Solucion Taller 7 ej 2

Ejercicios de análisis

a. Se define como abstracta ya que en la abstracción no tiene sentido crear una instancia de instrumento, sería demasiado vago, pero de la forma que se implementa puede crear clases distintas de instrumentos y heredarles atributos y métodos útiles.

b. public class Piano extends Instrumento
   {
   public Piano(String tipo){super(tipo)}
   public void Tocar() { System.out.println("Tocando Piano");}
   public void afinar() {System.out.println("Afinando Piano");}
   }

c. No genera errores, ya que se crea un apuntador de tipo instrumento, mas no una instancia, y este se puede usar para apuntar al objeto de saxofón o de guitarra por especificación.

d. Tocando Saxofon
   Tocando Guitarra

Ejercicios de código entregado:

a. Generaría error, ya que explotar es un método abstracto, así que no debería tener procesos adentro de la clase abstracta, solo dentro de sus subclases.

b. No es un error, una clase abstracta puede tener métodos normales, así que simplemente los heredan a sus subclases por herencia normal.

c. No, no es necesario que tenga métodos abstractos, pero sigue siendo útil en caso de que no se quieran crear instancias de esa clase.

d. Es valido ya que el método está definido también como abstracto en la clase padre, asi que tiene sentido que el apuntador pueda acceder a esa firma.

e. Es cierto, pero en la línea 28 esa misma posición se reasigna al objeto de SuperNova. Asi que imprime "Soy una Super Nova"

f. Porque estrella también es una clase abstracta, así que no hay necesidad de definir los métodos abstractos nuevamente.

g. Porque en la línea 8 de ObjTaller7 se usa este método, así que si fuera privado no podría acceder a él.

h. No la define porque ya hereda ese método de la clase estrella, y no es abstracto, así que no lo tiene que definir. Si lo definiera con la misma firma, se sobrescribiría.

i. Imprime el nombre de la clase, Galaxia, acompañado de la id del objeto. Se puede llamar normalmente ya que todos los objetos tienen el método toString por defecto, y si no se sobrescribe, la anterior es su repuesta estándar.

j. Porque se está generando solo el puntero, no la instancia de un objeto abstracto, así que es válido.

k. La B no, ya que esta intentando crear una instancia de una clase abstracta. Pero la C si, ya que solo crea el apuntador y lo asigna a un objeto existente perteneciente a una subclase de la abstracta.

l. La línea B es correcta ya que usa un apuntador de una clase padre para acceder al objeto de una clase hija, así que funciona normalmente por generalización. Sin embargo, la línea C es incorrecta ya que ese apuntador no puede acceder al método explotar de nova, ya que no lo hereda de la clase padre. Si quitamos la línea C, imprime: “Boom!” por la línea D, esta si funciona ya que por especificación se indica el tipo de objeto al cuyo método queremos acceder.

m. Porque todos los objetos son subclases de Object por defecto, así que si, para cualquier objeto esto será verdadero. En las líneas de código que se dan, imprime primero false, ya que obN apunta a null, es decir a ningún objeto, asi que no hereda de Object. Luego imprime true, ya que la concatenación de cadenas toma precedencia y se termina evaluando “null” como una cadena de texto o un string, lo cual si es un objeto.

n. No genera error, las clases abstractas pueden tener constructores, en el sentido de buen encapsulamiento, ya que cada que se cree una de sus subclases se ejecutará este constructor, siempre que tengan su constructor por defecto o se aseguren de implementar el super() en sus propios constructores.

o. Se presentan errores ya que la nueva clase no define los métodos abstractos de sus padres, tanto los de Estrella como los de ObjetoAstronomicoExtrasolar. Para corregir esto se deben implementar los métodos tipoCuerpo1(), descripción() y explotar()