Da Vinci
PRIMERA ESCUELA DE ARTE MULTIMEDIAL

Clases

Este capítulo describe como agrupar objetos en clases y cómo se heredan las características de una clase.

Da Vinci
PRIMERA ESCUELA DE ARTE MULTIMEDIAL

Objetivos

- Agrupar objetos con atributos similares y operaciones comunes en clases.
- Explicar cómo se utilizan las clases para definir objetos.
- Definir herencia y explicar cómo se relaciona con la reutilización de software.
- Definir generalización y especialización y su relación con la herencia.
- Definir polimorfismo y explicar cómo la herencia fomenta el polimorfismo.
- Definir clases abstractas.

Panorama General de las Clases

- **Un Objeto es una instancia :** una hoja, pelota, coche o moneda específico

- **“Coche" es una clase:** "mi coche" es un objeto de esa clase

Panorama General de las Clases

Ejemplo:

Las hojas de arce (maple), roble (oak) y álamo (poplar) son todas diferentes entre sí, pero comparten muchas características. Algo similar ocurre con las pelotas de fútbol, basquetbol y béisbol. Ellas lucen diferente y se utilizan de manera diferente, pero comparten varios rasgos.

Da Vinci
PRIMERA ESCUELA DE ARTE MULTIMEDIAL

Panorama General de las Clases

Definición:

Los objetos que tienen esa clase de similitud se agrupan en ***clases***. En el ejemplo, cada hoja es un objeto de la clase Hoja; las pelotas son objetos de la clase Pelota. Las clases no son objetos, sino que describen objetos. La clase es una plantilla para los objetos.

Panorama General de las Clases

Da Vinci
PRIMERA ESCUELA DE ARTE MULTIMEDIAL

Panorama General de las Clases

Definición:

Los objetos son instancias de ciertas *clases*. Las clases no son objetos porque carecen de *atributos* y *operaciones*. Los atributos y operaciones definidas por una clase son para sus objetos, no para la clase. No existe ninguna realización concreta de la clase HojaArce, pero sí existen objetos de esa clase.

Cuando se hace referencia a "una hoja", se habla de la clase Hoja; ya que se está haciendo referencia a cualquier instancia de la clasificación Hoja. Cuando se hace referencia a "esa hoja al final de la rama", se habla de un objeto, ya que es una instancia particular de la clase Hoja.

Panorama General de las Clases

Caso de Estudio

Para cada objeto del sistema de ingreso de órdenes de DirectClothing, se debe definir la clase correspondiente.

Generalización

- **Ejemplo:** Transporte es la generalización de varias clases que ofrecen servicios de traslado

- La Generalización identifica y define los atributos y operaciones comunes en una colección de objetos.

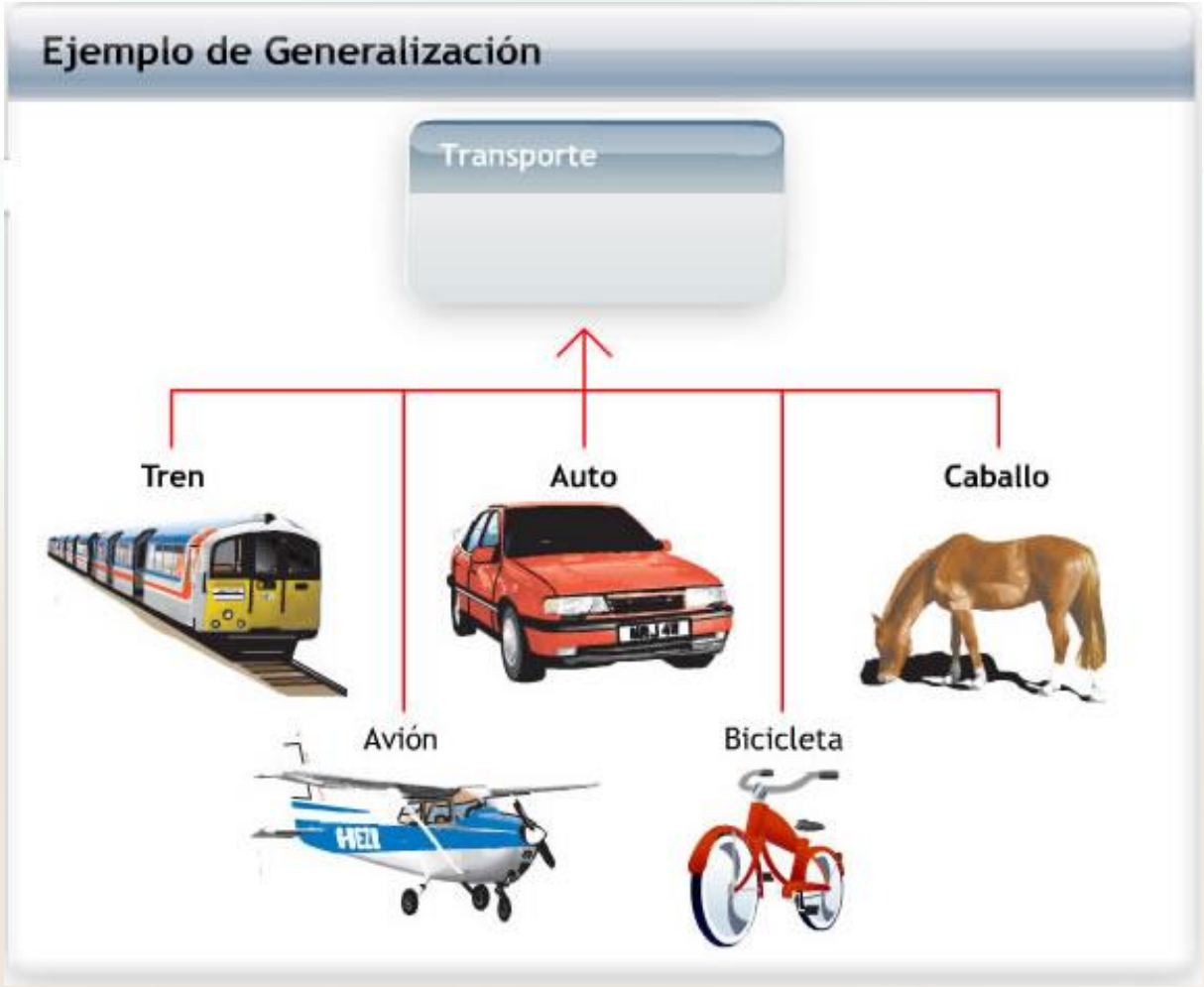
Generalización

Ejemplo:

Cuando se estudian los objetos, es conveniente clasificarlos en grupos con similares atributos y operaciones comunes. Por ejemplo, una clase Transporte puede incluir objetos tren, objetos coche e incluso objetos caballo. El hecho que dichos objetos de transporte operen en diferentes formas puede no ser relevante en el problema que se quiere resolver.

Si Transporte tiene que ver únicamente con moverse de un sitio a otro, entonces no interesa cuál sea la implementación que lo soporte. Las características de la clase Transporte pueden ser reutilizadas en todas las clases que estén contenidas en ella: Tren, Bicicleta, Avión, etc.

Generalización



Generalización

Caso de Estudio

Analizando los objetos “representante de servicio al cliente” (CSR) y “vendedor que ingresa la orden” (OEC), se determina que ambos son idénticos y se crea la clase Employee (Empleado). Se puede instanciar la clase Empleado y crear los objetos CustomerServiceRepresentative u OrderEntryClerk. Cuando se evalúan los atributos y las operaciones del objeto Customer y la clase Employee, se encuentran ciertas similitudes. Ambos tienen un único número, nombre, apellido y dirección. Se podría crear una clase Person que contenga dichos atributos. También se podría crear una clase Address (Dirección), puesto que el objeto Customer tiene dos direcciones.

Discusión – ¿Qué otras generalizaciones puede encontrar en los objets?

Herencia

Ejemplo:

La Clase PelotaFutbol puede heredar los miembros de la clase Pelota.

La clase Pelota es la clase más general en la siguiente jerarquía. Sus métodos, como por ejemplo lanzar, y sus atributos, como por ejemplo forma y tamaño, son compartidos, o heredados, por las subclases.

Herencia


Subclases heredadas desde una Superclase
Superclase
Pelota
Baloncesto
Futbol
Baseball
Subclases

Herencia

Definición:

Herencia es un mecanismo para definir una nueva clase en términos de una clase ya existente. Permite agrupar clases relacionadas de forma tal que puedan ser consideradas y controladas colectivamente y fomentar la reutilización. En el ejemplo anterior, la clase Pelota es la clase más general, o superclase, y las subclases específicas comparten o heredan miembros de la superclase tales como el atributo forma y el método lanzar. Como las clases comparten miembros, estos pueden ser manejados sólo una vez y los efectos tienen lugar en ambas.

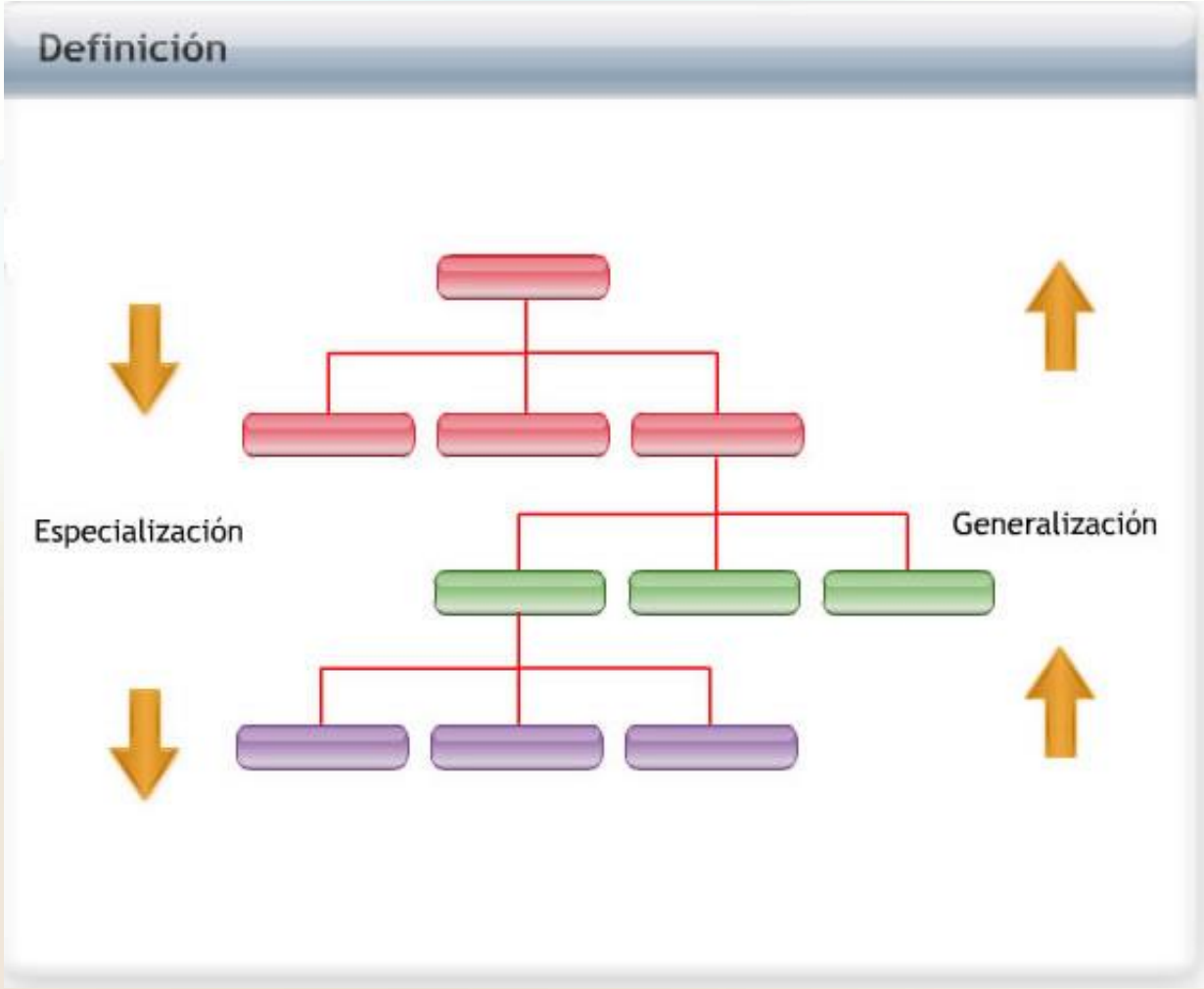
Herencia

Definición:

La herencia suele representarse como un árbol. Avanzando hacia las hojas, la clases resultan más especializadas, refinadas para una aplicación precisa. Recorriendo de las hojas hacia la raíz, las clases son más generales; contienen miembros adecuados para muchas clases pero suelen no estar completas.

Los objetos rara vez ocupan grupos completamente diferentes, en general hay muchas superposiciones en las operaciones de los grupos de objetos.

Herencia



Herencia
Modificación de Miembros Heredados

Da Vinci
PRIMERA ESCUELA DE ARTE MULTIMEDIAL

Muchas veces, cuando se especializa una clase, se puede desear modificar la implementación de algunos de sus comportamientos. Para retocar los miembros de una subclase, se puede sobreescribir los miembros de una clase. La sobreescritura mantiene los miembros pero cambia la implementación.

Se podría necesitar modificar la implementación de un miembro para tener en cuenta las particularidades de una clase especializada. Ciertas subclases pueden sobreescribir miembros de acuerdo con el contexto específico donde son utilizadas.

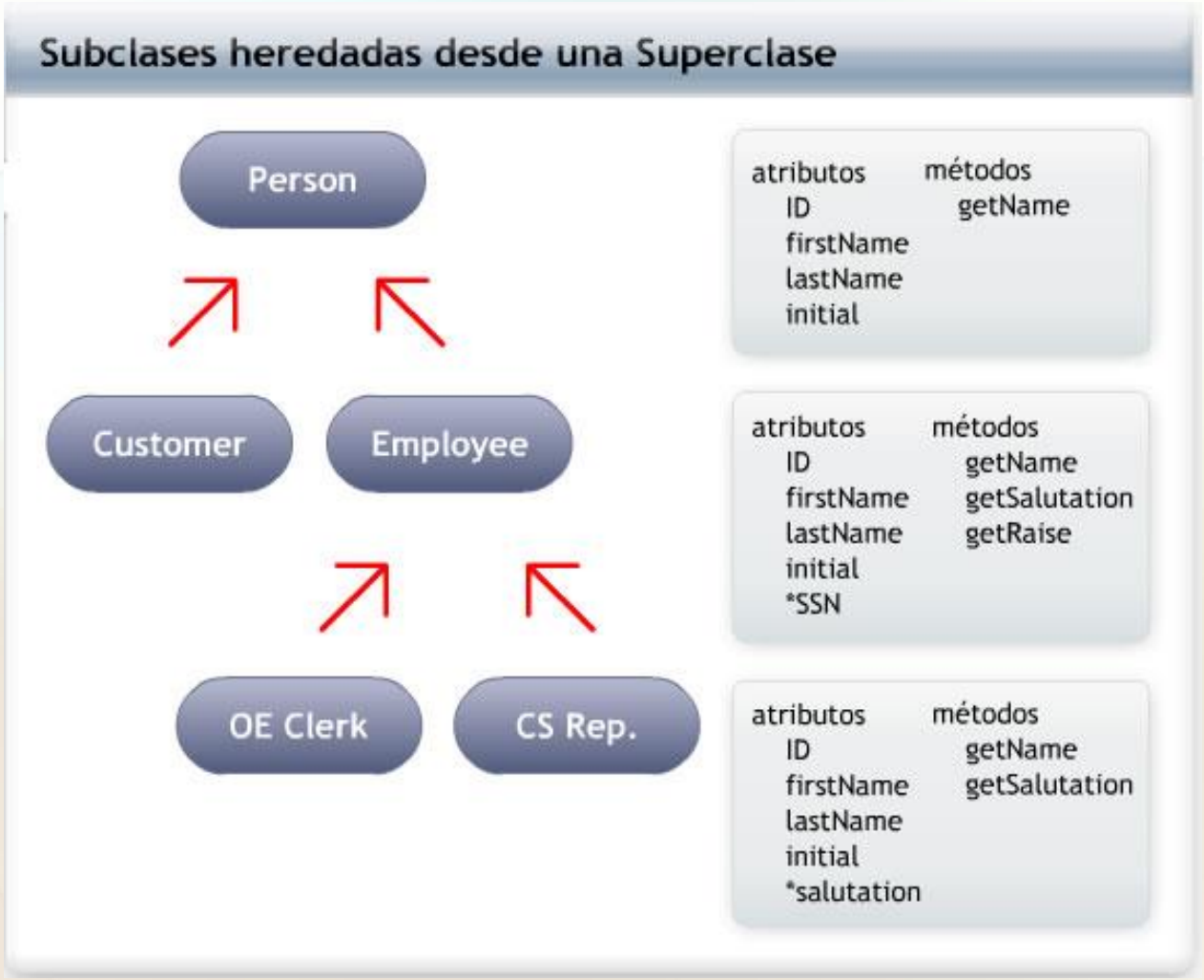
Herencia

Caso de Estudio

La compañía DirectClothing utiliza un objeto Person del que se hereda información general, como por ejemplo nombre (firstName, lastName, initial) e ID. Tanto la clase Customer como la clase Employee extienden a Person.

Discusión – ¿Qué otras clases en el sistema de ingreso de órdenes, serían apropiadas para fomentar la reutilización de miembros a través de la herencia?

Herencia – Caso de estudio



Especialización

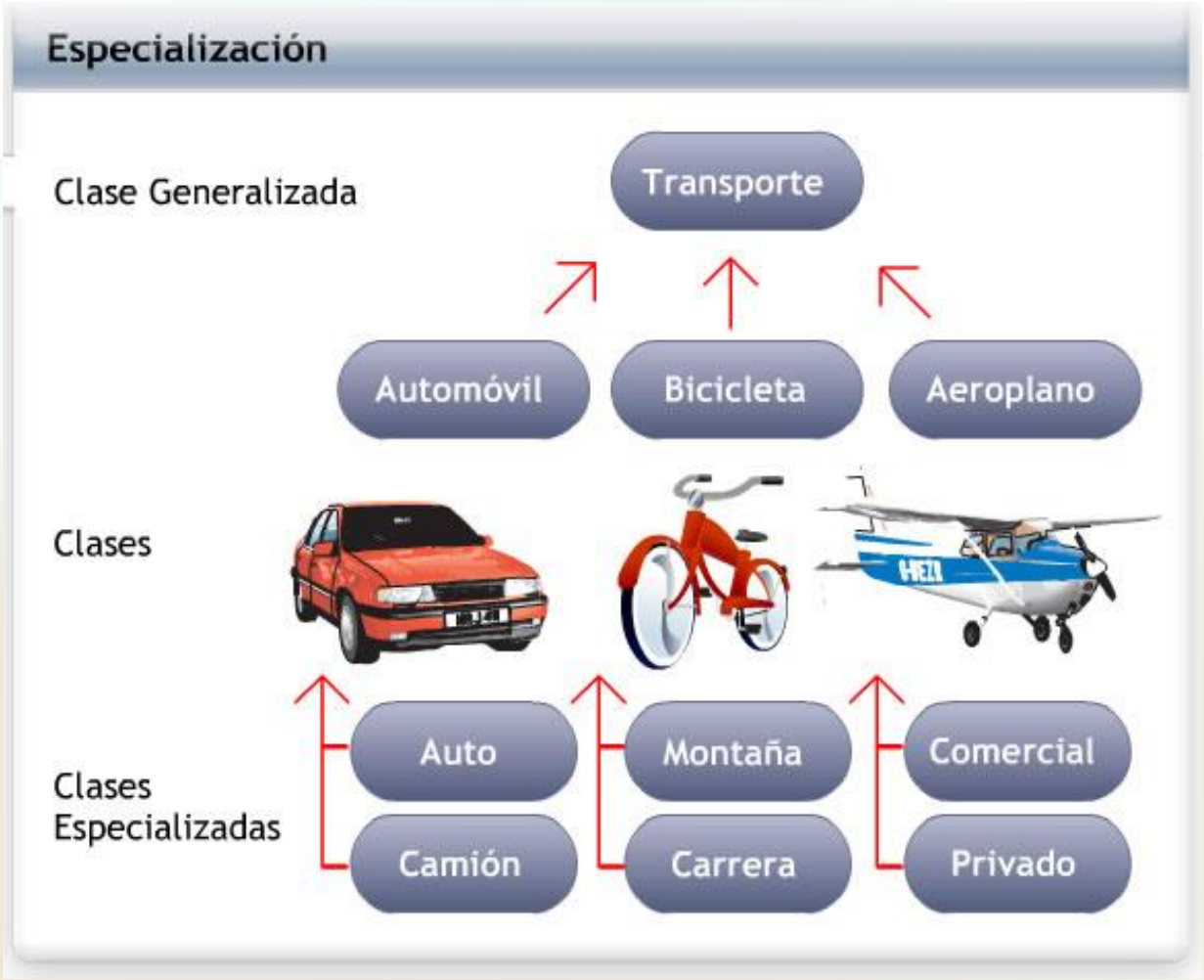
La especialización es una herencia con métodos agregados y modificados para resolver un problema específico.

Especialización

Ejemplo:

La figura muestra algunas clases con una clase generalizada de nombre Transporte. A su vez, aparecen clases más específicas, que son clases especializadas dentro de las clases Coche, Bicicleta y Avión.

Especialización - Ejemplo



Especialización

Definición:

Mientras la generalización se enfoca en las clases principales del sistema y ayuda a identificar nuevas clases, la especialización se centra en la creación de clases individuales más específicas que heredan de una clase principal.

La especialización es una herencia con el agregado y modificación de métodos para resolver problemas específicos.

Polimorfismo

El Polimorfismo permite implementar una operación heredada en una subclase, funciona solamente cuando la operación común tiene el mismo resultado semántico. La implementación de una función polimórfica depende del objeto al que se aplica. Puede utilizarse solamente con herencia.

Polimorfismo

Definición:

Una operación definida para más de una clase y que es similar en cada una de ellas se denomina polimórfica. Una operación *polimórfica* puede ser aplicada a objetos de diferentes clases para obtener el mismo resultado semántico. El mensaje "dibujar" enviado a un objeto Tarjeta genera una respuesta completamente diferente a cuando el mismo mensaje es enviado a un objeto Pintura. El resultado semántico no es el mismo, por lo tanto no se trata de polimorfismo.

El *polimorfismo* es un concepto importante en los sistemas orientados a objetos que permite la reutilización y mantenimiento de software.

Polimorfismo - Implementación

La implementación de una operación polimórfica depende del objeto al cual se aplica. Por ejemplo, las clases de pelotas Tenis y Fútbol son similares y ambas definen la operación lanzar. Por supuesto, la implementación de la operación es diferente. Una pelota de fútbol podría ser lanzada con las dos manos desde el área, mientras que una pelota de tenis podría ser lanzada por el aire con una mano para un servicio.

Polimorfismo - Implementación

Polimorfismo - Implementación

Otro Ejemplo - Discusión

Consideremos una clase general llamada InstrumentoDeEscritura, con una operación de nombre escribir. Enumeremos ejemplos de objetos que podrían ser clases especializadas (subclases que heredan) de la clase InstrumentoDeEscritura. Reflexionemos acerca de por qué es polimórfica la operación escribir.

Polimorfismo – Caso de estudio

En el sistema de ingreso de órdenes, se podría utilizar polimorfismo con la operación getName para el objeto Person. Las subclases especializadas Employee y Customer podrían heredar la operación getName. El objeto Empleado puede utilizar la definición de la operación getName tal cómo está (heredada y definida en la superclase Person). Sin embargo, para la clase Customer se podría desear utilizar un saludo junto con el nombre. En ese caso, se debería implementar la operación getName para lograr lo anterior.

Puntos clave del Polimorfismo

Es importante entender que en programación orientada a objetos, los objetos ejecutan acciones debido a los mensajes que ellos reciben. Las acciones específicas ejecutadas por un objeto dependen de la operación real utilizada. La operación podría ser una implementación especial ubicada en la subclase o una implementación ubicada en una superclase. Esto es posible debido al polimorfismo y la herencia. Los siguientes son puntos claves acerca del polimorfismo, que se deben recordar:

Da Vinci
PRIMERA ESCUELA DE ARTE MULTIMEDIAL

Puntos clave del Polimorfismo

- El polimorfismo se basa en la herencia.
- El polimorfismo se puede aplicar a cualquier operación que se hereda de una superclase.
- Las subclases que sobrescriben una operación heredada para reflejar mejor las acciones de un objeto de una subclase, están ofreciendo una forma diferente de la operación, de ahí el término polimorfismo.

Clases Abstractas

Consideremos la clase InstrumentoDeEscritura nuevamente. Es posible que la operación escribir para la clase IntrumentoDeEscritura sea tan general que resulte difícil dar una definición (crear el código) de ella. Esto es, la operación se especifica pero no se suministra ningún código. A una operación de este tipo se le denomina operación abstracta. Toda clase con al menos una operación abstracta se le considera una clase abstracta.

Clases Abstractas

Definición:

Una clase con al menos una operación abstracta es una clase abstracta. Estas clases pueden contener operaciones ordinarias (operaciones que tienen código). Toda subclase debería proveer una definición o implementación de todas las operaciones abstractas de la clase abstracta (de lo contrario dichas subclases son también abstractas). Las operaciones en una clase abstracta son expresiones que deben o pueden ser sobrescritas.

Clases Abstractas

Otro ejemplo:

Supongamos que se necesita implementar clases para un círculo, un rectángulo y un triángulo. Dichas figuras deben tener operaciones para asignar el color y el estilo del borde (sólido, punteado, etc.). Cada figura implementa estas operaciones de una manera diferente, por lo tanto se puede crear una clase abstracta Figura que tiene dos operaciones abstractas: *setColor y setBorderStyle*, para las que no se ofrece implementación alguna. Cuando Círculo hereda de la clase abstracta Figura, la clase Círculo se ve obligada a sobrescribir las operaciones *setColor y setBorderStyle* y escribir el código que implementa esas operaciones, específicamente para Círculo.

Clases Abstractas – Caso de estudio

El objeto Person en el sistema de ingreso de pedidos es una clase abstracta. Se puede instanciar un cliente o un empleado, pero no una persona.

Puntos clave de las clases abstractas

Recordar los siguientes puntos importantes relativos a clases abstractas:

- Una clase con al menos una operación abstracta (una operación para la cual no hay código definido) es una clase abstracta.
- Las subclases de una clase abstracta deberían suministrar una implementación de todas las operaciones abstractas heredadas. De no hacerlo, las subclases son también abstractas.
- Las clases abstractas se utilizan como un esqueleto para otras clases. No se pueden crear instancias de una clase abstracta.