un objeto de su clase.

**MÉTODO CONSTRUCTOR POR DEFECTO**

Si para una clase no se define ningún método constructor se crea uno automáticamente por defecto.

El **constructor por defecto** es un constructor sin parámetros que no hace nada. Los atributos del objeto son iniciados con los valores predeterminados por el sistema.

**Los Setters & Getters**: son métodos de acceso lo que indica que son siempre declarados públicos, y nos sirven para dos cosas:

Setters: Del Inglés Set, que significa establecer, pues nos sirve para asignar un valor inicial a un atributo, pero de forma explícita, además el Setter nunca retorna nada (Siempre es void), y solo nos permite dar acceso público a ciertos atributos que deseemos el usuario pueda modificar.

Getters: Del Inglés Get, que significa obtener, pues nos sirve para obtener (recuperar o acceder) el valor ya asignado a un atributo y utilizarlo para cierto método.

Mediante el uso de captadores y definidores, los programadores pueden controlar cómo acceder y actualizar sus variables importantes de una manera adecuada, como cambiar el valor de una variable dentro de un rango específico. Considere el siguiente código para el método de establecimiento.

Clases: Las**clases en Java** son plantillas para la creación de objetos, en lo que se conoce como programación orientada a objetos, considerada uno de los principales paradigmas de desarrollo de software en la actualidad.

Es la base sobre la cual se construye todo el [lenguaje Java](https://www.tokioschool.com/noticias/curiosidades-de-java/) porque la clase define la naturaleza de un objeto. Como tal, la **clase** forma la base para la programación orientada a objetos en Java.

Dentro de una **clase**se definen los datos y el código que actúa sobre esos datos. El código está contenido en métodos. Tanto las clases, como los objetos y los métodos son fundamentales para [Java](https://www.tokioschool.com/formaciones/cursos-programacion/java/). Tener una comprensión básica de estas características te permitirá escribir programas más sofisticados y comprender mejor el **lenguaje de programación Java.**

**Cuando defines una clase**, declaras su forma y naturaleza exactas. Para ello, especificarás las variables de instancia que contiene y los métodos que operan en ellas. Aunque las clases muy simples pueden contener solo métodos o solo variables de instancia, la mayoría de las clases del mundo real contienen ambas.

Aunque no existe una regla sintáctica que se deba cumplir, una **clase** bien diseñada debería definir una y solo una entidad lógica. Una clase generada de forma correcta agrupa la información conectada lógicamente. No olvides que poner información no relacionada en la misma clase desestructurará rápidamente tu código.

Una **clase** es, por tanto, una combinación específica de atributos y métodos y puede considerarse un tipo de dato de cualquier tipo no primitivo. Así, una clase define los atributos que componen ese tipo de objetos y los métodos que pueden emplearse para trabajar con esos objetos. Aunque, por otro lado, una **clase** también puede estar compuesta por métodos estáticos que no necesitan de objetos.

**Instanciar** objetos es el proceso de generar un ejemplar de una clase, es decir, la clase es como una declaración de una forma y el objeto es un caso o elemento concreto que responde a esa forma. Podemos crear infinitos objetos a partir de una clase, cada uno sería una instancia de la clase o un ejemplar de esa clase