Juan Nicolás Bustos Ramírez Código 286001

Jhoan Sebastián Restrepo Mendoza Código 2548926

Taller 1

Estructuras de Datos

Departamento de Ingeniería de Sistemas e Industrial

Facultad de Ingeniería

1. Explicar y dar un ejemplo de:
2. Abstracción.

Es el modelamiento de las características básicas y útiles de un objeto o concepto proveniente de la realidad, que será utilizado como base para la creación de una clase de objetos.

Ejemplo: Se puede abstraer un carro, en donde se tendría en cuenta el número de puertas y el color, o cualquier otro atributo que se desee tener en cuenta.

1. Clase.

Se encarga de alojar los métodos (tareas a realizar) y los atributos de los elementos que pertenezcan a la clase.

Ejemplo: A la clase Autos, pueden pertenecer distintos autos como auto1 o auto2, con distintos colores y números de puertas, pero todos con la misma funcionalidad básica.

1. Objeto.

Son elementos creados a partir de una clase, de modo poseen sus atributos y comportamientos. Puede haber objetos con las mismas características pero serán objetos distintos.

Ejemplo: Todos los objetos pertenecientes a la clase Autos tendrán color y número de puertas, como la opción de acelerar o frenar. Y aunque haya dos autos con el mismo color y número de puertas serán distintos objetos.

1. Atributo.

Son características que poseen los objetos pertenecientes a una clase y pueden variar entre objetos, pero todos los objetos las tendrán.

Ejemplo: Todo carro debe tener un número de matrícula, un color, un número de ruedas, un modelo, una marca. Todos estos son atributos.

1. Método.

Definen el comportamiento de los objetos pertenecientes a una clase, es decir, las tareas que cumplirá este, o que podrá realizar.

Ejemplo: Dos métodos de la clase Auto serían acelerar y frenar, de modo que se cambien atributos de los objetos, como velocidad y posición.

1. Encapsulamiento.

Las clases envuelven los atributos y métodos en los objetos. Los objetos de diferentes clases se pueden comunicar entre sí, pero por lo general no tienen acceso a saber cómo es el funcionamiento de otros objetos; los detalles de implementación están ocultos dentro de los mismos objetos. Este ocultamiento de información se conoce como encapsulamiento.

Ejemplo: La clase Auto posee un objeto auto1 que posee una velocidad, y la clase Radar posee un objeto velocímetro. Este puede acceder a la velocidad de auto1, poro no puede saber cómo varía esta internamente en el auto.

1. Herencia.

Con la herencia se pueden crear clases de un modo más eficiente, de modo que la nueva clase absorba las características de la clase existente, con la opción de personalizarlas y añadir sus características propias.

Ejemplo: De la clase Auto, puede ser heredada las clases BMW y Audi, que pueden tener distintos métodos, como prenderRadio, o parqueoAutomatico, o atributos como techoCorredizo.

1. Polimorfismo.

Es un mecanismo utilizado para que las clases heredadas de una clase superior (superclase) puedan utilizar el mismo método, que todas deben tener, pero de un modo distinto.

Ejemplo: Audi y BMW, que heredan de Auto, tienen que poder acelerar y frenar, pero seguramente lo harán de modos distintos, así que se define el comportamiento dentro de cada clase.

1. Diferencia entre tipo primitivo y de referencia.

Las variables de tipo primitivo son normalmente conocidas que asignan un espacio de memoria y solo pueden almacenar un valor de su tipo de dato, mientras que las variables de referencia son creadas por usuarios, pueden venir en bibliotecas o ser programadas por nuevas personas. En este caso se pueden almacenar distintos tipos de datos, incluyendo otros datos de referencia, y modificar sus valores o asignarles comportamientos.

Ejemplo: Un dato de referencia es un int, es decir un entero que podría definir el número de ventanas de nuestro auto, y un dato de referencia sería el quattro, un objeto perteneciente a la clase Audi, con métodos y distintos atributos.

1. Diferencia entre usar private, protected, public, default.

Se pueden declarar datos y métodos de distintos tipos. El tipo private indica que solo pertenece a esta clase y no podrá ser modificado, sobreescrito o heredado por ninguna otra clase, mientras que el tipo default permite que las clases que pertenecen al paquete donde se encuentra la clase también puedan acceder a los parámetros. En el caso de protected, se puede acceder no solo desde las clases pertenecientes al paquete, sino también desde clases heredadas, y en el caso de public, todas las clases pueden acceder a los datos.

Ejemplo: Los datos que se deseen únicamente para BMW, se declaran como private, los que hereda de Auto son protected, y los métodos de acelerar y frenar pueden ser públicos, para que otra clase los utilice.

1. Clase abstracta.

Estas clases no pueden poseer objetos, así que siempre serán superclases ya que obligatoriamente deberán tener al menos un método que sea abstracto, es decir, que no se encuentre definido en esta clase y deba ser completado en las subclases herederas.

Ejemplo: La clase Auto tendría que ser abstracta para poseer los métodos acelerar y frenar que serán redefinidos en las subclases BMW y Audi, por lo que la clase Auto no podría tener un objeto instanciado desde esta, sino desde BMW o Audi.

1. Interfaz.

Las interfaces funcionan como una clase abstracta, pero en este caso no hay ningún método declarado explícitamente, y no hace falta que se implemente la herencia, sino que una clase externa puede implementar la interfaz.

Ejemplo: La clase Audi podría implementar una interfaz donde se encuentran los métodos para el uso de un GPS.