**Programación Orientada a Objetos:**

La Programación orientada a objetos o POO es una forma de programar especifica donde se organiza el código en clases, de las cuales se crean los objetos que se relacionan entre si para conseguir los objetivos de los sistemas. La POO es la forma de programar mas cercana a como expresamos y resolvemos las cosas en la vida real.

La POO tiene varios conceptos y entre ellos están: Las clases, los objetos, los métodos, los atributos,el polimorfismo, la herencia, la abstracción, los estados, las identidades entre otros.

Unas de las ventajas de utilizar la POO es que se reutilización de código, a no duplicar código, a proteger la información y a construir sistemas complejos de forma mas sencilla

**Constructores Implícitos y Explícitos :**

Un constructor es un elemento de una clase cuyo identificador coincide con el nombre de la clase correspondiente, este constructor tiene como objetivo obligar y a controlar como se inicializa una instancia.

Un constructor explicito es el que se da cuando uno no define los parámetros. Los objetos o variables de instancia tendrán valores predefinidos y fijos.

El constructor implícito se da cuando uno da los parámetros, ya que necesita definir nuevos objetos con valores definidos por el usuario,Los constructores pueden aceptar parámetros que ayudan a establecer los valores de los atributos.

**Foreach:**

Es un bucle for se utiliza para recorrer un arrays, las matrices y las colecciones.

Su estructura es:

for(int x =nombre\_array){

Instrucciones

}

**Encapsulación:**

La encapsulacion consiste el estado de un objeto, esto ayuda a proteger nuestros datos dentro del sistema, estableciendo niveles de visibilidad o accesos a nuestros datos, se divide en 3 niveles:Publico, Privado, Protegido.

Para modificar estos datos encapsulados se utilizan 2 métodos que son Set (para enviar) y Get (para recibir), estos métodos deben ser públicos.

**Herencia**

La herencia es el mecanismo por el cual una clase hereda atributos y métodos a otra clase. Son estructuras jerárquicas, estas estructuras una clase solo puede tener una clase padre (Superclase) y una clase padre puede tener varias clases hijas (Subclase).

**Sobre-carga de métodos (Overloading)**

la sobre carga de métodos es la creación de varios métodos con el mismo nombre pero con diferentes parámetros.

**Sobre-escritura de métodos (Overriding)**

Es la forma de definir los método que se heredan si uno quiere definir un método con los mismos características que el método de la superclase, esto se hace ya que se quiere agregar o modificarla funcionalidad del método heredado, para buenas practicas se debe utilizar el @override ya que ayuda a el compilador a saber que estas sobre escribiendo sobre ese método.

**Excepciones en java (uso del Try Catch)**

Las excepciones son anomalías que ocurren durante la ejecución de las aplicaciones así afectando el flujo normal de las instrucciones dadas. En java existen muchos tipos de excepciones como cuando un objeto es nulo y no debería ser nulo, cuando no se abre correctamente un fichero, cuando se divide por 0, etc.

cuando se produce una excepción se crea un objeto de determinada clase que guarda toda la información de esa excepción, estas clases tiene como clase padre a throwable por lo tanto hay una jerarquía de excepciones.

Con la estructura try-catch-finally podemos capturar y procesar una posible excepción, evitando que el sistema se detenga sin necesidad. Así dándole seguimiento a la excepción.

**clases Abstractas en Java**

Una clase abstracta es igual a una clase normal solo que una clase abstracta no pueden ser instanciadas, los métodos abstractos no llevan cuerpo, para que la clase o método sean abstractos tienen que llevar la palabra “abstract”, si un método de una clase es abstracto la clase tiene que ser abstracta. Con la abstracción se podrá tener un código organizado, y las propiedades solo van a estar disponible para sus subclases