**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

**FACULTAD DE INGENIERIA EN ELECTRICIDAD Y COMPUTACION**

**PROGRAMACIÓN ORIENTADA A OBJETOS**

**SEGUNDA EVALUACIÓN**

**I TÉRMINO 2017**

1. **Seleccione Verdadero (V) o Falso (F) según corresponda (5 puntos)**

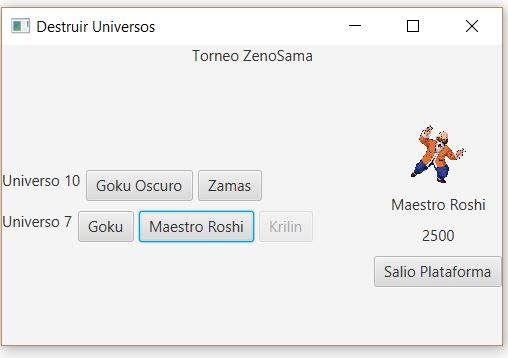
|  |  |  |
| --- | --- | --- |
| El layout BorderPane establece sus nodos de contenido en la parte superior, inferior, derecha, izquierda o región central. | V | F |
| El método sleep() de un Thread puede lanzar una excepción de tipo NullPointerException | V | F |
| Una clase interna es capaz de ver todos los atributos públicos de su clase contenedora, pero la clase contenedora desconoce de la existencia de los atributos de la clase interna. | V | F |
| Las colecciones pueden contener tipos de datos primitivos | V | F |
| Cuando una excepción es lanzada dentro de un bloque try, todas las sentencias dentro del bloque try son siempres ejecutadas | V | F |

**2. Responda brevemente las siguientes preguntas**

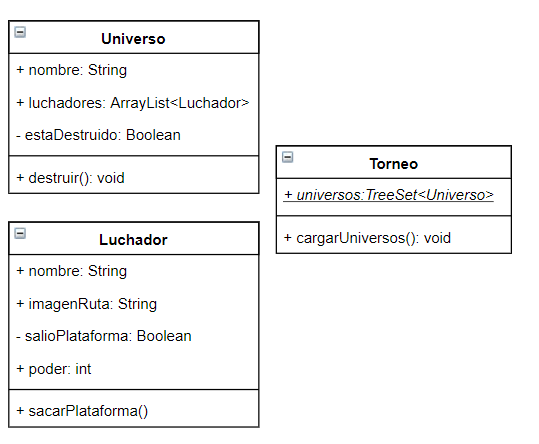
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **2.1 ¿Qué es Serialización e indique cuál es la interfaz que Java obliga a implementar para serializar (Nombre de la interfaz y métodos) (2 puntos)**   |  | | --- | |  |   **2.2 ¿Por qué es recomendable usar Buffered Streams en lugar de File Streams para leer archivos? (1 punto)**   |  | | --- | |  |   **2.3 ¿Cuál es la interfaz que debe ser implementada para manejar un evento en JavaFX y cuál es el método que debe ser sobreescrito? (1 punto)**   |  | | --- | |  |   **2.4 ¿Qué es la programación multithreading? (1 punto)**   |  | | --- | |  |   **3. Analice el código y seleccione la respuesta correcta**  **3.1 ¿Cuál es la salida del código anterior (Justifique su respuesta) (5 PUNTOS)**  import java.util.HashSet;  class Student{  public Student(int r) {  rollNo = r;  }  int rollNo;  }  class Test {  public static void main(String[] args){  HashSet<Student> students = new HashSet<>();  students.add(new Student(5));  Student s10 = new Student(10);  students.add(s10);  System.out.println(students.contains(new Student(10)));  System.out.println(students.contains(s10));  }  }   1. false   Justificación  true   1. false   false   1. true   false   1. true   true  **3.2 ¿Cuál es la salida de ejecutar el siguiente código? (Justifique su respuesta) (5 PUNTOS)**   |  | | --- | | class MyThread extends Thread {  MyThread() {  System.out.print(" MyThread");  }  public void run() {  System.out.print(" bar");  }  public void run(String s) {  System.out.print(" baz");  }  } public class TestThreads {  public static void main (String [] args) {  Thread t = new MyThread();  t.start();  }  } |  1. My Thread bar 2. My Thread baz 3. bar 4. baz   Justificación:  **3.3 ¿Cuál sería la salida del siguiente programa? (Justifique su respuesta) (5 PUNTOS)**  public class RTExcept{  public static void throwit (){  System.out.print("throwit ");  throw new RuntimeException();  }  public static void main(String [] args){  try{  System.out.print("hello ");  throwit();  }catch (Exception re ){  System.out.print("caught ");  }finally{  System.out.print("finally ");  }  System.out.println("after ");  }  }   1. hello throwit caught 2. Compilation fails 3. hello throwit RuntimeException caught after 4. hello throwit caught finally after   Justificación:  **3.3 Analice el siguiente código (5 PUNTOS)**   |  | | --- | | 1. import java.util.\*;  2. public class Conjuntos {  3. public static void main(String[] args) {  4. Set<Integer> s = new TreeSet<Integer>();  6. s.add(23); s.add(42); s.add(new Integer(5));  7. Iterator i = s.iterator();  8. //while(System.out.print(i.next())) { }  9. //for(Integer i2: i) System.out.print(i2);  10. //for(Integer i3: s) System.out.print(i3);  11. //while(i.hasNext()) System.out.print(i.get());  12. //while(i.hasNext()) System.out.print(i.next());  13. }  14. } |   **Dado el código anterior cuál de los siguientes enunciados es verdadero (Escoja todos los que apliquen) (Justifique su respuesta):**  A. Si solo la Línea 8 se descomenta el código compilara.  B. Si solo la Línea 9 se descomenta el código compilara..  C. Si solo la Línea 10 se descomenta el código compilara..  D. Si solo la Línea 11 se descomenta el código compilara..  E. Si solo la Línea 12 se descomenta el código compilara..  F. De esas líneas que compilan la salida será: 23425  G. De esas líneas que compilan la salida será: 52342  Justificación: |

**4. Tema de Desarrollo (50 PUNTOS)**

4.1 Grafique el Scene Graph de la pantalla que se muestra a continuación (gráfico jerárquico de nodos de la escena) **(5 puntos)**



Dado el siguiente diagrama de Clases:

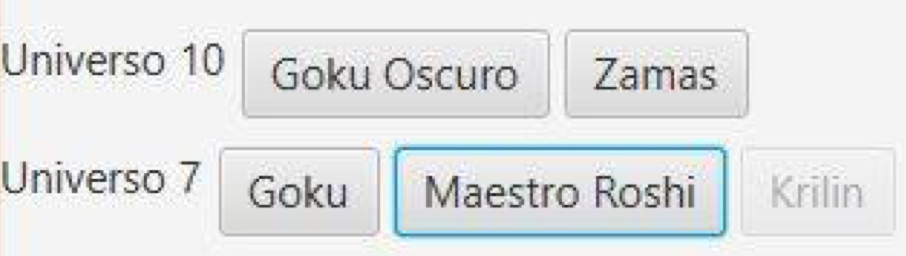


4.2 Considere que los datos de la clase Universo están serializados en un archivo llamado “torneo\_de\_la\_fuerza.dat”. Implemente el método ***cargarUniversos*** de la clase Torneo, con el fin de llenar el atributo “universos” de tipo **TreeSet** con los datos del archivo **(5 puntos)**.

4.3 Cree la clase **Universo** para que funcione de acuerdo a las especificaciones mostradas. (El método destruir() sólo cambia el estado de la variable de instancia **estaDestruido** a true) **(10 puntos)**.

4.4 La clase ***PaneZenoSama***, contiene los elementos necesarios para la creación de la interfaz. Implemente únicamente los métodos descritos a continuación y declare las variables de instancias de la clase.

***mostrarUniversos():*** Muestra en la ventana principal la información de los universos **que no han sido destruidos** con sus luchadores. **Si el luchador fue sacado de la plataforma, el botón se presenta deshabilitado**. (Su código debe crear el contenedor y los elementos de esa sección únicamente). **(10 puntos).**

****

Al dar clic en el botón de un Luchador se llamará al método clicLuchador, correspondiente al luchador

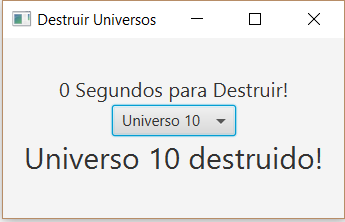
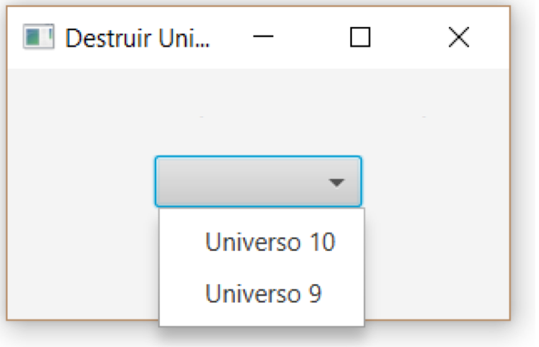
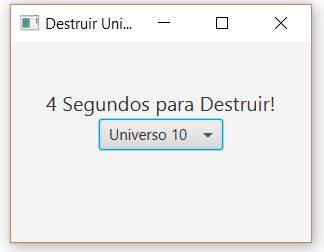
***clicLuchador:*** Este método presentará en la ventana principal la información de ese luchador (nombre, poder) y su imagen. (No tiene que implementar la funcionalidad del botón salir plataforma, esta se la implementará mas adelente). (Su código debe crear el contenedor y los elementos de esta sección únicamente) **(5 puntos)**.



***clicSalirPlataforma:*** Al dar clic en el botón “Salir Plataforma” se llama a este método y el luchador cambia de estado y el botón correspondiente al jugador se inactiva*.* **(5 puntos)**

4.5 Crear una clase principal que inicie la ejecución de la aplicación. La clase debe cargar la información de los universos en la clase Torneo y mostrar la pantalla TorneoZenoSama. **(10 puntos)**

4.6 Cree la siguiente pantalla que presenta solo aquellos universos, en los cuales todos los luchadores han salido de la plataforma. Después de 10 segundos el universo seleccionado se destruye. (Es un nuevo Stage. Use Threads. Trabaje con las clases definidas anteriormente, es libre de modificar las clases anteriormente definidas). **(20 puntos)**

****