| EJERCICIO PRÁCTICO  Desarrollo de Aplicaciones Web  (1WET)  Entornos de Desarrollo |
| --- |

| **DOCENTE**  Jorge César Montoya Valero. | **ALUMNO**  Hugo Ruiz Sánchez  . |
| --- | --- |

| **ENUNCIADO**  Intenta explicar que entiendes de los conceptos de Java del punto 2 de teoría (abstracción, encapsulación...) con una explicación sencilla y con algún ejemplo de uso en un programa Java. |
| --- |

* Abstracción.

En java, la abstracción permite la programación de clases y métodos iniciales cuyo objeto su utilización como referencia para arbolar el contenido del programa durante fases posteriores del desarrollo.

Un ejemplo de esto puede ser el de “Ser Vivo”:

| public abstract class SerVivo {    public abstract void alimentarse ();   } |
| --- |

* Encapsulación y principio de ocultación:

La encapsulación aísla unas clases con otras, y generalmente se aplica con los denominados métodos “get” y “set”, aplicadas generalmente a clases privadas para prevenir la modificación de los datos :

| private int numero; public int getNumero () { return numero; } public void setNumero (int numero) { this.numero = numero; } |
| --- |

* Herencia:

Adquirir, partiendo de un objeto, propiedades pertenecientes a diferentes objetos.

Por ejemplo, de la clase “SerVivo” creada anteriormente, puede derivarse la clase “Planta”, que a su vez está sobrecargando el método “Alimentarse” definido:

| public class Planta extends SerVivo {    @Override  public void alimentarse () {  System.out.println ("La planta se alimenta a través de la fotosíntesis.");  }   } |
| --- |

En el caso de las plantas, puede instanciarse de la siguiente forma:

| public static void main(String[] args) {    Planta pla1 = new Planta ();  } |
| --- |

* Polimorfismo

Partiendo de un objeto padre, se extienden sus propiedades a sus hijos; permitiendo instanciarlos con mayor facilidad en el programa.

Véase, la clase padre “vehículo”:

| public class VehiculoTurismo extends Vehiculo {    // ATRIBUTOS    private int numeroPuertas;    // CONSTRUCTOR    public VehiculoTurismo (String matricula, String marca, String modelo, int numeroPuertas) {  super (matricula, marca, modelo);  this.numeroPuertas = numeroPuertas;  }     // MÉTODOS OVERRIDE    @Override  public void mostrarDatos () {    System.out.println ("Matrícula: "+matricula);  System.out.println ("Marca: "+marca);  System.out.println ("Matrícula: "+modelo);  System.out.println ("Número de puertas: "+numeroPuertas);    } |
| --- |

De la que heredan “VehiculoFurgoneta”:

| public class VehiculoFurgoneta extends Vehiculo {    // ATRIBUTOS    private int carga;    // CONSTRUCTOR    public VehiculoFurgoneta (String matricula, String marca, String modelo, int carga) {  super (matricula, marca, modelo);  this.carga = carga;  }    // MÉTODOS OVERRIDE    @Override  public void mostrarDatos () {    System.out.println ("Matrícula: "+matricula);  System.out.println ("Marca: "+marca);  System.out.println ("Matrícula: "+modelo);  System.out.println ("Carga: "+carga);    } |
| --- |

Y “VehiculoDeportivo”:

| public class VehiculoDeportivo extends Vehiculo {    // ATRIBUTOS    private int cilindrada;      // CONSTRUCTOR    public VehiculoDeportivo (String matricula, String marca, String modelo, int cilindrada) {    super (matricula, marca, modelo);    this.cilindrada = cilindrada;    }    // MÉTODOS OVERRIDE (SOBREESCRITURA)    @Override  public void mostrarDatos () {    System.out.println ("Matrícula: "+matricula);  System.out.println ("Marca: "+marca);  System.out.println ("Matrícula: "+modelo);  System.out.println ("Cilindrada: "+cilindrada);    } |
| --- |

Que pueden instanciarse a partir de la clase “Vehiculo” desde el método principal:

| public static void main(String[] args) {    Vehiculo [] misVehiculos = new Vehiculo [4];    misVehiculos [0] = new Vehiculo ("12345678", "Ferrari", "A89");  misVehiculos [1] = new VehiculoTurismo ("87654321", "Audi", "P14", 4);  misVehiculos [2] = new VehiculoDeportivo ("246810", "Toyota", "KJ8", 500);  misVehiculos [3] = new VehiculoFurgoneta ("359112", "Toyota", "JJ9", 2000); } |
| --- |