Trabajo práctico Nro. 1



| Logo Departamento 2.jpg | **Asignatura: Programación II** | |
| --- | --- | --- |
|  | |
| **Cursado:** Segundo semestre | **Horas semanales**: |
|  | **Horas semestrales:**  *Cantidad estimada de horas*  *semestrales/anuales.* |
| **Carrera**: *Tecnicatura Universitaria en*  *Programación* | **Nivel (Año):**  1°  2°  3° |
| **Ciclo Lectivo: 2024** |

**Integrantes de la Cátedra:**

* **DOCENTES:**

| **Nombre del Profesor** | **Periodo** | **Cantidad horas materia** |
| --- | --- | --- |
| **Cinthia Rigoni** |  | 6 horas |

* 1. **¿Qué es un TAD?**

TAD (Tipo Abstracto de Datos o Tipo de Datos Abstractos):

Se refiere a una especificación matemática de un conjunto de datos y las operaciones que pueden realizarse en esos datos.

Es un modelo que define valores y las operaciones que se pueden realizan sobre ellos. Y se denomina abstracto ya que la intención es que quien lo utiliza, no necesita conocer los detalles de la representación interna o bien el cómo están implementadas las operaciones.

Es por esto una práctica que nos provee un grado de abstracción que permite desacoplar al código que usa un TDA de aquel código que lo implementa.

Los TAD son útiles porque permiten a los programadores trabajar con abstracciones de datos sin preocuparse por los detalles de implementación. Por ejemplo, un TAD "pila" podría definir operaciones como "apilar" y "desapilar", pero no especificaría cómo se implementan estas operaciones. Esto permite a los programadores cambiar fácilmente la implementación subyacente sin afectar el resto del programa, siempre y cuando se mantenga la misma interfaz de operaciones definida por el TAD.

* 1. **¿Dónde se produce el encapsulamiento?**

El encapsulamiento es la separación del "qué" del "cómo". Esta distinción implica que el "qué" se refiere a la representación del objeto, mientras que el "cómo" se relaciona con su comportamiento dentro del sistema.

El encapsulamiento se describe como el proceso de agrupar en una misma sección los elementos de una abstracción que constituyen tanto su estructura como su comportamiento. Esta práctica ayuda a separar la interfaz entre una abstracción y su implementación.

Existen tres niveles de acceso para el encapsulamiento, los cuales son:

Público (Public): Permite que todos los datos o métodos de una clase sean accesibles desde cualquier parte del programa. Este nivel es el más bajo y se utiliza para lo que se desea que la parte externa del programa vea.

Protegido (Protected): Este nivel restringe el acceso a los datos o métodos de una clase, limitándolos a la clase misma y a sus subclases. Es una forma de proteger ciertos aspectos de la clase.

Privado (Private): En este nivel, los miembros solo son accesibles dentro de la propia clase. Se utilizan para ocultar detalles de implementación y evitar interferencias externas.

* 1. **¿Cuáles son las semejanzas y las diferencias entre funciones, procedimientos y métodos?**

| Características | Funciones | Métodos | Procedimientos |
| --- | --- | --- | --- |
| Tipo de bloque | Conjunto de líneas de código encapsuladas | Conjunto de líneas de código encapsuladas | Conjunto de líneas de código encapsuladas |
| Parámetros | Pueden recibir parámetros | Pueden recibir parámetros | Pueden recibir parámetros |
| Retorno de valor | Retornan un valor | Retornan un valor | No retornan valor |
| Asociación | Independientes de un objeto | Asociados a un objeto o clase | Asociados a un objeto o clase |
| Contexto | No necesitan un objeto para ser utilizadas | Necesitan un objeto o clase para ser usados | No necesitan un objeto para ser usados |
| Modificador en Java | Utilizan el modificador static | No necesitan el modificador static | Utilizan el tipo de retorno void |

**Diferencias:**

**Asociación:** Las funciones existen por sí solas y no están asociadas a un objeto específico, mientras que los métodos están asociados a un objeto o clase en Java. Los procedimientos, aunque pueden recibir argumentos, no están asociados a un objeto específico y no retornan ningún valor.

**Tipo de Retorno:** Las funciones y los métodos pueden retornar un valor, mientras que los procedimientos no retornan ningún valor (tienen un tipo de retorno void en Java).

**Modificador en Java:** Las funciones en Java suelen usar el modificador static, mientras que los métodos no necesitan este modificador. Los procedimientos, al ser considerados métodos con tipo de retorno void, no necesitan este modificador.

**Semejanzas:**

**Bloque de Código:** Tanto las funciones, los métodos y los procedimientos son conjuntos de líneas de código encapsuladas.

**Parámetros:** Todos pueden recibir parámetros, es decir, valores que se utilizan para realizar operaciones dentro de su bloque de código.

**Utilidad:** Las tres estructuras permiten encapsular un conjunto de instrucciones para su reutilización y modularidad en un programa.

* 1. **¿Qué es UML? ¿Y cómo se representa una clase en ese lenguaje?**

**UML:** Unified Modeling Language (Lenguaje Unificado de Modelado), es un lenguaje estándar utilizado en el campo de la ingeniería de software para visualizar, especificar, construir y documentar sistemas software.

UML es un lenguaje de modelado, de propósito general, usado para la visualización, especificación,   
construcción y documentación de sistemas Orientados a Objetos.

Una clase en UML se representa mediante un rectángulo dividido en tres secciones horizontales. En la parte superior del rectángulo se coloca el nombre de la clase. En la sección intermedia se listan los atributos de la clase, que son las variables que describen las características o estados de los objetos de esa clase. Estos atributos suelen incluir el tipo de dato y el nombre del atributo. En la sección inferior se listan los métodos de la clase, que son las funciones que describen el comportamiento de los objetos de esa clase. Estos métodos suelen incluir el nombre del método, los parámetros de entrada y el tipo de retorno.

* 1. **Marcar con cruz.**

|  | No se aplica a clases | Solo se aplica a atributos | Solo se aplica a clases | Se aplica a atributos, métodos y clases |
| --- | --- | --- | --- | --- |
| Public |  |  |  | X |
| Private |  |  |  | X |
| Protected |  |  |  | X |
| Static | X |  |  |  |
| Final |  |  |  | X |
| Primera letra en  minúscula |  | X |  |  |
| Primera letra en  mayúscula |  |  | X |  |

* 1. **Verdadero o falso**
* Un constructor…
  + Es el método principal para ejecutar un programa. **F**
  + Crea instancias. **V**
  + Devuelve el valor de un atributo privado. **F**
  + Tiene sentencia return. **F**
  + Siempre existe uno por defecto, sin parámetros ni inicializaciones de atributos. **V**
  + Se puede sobreescribir. **F**
  + Se puede sobrecargar. **V**
  + Su nombre se escribe con mayúscula. **V**
  + Su calificador de acceso es static. **F**
  + Su tipo de devolución no se indica y corresponde a la clase. **V**
* Un método…
  + Puede tener múltiples parámetros con el mismo nombre, siempre y cuando tengan tipos diferentes. **V**
  + Puede sobrecargarse. **V**
  + Puede sobreescribirse. **V**
  + Puede ser static. **V**
  + Puede ser tanto public como protected, pero no private. **F**
  + Un método puede tener un modificador de acceso final. **V**
  1. Calificadores de acceso. Completa.
     1. Se necesita que cualquiera pueda acceder al color de un vehículo. Entonces, declaro color como:

public

* + 1. Se necesita que color se pueda acceder a través no sólo de vehículo, sí no ahora también de Buses, y como todos sabemos un bus es un tipo de vehículo, entonces también deberá tener acceso a color. Entonces, declaro color como: protected

c- Se necesita que color se pueda acceder solamente para vehículo. Entonces, declaro color como:

private

**A CODIFICAR!**

* 1. Se desea desarrollar un sistema de gestión de empleados para una empresa. El sistema debe permitir registrar empleados de dos tipos diferentes: gerentes y trabajadores. Cada empleado debe tener un nombre, una edad y un salario.

Los gerentes tienen la capacidad de organizar actividades dentro de un departamento específico, mientras que los trabajadores están encargados de producir en un área determinada.

Implementa un sistema que modele esta situación utilizando herencia en Java. Define una clase base llamada Empleado que contenga los atributos y métodos comunes para todos los empleados, como el nombre, la edad, el salario y la capacidad de trabajar.

Luego, crea dos subclases: Gerente y Trabajador, que hereden de Empleado. Los gerentes deben tener un atributo adicional para almacenar el departamento en el que trabajan, así como un método para organizar actividades dentro de ese departamento.

Por otro lado, los trabajadores deben tener un atributo para indicar el área en la que trabajan y un método para producir en esa área.

Finalmente, en el programa principal, crea instancias de ambas subclases y muestra cómo se utilizan los métodos específicos de cada tipo de empleado, así como sus atributos.

* 1. Se desea implementar un programa en Java para modelar diferentes figuras geométricas, como círculos y rectángulos. Cada figura geométrica debe tener la capacidad de calcular su área y su perímetro.

Define una clase base llamada FiguraGeometrica que contenga métodos abstractos para calcular el área y el perímetro de la figura. Luego, crea subclases para representar diferentes tipos de figuras geométricas, como Circulo y Rectángulo, que hereden de la clase base FiguraGeometrica.

En la subclase Círculo, implementa métodos para calcular el área y el perímetro de un círculo, utilizando el radio como atributo de la clase. En la subclase Rectángulo, implementa métodos para calcular el área y el perímetro de un rectángulo, utilizando la longitud y la anchura como atributos de la clase.

En el programa principal, crea instancias de diferentes figuras geométricas (al menos un círculo y un rectángulo) y muestra sus áreas y perímetros.

* 1. Definir una clase Libro para manejar la información asociada a un libro. La información de interés para un libro es: el título, el autor y el precio. Los métodos de interés son:
* Un constructor para crear un objeto libro, con título y autor como parámetros.
* Imprimir en pantalla el título, los autores y el precio del libro.
* Métodos get y set para cada atributo de un libro.

Se debe extender la clase Libro definiendo las siguientes clases:

* Libros de texto con un nuevo atributo que especifica el curso al cual está asociado el libro.
* Libros de texto de la Universidad Nacional de Colombia: subclase de la clase anterior. Esta subclase tiene un atributo que especifica cuál facultad lo publicó.
* Novelas: pueden ser de diferente tipo, histórica, romántica, policíaca, realista, ciencia ficción o aventuras.

Para cada una de las clases anteriores se debe definir su constructor y redefinir adecuadamente el método para visualizar del objeto.