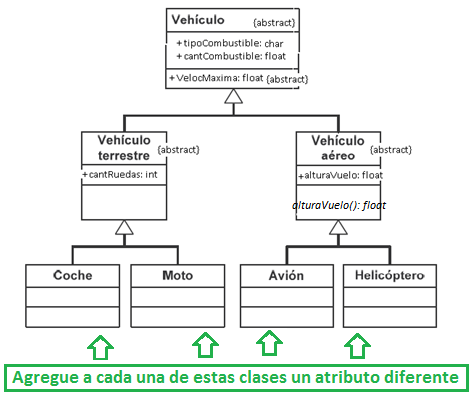
|  |  |
| --- | --- |
| Alumno: Cristian Krahulik, Lautaro Castillo, Alex Filchel, Tomas Mastropietro, Gianfranco Guzman | **Fecha límite de entrega: 14/4/2025** |

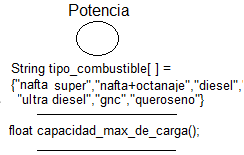
Herencia - Interfaces

1. Codifique el siguiente modelo

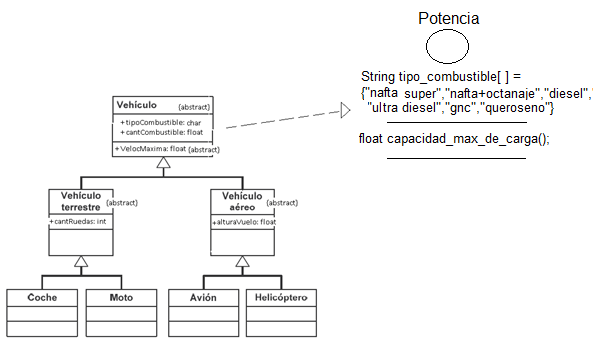
**

Pruebe el funcionamiento de la siguiente manera:

1. Implemente todo el modelo.
2. Si al método alturaVuelo() en la clase Avión, se le califica final. Indique verdadero o falso en las siguientes afirmaciones:
   1. El método se puede sobrecargar. **VERDADERO**
   2. El método se puede sobreescribir o re-definir. **FALSO**
   3. El método se puede ejecutar sólo desde un objeto de la clase Avión. **VERDADERO**
   4. El método se puede ejecutar sólo desde un objeto de la clase VehículoAereo. **FALSO**
   5. El método se puede ejecutar invocando desde la clase. **FALSO**
3. Cree un método para mostrar atributos en la superclase. Luego, cree un método con el mismo nombre en la subclase (redefinición), que:
   * 1. Muestre sus atributos propios.
     2. Para mostrar los atributos heredados, que invoque al método de la superclase.
4. Cree un objeto de cada subclase concreta y muestre los atributos propios y heredados.
5. Desde uno de los objetos:
   1. Active un método heredado y muestre su devolución.
   2. Muestre el valor de un atributo propio y de uno heredado.
6. Agregue al modelo anterior, la implementación de la siguiente interfaz, por parte de la superclase:



1. La interface es implementada por la clase Vehiculo.
2. Cree un método en la clase Vehiculo con las siguientes características:
   1. Tipo de devolución: void.
   2. Nombre: desdePotencia
   3. Parámetros que recibe: dos enteros. El primero indica la posición en el arreglo: tipo\_combustible; el segundo indica la cantidad de combustible que requiere.
   4. Implementación: el método debe tomar el elemento del arreglo y mostrarlo con el segundo parámetro recibido. Siempre deberá controlar que el entero recibido como parámetro corresponda a las posiciones ocupadas del arreglo. Si no fuera así, debe indicarse con un mensaje.
   5. En el main, desde alguna de las clases, se debe ejecutar el método implementado.
   6. También desde un objeto de alguna clase concreta, implemente el método obtenido de la interface Potencia.



1. Vemos que las clases Vehículo, VehiculoTerrestre y VehiculoAereo son abstractas. Esto indica que tienen al menos un método sin implementar que es el que provee la interface Potencia: capacidad\_max\_de\_carga().

Este debe ser implementado en las subclases Coche, Moto, Avión y Helicóptero.

Cada uno tendrá un funcionamiento diferente, y todos deben tener tipo de devolución float.

1. Luego en la clase ejecutable: main()
   1. Cree un objeto de alguna subclase concreta (Coche, Moto, Avión y Helicoptero).
   2. Muestre en pantalla el/los atributos del objeto creado, aclarando a qué clase corresponde (use getClass() ).
   3. Muestre en pantalla, el resultado de la ejecución de los métodos velocMaxima() y capacidad\_max\_de\_carga(), indicando con un texto desde qué clase se ejecuta.
2. Indique cuáles de las siguientes afirmaciones son verdaderas comprobando en el código:

Una clase abstracta…

* 1. no puede ser calificada como final. **VERDADERO**
  2. no puede tener métodos abstractos.
  3. puede heredar de otra clase no abstracta. **VERDADERO**
  4. no puede heredar de otra clase abstracta.
  5. necesita en su cabecera la palabra abstract. **VERDADERO**
  6. no necesita en su cabecera la palabra abstract.
  7. tiene al menos un método abstracto. **VERDADERO**
  8. tiene todos sus métodos abstractos.

Una interface …

* 1. tiene al menos una variable no final.
  2. tiene todos sus métodos abstractos. **VERDADERO**
  3. todos sus atributos static y final. **VERDADERO**
  4. tiene al menos uno de sus atributos private, static y final.

1. Responda investigando.
   1. Dé ejemplos de interfaces existentes en java.

La interface Set que hereda de Collection. Array para los arreglos. List para listar. La interface Iterator.

* 1. ¿Existe la herencia de interfaces?

No existe la herencia de interfaces. Cuando se utiliza la palabra extends para marcar una herencia, java espera clases concretas o abstractas, no permite interfaces después de extends.

* 1. ¿Puede una subclase implementar a una interface?

Si es posible.

* 1. ¿Es posible que una clase implemente más de una interface?

Si es posible que se implemente mas de una interface

Punto 5.

Afirmaciones correctas: 🡪 -a-,-c-,-e-,-g-,-j-,-k-