UNIVERSIDAD TECNOLÓGICA DE PANAMÁ FACULTAD DE INGENIERÍA DE SISTEMAS COMPUTACIONALES

LICENCIATURA EN CIBERSEGURIDAD

PROGRMACIÓN I

INVESTIGACIÓN 1

PREPARADO POR:

GUERRERO STEFANY, 4-833-2081

A CONSIDERACIÓN DE

NAPOLEÓN IBARRA

GRUPO:

2S3111

I SEMESTRE

2025

Contenido

Procedimiento

TEMA:

1.3. Construcción de clase (JAVA)

1.3.1 Miembros de una clase

1.3.2 Modificadores de acceso

1.3.3 Otras opciones

Utilice una técnica conocida para desarrollar las siguientes preguntas en base al tema propuesto.

PREGUNTAS:

1. ¿Cuál es su concepto?
2. ¿Importancia y/o relevancia?
3. ¿Ventajas y Desventajas? Mínimo 3, Máximo 5 ítem.
4. Ejemplo de cada subpunto.

Desarrollo

1.3. Construcción de clase (JAVA)

¿Cuál es su concepto?

La construcción de una clase en Java consiste en crear un “molde” para objetos, definiendo sus atributos (lo que tienen), métodos (lo que pueden hacer) y constructores (cómo se crean), para poder representar cosas del mundo real dentro del programa.

¿Importancia y/o relevancia?

La construcción de clase es importante porque nos ayuda a mantener el código organizado y fácil de entender. Nos permite crear varios objetos usando la misma clase sin tener que repetir todo, ahorrando tiempo y esfuerzo.

¿Ventajas y Desventajas? Mínimo 3, Máximo 5 ítem.

Ventajas:
Mantiene el código ordenado y fácil de entender.
Permite usar el mismo molde para crear muchos objetos, ahorrando tiempo.
Facilita modificar o mejorar el programa sin complicaciones.
Desventajas:
Al principio puede ser difícil de entender si nunca has usado clases.
Si no se usan bien, los objetos pueden consumir mucha memoria.
Planear la clase y sus objetos puede tardar más que escribir código rápido.

1.3.1 Miembros de una clase

¿Cuál es su concepto?

Los miembros de una clase son todas las partes que forman una clase en programación. Por un lado, están los atributos, que son las características o datos del objeto, como su nombre, edad o color. Por otro lado, están los métodos, que son las acciones que el objeto puede hacer, como caminar, hablar o calcular algo. Juntos, los miembros le dan al objeto su identidad y le permiten funcionar, para que cada objeto creado de la clase tenga su propio estado y comportamiento.

¿Importancia y/o relevancia?

La importancia de los miembros de una clase radica en que permiten definir claramente cómo es un objeto y qué puede hacer. Además de permitir reutilizar código creando varios objetos a partir de la misma clase sin tener que escribir todo desde cero.

¿Ventajas y Desventajas? Mínimo 3, Máximo 5 ítem.

Ventajas:
Mantienen el código ordenado.
Puedes usar la misma clase para muchos objetos sin repetir nada.
Si cambias algo en la clase, todos los objetos se actualizan.
Desventajas:
Al principio puede ser difícil de entender cómo organizar todo.
Cada objeto ocupa memoria, y muchos objetos pueden gastar mucho.
Si la clase tiene errores, todos los objetos también los tendrán.

Ejemplo de cada subpunto.

```
class Perro {
  String nombre; // Atributo
  int edad;     // Atributo

  void ladrar() { // Método
    System.out.println(nombre + " dice: ¡Guau!");
  }
}
```

1.3.2 Modificadores de acceso

¿Cuál es su concepto?

Los modificadores de acceso son palabras clave que se usan para controlar quién puede ver o usar los atributos y métodos de una clase. Básicamente, determinan si algo es público, privado o protegido, lo que ayuda a proteger los datos y a organizar mejor el código.

¿Importancia y/o relevancia?

La importancia de los modificadores de acceso está en que permiten controlar la visibilidad de los atributos y métodos de una clase. Esto es relevante porque ayuda a proteger la información sensible, evita que el código de otras partes del programa haga cambios indebidos y mantiene una mejor organización y seguridad en el desarrollo.

¿Ventajas y Desventajas? Mínimo 3, Máximo 5 ítem.

Ventajas:
Protegen la información.
Hacen el código más ordenado.
Permiten decidir qué se comparte y qué no.
Desventajas:
Al inicio puede ser difícil de entender.
A veces toca escribir más código (getters y setters).
Si se usan mal, complican el programa.

Ejemplo de cada subpunto.

```
// Programa principal
public class Main {
  public static void main(String[] args) {
    // Objeto (producto final)
    Persona p1 = new Persona("Ana", 20);

    System.out.println("Nombre: " + p1.nombre);
    System.out.println("Edad: " + p1.getEdad());
  }
}
```

1.3.3 Otras opciones (Modificadores no de acceso)

¿Cuál es su concepto?

Los modificadores no de acceso en Java son palabras que se usan en clases, atributos o métodos para definir cómo se van a comportar, sin importar quién

pueda verlos. No controlan la visibilidad, sino que agregan características especiales.

¿Importancia y/o relevancia?

Son importantes porque permiten controlar cómo se usan las cosas dentro de una clase.

¿Ventajas y Desventajas? Mínimo 3, Máximo 5 ítem.

Ventajas
Ayudan a organizar mejor el código.
Evitan errores, como cambiar algo que no debe cambiar.
Permiten crear reglas claras para los objetos.
Desventajas
Si no se entienden bien, pueden confundir al programar.
Pueden hacer el código más rígido (menos flexible).
Mal usados, pueden limitar lo que se puede hacer con una clase.

Ejemplo de cada subpunto

```
class Ejemplo {
  static int contador = 0;    // static: compartido por todos
  final double PI = 3.1416;   // final: no se puede cambiar
}
abstract class Animal {       // abstract: clase incompleta
  abstract void hacerSonido(); // método abstracto, se define en otra clase
}
```

Referencias

Baeldung. (2023). *Access Modifiers in Java*. Recuperado de https://www.baeldung.com/java-access-modifiers

DataCamp. (2024). *Java Modifiers*. Recuperado de https://www.datacamp.com/doc/java/modifiers

GeeksforGeeks. (2023). *Access Modifiers in Java*. Recuperado de https://www.geeksforgeeks.org/java/access-modifiers-java/

GeeksforGeeks. (2023). *Final vs Static vs Abstract Non-Access Modifier*. Recuperado de https://www.geeksforgeeks.org/java/final-vs-static-vs-abstract-non-access-modifier/

W3Schools. (2024). *Java Modifiers*. Recuperado de https://www.w3schools.com/java/java_modifiers.asp

Cartagena99. (s.f.). *Desarrollo de Clases – Apuntes de Programación en Java*. Recuperado de https://www.cartagena99.com/recursos/alumnos/apuntes/T5-01-Desarrollo%20de%20Clases.pdf