



**UNIVERSIDAD TECNOLÓGICA DE PANAMÁ**  
**CENTRO REGIONAL DE CHIRIQUÍ**  
**LICENCIATURA EN CIBERSEGURIDAD**



**Investigación #1**

**Asignatura:**

Programación 1

**Código:**

2S3901

**Integrantes:**

Magdalena Gonzalez 4-819-1590

**Profesor:**

Napoleón Ibarra

**PRIMER SEMESTRE**

**2025**

### 1.3. Construcción de clase (JAVA)

#### 1- ¿Cuál es su concepto?

Una clase en el lenguaje de java se trata de la base donde se definen objetivos y que a su vez es que la cuenta con los modificadores de acceso y también se le puede dar un nombre. Funciona como una estructura que agrupa datos (atributos) y comportamientos (métodos). No se limita a contener variables, sino que organiza la lógica que describe cómo se crean, utilizan y manipulan instancias dentro de un programa.

#### 2- ¿Importancia y/o relevancia?

Permite definir todo lo relacionado con los modificadores de acceso a los datos internos, facilitando así su seguridad y evitando errores. Al igual que facilita un uso reutilizable en cuanto a los componentes se refiere, ya que se reduce la duplicación de la lógica y permite que cada clase se concentre en funciones específicas en proyectos.

#### 3- Ventajas y desventajas

##### Ventajas

1. **Organización del código:** centraliza atributos y métodos en un mismo bloque.
2. **Reutilización:** permite que una misma clase se instancie varias veces, sin tener que escribir lógica repetida.
3. **Encapsulación:** los datos sensibles se protegen mediante modificadores de acceso, reduciendo riesgos de manipulación indebida.

##### Desventajas

1. **Curva de aprendizaje:** al inicio puede resultar complicado entender cómo interactúan clases y objetos.
2. **Sobrecarga en proyectos pequeños:** en sistemas muy simples, definir varias clases puede ser innecesario y aumentar el trabajo.

3. **Confusiones en su uso:** se confunde el uso de modificadores de acceso y clases hijas.

#### **4- Ejemplo**

```
public class Animal {  
    protected class Perro {  
        String nombre;  
    }  
}
```

Se crea una clase animal y una interna que esta protegida llamada perro.

### **1.3.1 Miembros de una clase**

#### **1- ¿Cuál es su concepto?**

Son los elementos internos que le dan vida a la clase, aspectos como los atributos, métodos y constructores, En el caso de los atributos son los encargos de representar el estado que tenga el objeto. Con los métodos es que se define el comportamiento y los constructores cran objetivos con sus características.

#### **2- ¿Importancia y/o relevancia?**

En el caso que la clase no cuente con estos, sería solamente un espacio vacío, ya que son los objetos los que indican aspectos como el estado y sus atributos.

#### **3- Ventajas y desventajas**

##### **Ventajas**

- 1. Centralizan la definición de datos y operaciones en un mismo lugar.**
- 2. Facilitan la personalización de objetos al permitir constructores flexibles.**

### **3. Refuerzan la coherencia del programa al encapsular comportamientos específicos.**

#### **Desventajas**

1. Requiere un gran nivel de manejo debido a que se pueden crear demasiados miembros en una clase.
2. Un mal diseño de los atributos puede provocar inconsistencias en el estado del objeto.
3. Una lógica muy dispersa en métodos hace que la clase pierda cohesión.

#### **4- Ejemplo**

```
public class Animal {  
    protected class Perro {  
        String nombre;  
        Public Perro(String nombre){  
            This.nombre = perro;  
        }  
    }  
}
```

En la clase animal y su clase interna perro, se crea el constructor con un valor específico.

#### **1.3.2 Modificadores de acceso**

##### **1- ¿Cuál es su concepto?**

Son los encargados de definir todo lo relacionado con el alcance y visibilidad de las clases, métodos o constructores. Permiten controlar quién puede acceder o modificar determinada información, lo que fortalece la seguridad y el diseño.

## **2- ¿Importancia y/o relevancia?**

Facilita la seguridad de los datos, ya que sin su presencia estarían expuesto a errores, lo que gracias a los modificadores, se puede decidir qué parte del código es accesible para todos.

## **3- Ventajas y desventajas**

### **Ventajas**

- 1. Mejoran la seguridad al ocultar datos sensibles.**
- 2. Favorecen el mantenimiento porque limitan dependencias entre clases.**
- 3. Permiten herencia controlada gracias a protected.**

### **Desventajas**

1. Un mal uso (ejemplo: abusar de public) compromete la integridad del programa.
2. El exceso de restricciones puede dificultar la reutilización de código.
3. Confundir sus reglas de alcance genera errores frecuentes en principiantes.

## **4- Ejemplo**

```
public class Animal {  
    protected class Perro {  
        private String nombre;  
    }  
}
```

La clase interna solo puede ser usada por subclases de Animal o por clases dentro del mismo paquete.

### 1.3.3. Otras opciones

#### 1- ¿Cuál es su concepto?

Además de los atributos, constructores y métodos básicos, una clase en Java admite otras construcciones que amplían su funcionalidad. Entre estas destacan:

- **Miembros estáticos (static)**, que pertenecen a la clase y no a instancias concretas.
- **Constantes (final)**, que impiden la modificación de un valor una vez asignado.

#### 2- ¿Importancia y/o relevancia?

El aporte de estas opciones radica en que permiten adaptar las clases a necesidades más específicas. Con static se evita crear objetos innecesarios para operaciones comunes, con final se garantizan valores inmutables.

#### 3- Ventajas y desventajas

##### Ventajas

1. El uso de miembros static reduce consumo de memoria en operaciones repetitivas.
2. Las constantes final mejoran la consistencia al proteger datos que no deben alterarse.
3. Las clases abstractas permiten crear jerarquías organizadas que refuerzan la coherencia del programa.

##### Desventajas

1. El abuso de static rompe el enfoque orientado a objetos y dificulta la extensibilidad.
2. Un uso excesivo de clases internas puede complicar la legibilidad del código.
3. La herencia de clases abstractas puede volverse rígida si se diseña sin cuidado.

#### 4- Ejemplo

```
public class Animal {
```

```
protected class Perro {  
    private String nombre;  
    public String getNombre() {  
        return nombre;  
    }  
    public void setNombre(String nombre) {  
        this.nombre = nombre;  
    }  
}  
}
```

La clase Perro es interna y protegida. Solo puede ser usada desde subclases de Animal o desde clases del mismo paquete.