UNIVERSIDAD TECNOLÓGICA DE PANAMÁ

FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES

LICENCIATURA EN CIBERSEGURIDAD

NOMBRE DE LA ASIGNATURA:

Programación I

“Investigación #1”

PREPARADO POR:

Rodríguez, Yisuari 1-760-982

A CONSIDERACIÓN DE

Napoleón Ibarra

2S3111

20/08/2025

Contenido

Construcción de Clase (JAVA)

¿Cuál es su concepto?

Una clase en Java es una plantilla o modelo que define las propiedades (atributos) y comportamientos (métodos) de los objetos que se crean a partir de ella. Las clases son las definiciones de los tipos de objetos que pueden existir en el sistema.

Importancia o/y Relevancia

La clase de Java. Es la base sobre la cual se construye todo el lenguaje 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. Tener una comprensión básica de estas características te permitirá escribir programas más sofisticados y comprender mejor el lenguaje de programación Java.

Ventajas y Desventajas

Ventajas	Desventajas
Reutilización de código: Puedes crear una clase una vez y reutilizarla en diferentes programas o proyectos	**Mayor complejidad inicial:** Para programas muy pequeños, crear clases puede parecer más complicado que usar código simple.
Organización y modularidad: El código se organiza mejor, separando responsabilidades en distintas clases.	**Consumo de memoria:** Cada objeto creado ocupa memoria, lo que puede ser un problema si se instancian demasiados.
Facilita el mantenimiento y escalabilidad: Si necesitas cambiar algo, solo modificas la clase, y todos los objetos que la usan se benefician.	**Curva de aprendizaje:** Para principiantes, entender conceptos como herencia, encapsulación y polimorfismo puede ser difícil.

Ejemplo

```
1
2  public class Persona {
3      // Atributos
4      String nombre;
5      int edad;
6
7      // Constructor
8      public Persona(String nombre, int edad) {
9          this.nombre = nombre;
10         this.edad = edad;
11     }
12
13     // Métodos
14     public void saludar() {
15         System.out.println("Hola, mi nombre es " +
   nombre);
16     }
17 }
18
```

Miembros de una clase

¿Cuál es su concepto?

En Java, los atributos de clase, también conocidos como campos o variables miembro, son variables declaradas dentro de una clase. Estos atributos definen las propiedades o el estado de un objeto creado a partir de la clase.

Importancia o/y Relevancia

Comprender los atributos de las clases es fundamental para la programación orientada a objetos en Java. Los atributos de clase se utilizan para almacenar datos específicos de un objeto. Pueden tener varios modificadores de acceso para controlar su visibilidad y accesibilidad desde otras clases.

Ventajas y Desventajas

Ventajas	Desventajas
Encapsulan información: Guardan datos relevantes del objeto dentro de la clase, manteniendo la información organizada.	**Consumo de memoria:** Cada objeto creado ocupa memoria para sus atributos, lo cual puede ser costoso si se crean muchos.
Personalización de objetos: Permiten que cada objeto tenga sus propios valores, aunque compartan la misma clase.	**Mala práctica si no se controlan:** Si se declaran atributos públicos en lugar de privados, cualquiera puede modificarlos, rompiendo la seguridad del código.
Facilitan la reutilización: Una vez definidos, los atributos se pueden usar en varios métodos sin necesidad de declararlos repetidament	**Dependencia excesiva:** Si una clase tiene demasiados atributos, se vuelve rígida, difícil de mantener y poco reutilizable

Ejemplo: Definir Atributos de una clase

```
public class Car {
    private String model;
    private int year;
    private double price;
}
```

Modificadores de Acceso

¿Cuál es su concepto?

Los *modificadores* de acceso nos introducen al concepto de *encapsulamiento*. El encapsulamiento busca de alguna forma controlar el *acceso a los datos* que conforman un objeto o instancia, de este modo podríamos decir que una clase y por ende sus objetos que hacen uso de modificadores de acceso (especialmente privados) son objetos encapsulados.

Importancia o/y Relevancia

Los modificadores de acceso permiten dar un nivel de seguridad mayor a nuestras aplicaciones restringiendo el acceso a diferentes atributos, métodos, constructores asegurándonos que el usuario deba seguir una "ruta" especificada por nosotros para acceder a la información.

Ventajas y Desventajas

Ventajas	Desventajas
Encapsulamiento y seguridad: Protegen los datos de cambios accidentales desde fuera de la clase.	**Mayor complejidad inicial:** Para principiantes, decidir qué acceso usar puede ser confuso.
Control sobre la visibilidad: Permite decidir qué atributos y métodos deben ser accesibles, evitando que otros programadores dependan de detalles internos de la clase.	**Restricciones de reutilización:** Un atributo private no puede ser usado directamente fuera de la clase, lo que puede requerir métodos **getter/setter** adicionales.
Facilita el mantenimiento: Si cambias algo en un atributo privado, solo necesitas ajustar la propia clase, sin afectar al resto del programa	**Posible sobrecarga de código:** Implementar getters y setters para muchos atributos puede hacer el código más largo y pesado.

Ejemplo

```
package aap.ejemplo1;

public class Ejemplo1
{
   private int atributo1;//Este atributo es privado
   private int contador = 0; //Contador de registro

   //Si un atributo es privado podemos crear método get y set ...
   //... para éste y permitir el acceso a él desde otras instancias
```

```
   public void setAtributo1(int valor)
   {
      contador++;//Contador que lleva el registro de ediciones del atributo1
      atributo1 = valor;//Establecemos el valor del atributo
   }

   public int getAtributo1()
   {
      return atributo1;//Retornamos el valor actual del atributo
   }

   //Get para el contador
   public int getContador()
   {
      return contador;
   }
   //Notar que no ponemos un set, pues no nos interesa que el contador pueda ser cambiado.
}
```

Otras Opciones

Paquetes

¿Cuál es su concepto?

Los paquetes en Java (packages) son la forma en la que Java nos permite agrupar de alguna manera lógica los componentes de nuestra aplicación que estén relacionados entre sí.

Importancia o/y Relevancia

Los paquetes permiten poner en su interior casi cualquier cosa como: clases, interfaces, archivos de texto, entre otros. De este modo, los paquetes en Java ayudan a darle una buena organización a la aplicación ya que permiten modularizar o categorizar las diferentes estructuras que componen nuestro software.

Ventajas y Desventajas

Ventajas	Desventajas
Organización del código y estructura clara: Facilitan buscar y localizar clases relacionadas, lo que mejora la usabilidad del proyecto	**Sobrecarga organizativa si no se planifica bien:** Una estructura de paquetes mal diseñada o demasiado profunda puede dificultar la navegabilidad y sobrecomplicar la estructura del proyecto
Evitan conflictos de nombres: Permiten que múltiples clases con el mismo nombre coexistan sin colisiones, gracias al uso de namespaces diferenciados	**Complejidad inicial para principiantes:** Entender la relación entre nombres de paquetes, directorios y declaraciones de importación puede ser confuso al inicio
Control de acceso y encapsulamiento: La visibilidad puede restringirse al paquete o a subclases, reforzando la seguridad interna y la integridad del diseño	**Limitaciones al usar el paquete por defecto:** No poder reutilizar clases desde el paquete predeterminado (no nombrado) impide su importación desde otros paquetes, lo que limita su utilidad en proyectos modularizados

Ejemplo

package ruta.del.paquete;

Referencias

https://www.tokioschool.com/noticias/que-es-clase-java/

https://oregoom.com/java/clases-y-objetos/#:~:text=Las%20clases%20y%20objetos%20son,interact%C3%BAan%20dentro%20de%20un%20programa.

https://docs.oracle.com/javase/tutorial/reflect/member/index.html#:~:text=Note:%20According%20to%20The%20Java,%2C%20interfaces%2C%20and%20enumerated%20types.

https://www.datacamp.com/es/doc/java/class-attributes

https://www.programarya.com/Cursos/Java/Java-Basico#google_vignette