Ejercicio 2 taller 7 – Respuestas Sergio Morales Martinez

1) Ejercicio de Analisis

a) Explique por qué la clase Instrumento debe definirse como abstracta y qué la diferencia de una clase normal, en el ejemplo siguiente:
R// La clase instrumento debe ser definida como abstracta, puesto que esta implementa metodos abstractos, los cuales solo pueden implementar las clases abstractas, además del diseño conceptual de las clases, lo cual hace conveniente tener esta clase abstracta como base, ya que cada instrumento implementa estos metodos de forma distinta.
Esta clase abstracta se diferencia de una clase normal ya que no debe ser instanciada, además de que implementa metodos obligatorios para las clases hijas.

b) Elabore una clase denominada Piano heredada de Instrumento, añada un constructor y redefina para ella los métodos Tocar y Afinar.
R//

```
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) ¿Señale cuál de las siguientes líneas no se pueden ejecutar y por qué?
R// El código se ejecuta sin problema, ya que a pesar de que Instrumento es una clase abstracta, se pueden crear variables de este tipo. Lo que no seria posible seria crear una instancia de esta, sin embargo eso no ocurre.

d) ¿Qué imprime el siguiente programa?
R// Imprime lo siguiente:
Tocando Saxofon
Tocando Guitarra

2. Ejercicio de código en el repositorio

a. ¿Qué sucede si se define el método explotar() de la clase Estrella como se indica a continuación? Explique su respuesta

R// Arrojaría error, pues los metodos abstractos no pueden llevar código dentro.

b. ¿Qué significa que los métodos tipoCuerpo2() y getID() de la clase ObjetoAstronomicoExtraSolar, no se definan como abstract? ¿Podría considerarse esta situación un error? Explique.

R// No arrojaría error como punto de partida. Esto podría estar diseñado de esta forma, ya que al no estar declarados como abstractos significa que se heredaran directamente sin necesidad de que las clases hijas lo tengan que escribir, esto resulta útil ya que no fuerza directamente a las subclases.

c. Si se define como abstracta la clase ObjetoAstronomicoExtraSolar, como se indica a continuación, ¿puede considerarse un error definir una clase abstracta sin métodos abstractos? Explique.
R// No es realmente un error, este cambio de diseño podría estar dado de esta forma con la intención de que no se creen instancias directas de esta clase.

d. Explique por qué el arreglo oa (línea 19) hace referencia a una clase abstracta y sin embargo, en la línea 25 se invoca el método correspondiente a cada clase derivada.
R// Esto se hace gracias a la ligadura dinámica, ya que cada posición del arreglo esta asociada a una subclase normal de la clase abstracta, las cuales implementan estos metodos, y por ligadura dinámica lo ejecutan.

e. ¿Por qué la línea 29 imprime “Soy una Super Nova” sabiendo que el arreglo oa en esa posición fue inicializado con un objeto de tipo Galaxia?
R// Lo que imprime cambio, porque en la línea 28 se hace la siguiente reasignación:
oa[0] = oa[2]; Por lo que al asignarle el valor de la tercera posición a la primera, la ligadura dinámica hace que ejecute el método del tipo de su apuntador.

f. ¿Por qué en la clase Estrella no se define el método descripcion() si la superclase lo está solicitando, ya que en este método descripción en abstracto?
R// Como la clase estrella también es una clase abstracta, esta no esta en la obligación de declarar este método abstracto, sin embargo sus subclases si.

g. ¿Qué sucede si el método tipoCuerpo1() de la clase Galaxia se define como privado? ¿Por qué se genera error?
R// Esto genera error porque un método abstracto no se puede definir como privado, su visibilidad debe de ser tal que sus subclases tengan acceso a este.

h. ¿Por qué la clase Nova no define el método tipoCuerpo1()? ¿Se podría definir? Si lo define, ¿qué interpreta de esta situación?
R// No es necesario definirlo, ya que su clase padre (Estrella) Redefinió este método de forma que deja de ser abstracto, por lo que Nova lo hereda y lo puede usar sin problema.

i. ¿Qué imprime la línea 9? ¿Por qué se puede llamar al método toString() si la clase Galaxia no lo define y su papá ObjetoAstronomicoExtraSolar tampoco?
R// Porque el método toString() Es heredado directamente de la clase padre de todas, la clase Object, por lo que a pesar de no ser definido, se puede llamar sin problemas.

j. ¿Por qué en la línea 11 se puede crear un puntero obN de tipo ObjetoAstronomicoExtraSolar si esta es una clase abstracta?
R// Esto ocurre porque en java se puede crear un apuntador de una clase abstracta, lo que no se puede es instanciarla, y siendo que el objeto real al que apunta es de tipo Nova, no hay ningún problema.

k. ¿Las siguientes instrucciones (instrucciones en las líneas B y C) son válidas? Explique
R// La línea B es invalidad puesto que al ser esta una clase abstracta, no se pueden generar instancias directas de esta. Sin embargo la C si esta correcta ya que al ser nova un objeto heredado de la clase Abstracta, este puede ser asignado a una referencia de una clase base.

l. Explique por qué (ver código a continuación) la siguiente instrucción en la línea B es correcta y la instrucción en la línea C es incorrecta. Omitiendo la instrucción en la línea C, ¿qué se imprime por pantalla? Explique su respuesta.
R// Esto no es válido porque el método explotar() no está definido en la clase ObjetoAstronomicoExtraSolar, que es el tipo de la referencia oa.
En pantalla se imprime Boom! Pues en la línea D se realiza la conversión explicita para que este método pase como de la clase Nova.

m. ¿Por qué la línea 15 imprime true? ¿Para cualquier objeto que se cree siempre imprimirá lo mismo? ¿Qué imprimen las siguientes líneas? ¿Por qué?
R// La línea 15 imprime True porque todas las clases son hijas de Object, por lo que todo objeto que no sea declarado como nulo será una instancia de Object.
La primera línea imprime false puesto que al ser este cambiado a null deja de ser una instancia. Y la segunda imprime True ya que esto pasa a ser evaluado como una String, y todas las instancias de String también son instancia de Object.

n. Agregue el siguiente constructor. ¿Se genera error? ¿Se pueden colocar constructores a clases abstractas? ¿Qué sentido tiene?
R// No genera ningún error, y es totalmente posible y valido. Las clases abstractas pueden llevar constructores, su propósito servir como constructor para instancias directas de la clase, sino para que las subclases puedan llamar a este constructor.

o. Suponga que agrega la clase EnanaBlanca como se indica a continuación. ¿Qué se puede concluir de esta nueva situación? ¿Qué errores se presentan y cómo podría corregirlos?
R// Los errores que presenta están en que no implementa los metodos abstractos que debería, asi quedaría su código arreglado:

```
public class EnanaBlanca extends Estrella {
  void agotarCombustible() {
    System.out.println("Enana blanca muere");
  }

  public void explotar() {
    System.out.println("Boom!");
  }

  public void descripcion() {
    System.out.println("Soy una enana blanca");
  }
}
```

Ademas, seria conveniente cambiar a public la visibilidad del método agotarCombustible