Resúmen:

Se realizo ejercicio de crear 3 clases con sus respectivos atributos y crear distintas instancias.

Personalmente Modele 3 clases de tecnología , la clase celular, notebook y smartphone

Reflexión: Me parece muy interesante la forma en la que funciona la programación Orientada a objetos y la capacidad de modelar los entornos.

Estudio Extra:

Se profundizo en el Funcionamiento de los métodos constructores .

Un **constructor** es un **método** que se ejecuta de manera automática al instanciar un objeto de una clase. El **constructor** tiene como finalidad la incialización de las variables de la clase y posiblemente ejecutar algunos de los métodos de la clase.

|  |
| --- |
| 4.4.5 Definición de constructores de una clase. |
| Cuando se construye un objeto es necesario inicializar sus variables con valores coherentes, imaginemos un objeto de la clase Persona cuyo atributo color de pelo al nacer sea verde, un estado incorrecto tras construir el objeto persona. La solución en los lenguajes orientados a objetos es emplear los constructores. Un constructor es un método perteneciente a la clase que posee unas características especiales:   * Se llama igual que la clase. * No devuelve nada, ni siquiera void. * Pueden existir varios, pero siguiendo las reglas de la sobrecarga de funciones. * De entre los que existan, tan sólo uno se ejecutará al crear un objeto de la clase.   Dentro del código de un constructor generalmente suele existir inicializaciones de variables y objetos, para conseguir que el objeto sea creado con dichos valores iniciales.  Para definir los constructores se emplea la siguiente sintaxis:    *[modifVisibilidad] nombreConstructor (listaParámetros) [throws listaExcepciones]*  *{*  *}*    Para modifVisibilidad se aplica las mismas normas que para atributos y métodos:   * public: indica que es un método accesible a través de una instancia del objeto. * private: indica que a través de una instancia no es accesible el método. Al heredar el método se convierte en inaccesible. * protected: indica que a través de una instancia no es accesible el método. Al heredar si se puede usar desde la clase derivada. * Sin especificar: indica visibilidad de paquete, se puede acceder a través de una instancia, pero sólo de clases que se encuentren en el mismo paquete.   nombreConstructor debe de coincidir con el nombre de la clase.  listaParámetros es la lista de los parámetros que tomará la función separados por comas y definidos cada uno de ellos como:  *tipo nombreParámetro*    La cláusula opcional throws es empleada para indicar que dentro del método se pueden generar errores en su ejecución, y que debemos estar preparados para tratarlos.  listaExcepciones es el nombre de todos esos posibles errores, su utilización la veremos en el punto dedicado a la gestión de errores mediante try y catch.  El constructor posee un par de llaves, dentro de las cuales estará el código que se ejecutará al ser llamada la función. |