Facultad de Informática - Universidad Complutense de Madrid

Tecnología de la Programación I - Curso 2023/2024

Examen Extraordinario - Parte teórica

Apellidos, Nombre: _____

Instrucciones

- Duración máxima: 1 hora
- Puntuación máxima: 3 puntos
- Cuando entregues este examen teórico, se te proporcionará el enunciado del examen práctico

1. **[1 punto]** Dadas las siguientes definiciones de clases, indica la salida por pantalla o el error que impide la ejecución del código para cada uno de los apartados señalados en el método *main*, asumiendo que cada apartado se ejecuta de forma independiente al resto (quedando comentadas las líneas del resto de apartados). Por ejemplo, para realizar el apartado (a), no se ejecutan las líneas relativas a los apartados (b), (c) y (d). De la misma forma, para realizar el apartado (b) no se ejecutan las líneas relativas a los apartados (a), (c) y (d).

```java
public interface Animal {
    default void attack(Animal other) {
        System.out.println("Animal attacks other!");
    }
}

public class Elephant implements Animal {
    public void attack(Elephant other) {
        System.out.println("Two elephants fight!");
    }
}

public abstract class Feline implements Animal {
    public void attack(Feline other) {
        System.out.println("Two felines fight!");
    }
}

public class Tiger extends Feline {
    private static int count = 0;
    private int id = 0;

    public Tiger() {
        count++;
        id = count;
    }

    public Tiger(int y) {
        count += y;
        id = count;
    }

    public void attack(Animal other) {
        System.out.println("Two tigers interact: " + count);
    }
```

```
37      public void setID(int newID) {
38          id = newID;
39      }
40
41      public int getID() {
42          return id;
43      }
44  }
45
46  public class BengalaTiger extends Tiger {
47      public BengalaTiger() {
48          super(10);
49      }
50
51      public void attack(BengalaTiger other) {
52          System.out.println("Bengala tiger...");
53      }
54  }
55
56  public class AnimalExample {
57      public static void main(String[] args) {
58          // Apartado (a)
59          Animal a = new Feline();
60          a.attack(new Tiger());
61
62          // Apartado (b)
63          Feline f = new Tiger();
64          Tiger t = new Tiger();
65          f.attack(t);
66
67          // Apartado (c)
68          Tiger t = new BengalaTiger();
69          t.attack(new BengalaTiger());
70
71          // Apartado (d)
72          Tiger t1 = new Tiger();
73          Tiger t2 = new BengalaTiger();
74          t1.attack(t1);
75          t1.attack(t2);
76          t2.attack(t1);
77          t2.attack(t2);
78          System.out.println("Individual ID of t2 = " + t2.getID());
79      }
80  }
```

2. **[1 punto]** Muestra la salida que se imprime al ejecutar el siguiente código. Cada vez que se lance una excepción, indica la causa y la línea en la que se produce.

```
1  public class Excepcionando {
2
3      public static void main(String[] args) {
4          int i = 1, j = 2;
5          int guardo[] = new int[i + j];
6
7          try {
8              for (i = 0; i < 2; i++)
9                  try {
10                     for (j = 2; j >= 0; j--) {
11                         m_1(i, j);
12                         guardo[i + j] = i + j;
13                     }
14                 } catch (MiExcepcion e) {
15                     System.out.println("Mi excepción: " + e.getMessage());
16                 }
17         } catch (Exception e) {
18             System.out.println("Capturada excepción en main: "+ e.getMessage());
19         } finally {
20             System.out.println("Finalizando ejecución !");
21         }
22
23         int cinco = Integer.parseInt("cinco");
24         System.out.println("Código terminación: " + cinco);
25     }
26
27     public static void m_1(int a, int b) throws MiExcepcion {
28         try {
29             m_2(a, b);
30         }
31         catch (MiExcepcion me) {
32             System.out.println ("Capturada mi excepción en m_1");
33         }
34         catch (Exception e) {
35             System.out.println ("Capturada otra excepción: " + e.getMessage());
36             throw new MiExcepcion ("Error en m_2 !");
37         }
38
39         System.out.println("Finalizado m_1: " + a + ", " + b );
40     }
41
42     public static void m_2(int x, int y) throws Exception {
43         if (x < y)
44             System.out.println (x +  " < " + y);
45         else
46             System.out.println (x + " / " + y + " = " + x/y );
47     }
48 }
49
50 public class MiExcepcion extends Exception {
51
52     public MiExcepcion() {
53         super();
54     }
55
56     public MiExcepcion (String message) {
57         super(message);
58     }
59 }
```

3. **[1 punto]** Considerando el código que aparece a continuación:

   a) Explica y corrige los errores de compilación.

   b) Una vez arreglados los errores, muestra la salida que se imprime al ejecutar el código.

```java
public interface Coleccionable {
   public double valorMercado();
}

public abstract class Item {
   private String titulo;
   private String autor;
   private Integer publicacion;

   public Item (String titulo, String autor, Integer publicacion) {
      this.titulo = titulo;
      this.autor = autor;
      this.publicacion = publicacion;
   }

   public String toString() {
      return titulo + " de " + autor + " (" + publicacion + ")";
   }
}

public abstract class Vinilo extends Item  implements Coleccionable {
   protected double precio;
   protected double ratio = 2;

   public Vinilo (String titulo, String autor, Integer publicacion, double
    precio) {
      this.precio = precio;
      super(titulo, autor, publicacion);
   }

   public String toString() {
      return super.toString() + " está valorado en " + valorMercado() + " euros"
    ;
   }
}

public class LP extends Vinilo {
   public LP (String titulo, String autor, Integer publicacion, double precio) {
      super(titulo, autor, publicacion, precio);
      ratio = 5;
   }

   public double valorMercado() {
      return precio * ratio;
   }
}

public class Sencillo extends Vinilo {
   private String cancion;

   public Sencillo (String titulo, String autor, Integer publicacion, double
    precio, String cancion) {
      super(titulo, autor, publicacion, precio);
      this.cancion = cancion;
   }
```

```
55      public double valorMercado() {
56          return precio * ratio;
57      }
58
59      public String toString() {
60          return titulo + " contiene la canción " + cancion;
61      }
62  }

64  public class CD extends Item {
65      public CD (String titulo, String autor, Integer publicacion) {
66          super(titulo, autor, publicacion);
67      }
68  }

70  public class Streaming extends Item {
71      private String plataforma;
72
73      public Streaming (String titulo, String autor, Integer publicacion, String
        plataforma) {
74          super(titulo, autor, publicacion);
75          this.plataforma = plataforma;
76      }
77
78      public String toString() {
79          return super.toString() + " está en la plataforma " + plataforma + ". ";
80      }
81  }

83  public class ColeccionMusica {
84      public static void main(String[] args) {

86          Item[] lista = new Item[5];
87
88          lista[0] = new LP("The Wall","Pink Floyd", 1979, 30);
89          lista[1] = new Sencillo("Revolution","The Beatles", 1968, 15, "Hey Jude");
90          lista[2] = new CD("Nevermind","Nirvana", 1991);
91          lista[3] = new Streaming("As It Was","Harry Styles", 2022, "Spotify");
92          lista[4] = new Item("American Idiot", "Green Day", 2004);
93
94          for (int i = 0; i < lista.length; i++) {
95              System.out.println(lista[i]);
96          }
97      }
98  }
```