

Prueba Diagnóstica

Ronny Wgmanía

1) respuesta B) 105



2) falta condición / 3 X

3) $i <= 100$, suma + ≠ i X

4) Análisis, Diseño, Programación, Pruebas, ejecución, mantenimiento



5) uso de primitivas X

6) Desconozco

7) Desconozco

Ronny Lugmaña

NRC: 1323

Fecha: 13-11-2024

Tarea N° 1

Pilares fundamentales de la programación orientada a objetos

1) Abstracción:

Permite enfocarse en los aspectos esenciales de un objeto, ignorando los detalles irrelevantes para el contexto. Se logra mediante la definición de clases que representan conceptos del mundo real.

2) Encapsulamiento

Consiste en ocultar los detalles internos de los objetos y exponer solo lo necesario a través de métodos públicos. Esto permite proteger los datos y controlar el acceso a ellos.

3) Herencia

Es el mecanismo que permite que una clase (subclase) herede atributos y comportamientos de otra clase (superclase), promoviendo la reutilización de código.

4) Polimorfismo

Permite que diferentes clases implementen el mismo método de manera específica. Esto facilita que el mismo mensaje desencadene comportamientos distintos según el objeto que lo reciba.

Consulta

Tipos de Datos Primitivos

Representan valores básicos que el lenguaje de programación ya tiene definidos. No son objetos, por lo que no tienen métodos ni atributos, y se almacenan directamente en la memoria.

Los tipos de datos primitivos son:

Enteros

- byte : de 8 bits

byte edad = 25;

- short : 16 bits

short salario = 15000;

- int : 32 bits

int población = 1000000;

- long : 64 bits

long distancia = 987642162160;

Decimales

- float : 32 bits , precisión simple

float altura = 1,75;

- double : 64 bits , precisión doble

double peso = 70,5;

Caracteres

- char : 16 bits , almacena un solo carácter unicode

char letra = 'A';

- boolean : Almacena dos posibles valores : true o false.

boolean esMayorDeEdad = true;

Tipos de Datos Referenciados

Los tipos de datos referenciados son objetos que almacenan una referencia a la ubicación en memoria donde se guardan los datos. Esto incluye clases, interfaces y arreglos.

Ejemplos:

- String: En Java permite tratar "String" como un tipo básico aunque sea un objeto.

- Arreglos: En Java, los arreglos son objetos que pueden almacenar múltiples elementos de un tipo específico.

- Clases y Objetos: Cualquier clase definida se convierte en un tipo de dato referenciado.

- Interfaces: Son tipos abstractos que otros tipos de datos referenciados pueden implementar.

- **String:**

```
String nombre = "Ronny";
```

- **Arreglos:**

```
int [] numeros = {1, 2, 3, 4, 5};
```

```
string [] nombres = {"Ana", "Felipe", "Kaina"};
```

- **Clases y Objetos:**

```
Persona persona = new Persona ("Andres", 30);
```

- **Interfaces:**

```
Animal perro = new Perro();
```

Los tipos primitivos almacenan el valor directamente en memoria, mientras que los tipos referenciados almacenan una referencia a la ubicación de un objeto en memoria, lo que permite crear estructuras más complejas y funcionales.

Preguntas

1) ¿Qué es paradigma de Programación Orientada a Objetos?

Es una forma de estructurar y organizar el código en función de objetos y sus relaciones. Este paradigma se basa en la idea de que el software debe modelar entidades y conceptos del mundo real o de un problema, en los cuales cada objeto representa un componente de ese sistema con sus propios atributos y comportamientos.

2) ¿Qué es una clase, objeto, atributo y método?

- **Clase**

Es una plantilla o modelo que define los atributos y métodos comunes a un tipo de objeto.

- **Objeto**

El objeto es una instancia de una clase, es decir, un ejemplar concreto que tiene sus propios valores para los atributos definidos en la clase.

- Atributos

Son las características o propiedades de una clase que describen el estado o las cualidades de los objetos de esa clase. Se implementan como variables dentro de la clase y pueden tener diferentes valores en cada objeto.

- Métodos

Son las acciones o comportamientos que pueden realizar los objetos de una clase, es la función dentro de la clase que define algo que un objeto de esa clase puede hacer o cómo puede interactuar con otros objetos o consigo mismo.

3) ¿Qué es un sistema de control de versionamiento y para qué sirve?

Un sistema de control de versiones (SCV) es una herramienta que permite registrar, gestionar y coordinar los cambios realizados en el código fuente de un proyecto a lo largo del tiempo manteniendo un historial de los cambios realizados.

¿Para qué sirve un SCV?

- Registro de cambios y seguimiento de historial

El SCV guarda un historial de cada cambio realizado en el código, permitiendo ver qué modificaciones se han hecho, quien las ha realizado y cuando. Esto nos ayuda a entender la evolución de cada componente.

- Facilita el trabajo en equipo.

Con un SCV, los cambios realizados por distintos miembros del equipo se pueden integrar de forma ordenada facilitando el trabajo simultáneo.

- Manejo de conflictos

Los sistemas de control de versiones como Git facilitan la detección y resolución de conflictos cuando hay cambios simultáneos en las mismas partes del código.

- Recuperar versiones anteriores

Si un cambio provoca errores en el sistema, el SCV permite regresar a una versión anterior del código que estaba funcionando correctamente.

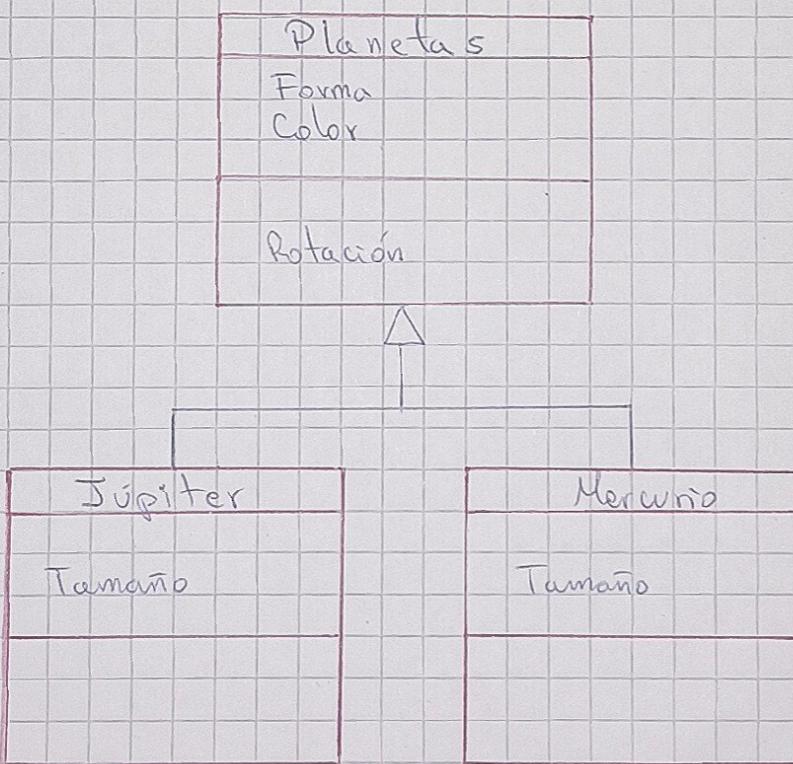
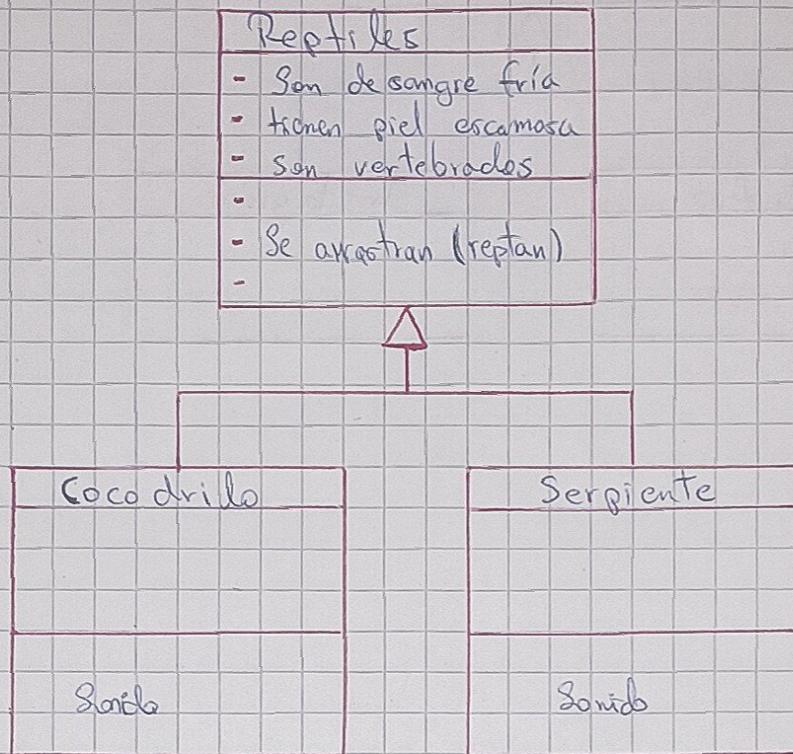
- Implementación de nuevas funcionalidades

El SCV permite crear ramas (branches) para desarrollar nuevas funcionalidades sin afectar al código principal.

• Documentación y responsabilidad

Los SCV suelen registrar quién realizó cada cambio y pueden incluir mensajes de registro (commits) que documentan el motivo de cada cambio.

4) Diseñe 3 NML de 2 clases hijas, 1 clase padre



Redes Sociales

- Plataformas digitales
- funcionan con internet
- Comunicación tiempo real
- Intercambio de información



WhatsApp

Instagram