**EXÁMEN SOBRE CONOCIMIENTOS DE JAVA &**

**ESTÁNDARES DE INTEGRACIÓN EAI**

* **NOMBRE:**
* **ÁREA:** INTEGRACIÓN

*“La presente prueba ha sido desarrollada para validar los conocimientos existentes y requeridos de integración en lo relacionado a lenguaje JAVA, estándares y herramientas EAI (No OSB)”.*

**PREGUNTA #3: Grafique utilizando conocimientos de BPMN, como diseñaría en Bizagi el escenario siguiente:  *“Un flujo WS síncrono que consuma 1 BD (Procedure), 1 WS (Proxy) y persista en 1 JMS queue”.* Considerar las posibles excepciones personalizadas estándar para controlar el posible error en cada interface de consumida. (1 punto)**

**PREGUNTA #4: Para el manejo de Caching en los desarrollos EAI v10g, que librerías son requeridas ingresar en las aplicaciones. Marcar la respuesta correcta: (1 punto)**

1. *ehcache-core.jar, slf4j-api.jar, slf4j-log4j12.jar*
2. *cache-core.jar, slf4j-api.jar, slf4j-log4j12.jar*
3. *ehcache-core.jar, log4j.jar, commons-logging.jar*
4. *cache-core.jar, log4j.jar, commons-logging.jar*
5. *Ninguna de las anteriores.*

**PREGUNTA #5: Explique en qué casos se recomienda el manejo de Caching en las aplicaciones EAI (WebService): (1 punto)**

**PREGUNTA #6: Para el manejo de Logging cuáles son los NIVELES estándar usados para en nuevos desarrollos EAI v11g (1 punto)**

1. *INFO, ERROR, DEBUG, WARN*
2. *INFO, ERROR, DEBUG*
3. *INFO, DEBUG, WARN*
4. *INFO, ERROR*
5. *Ninguna de las anteriores.*

**PREGUNTA #7: Para desarrollos v11g que propiedad de Spring es utilizada para el mapping de los archivos .properties: (1 punto)**

1. *context:annotation-config.*
2. *jaxwsCustomProperties*
3. *context:component-scan*
4. *context:property-placeholder*
5. *Ninguna de las anteriores.*

**PREGUNTA #8: Para el manejo de objetos JMS dentro de Weblogic, en los desarrollos de componente EAI, seleccionar cuales son dichos objetos que son requeridos de su creación:   
 (1 punto)**

1. *Servidor JMS, Almacén Persistente, Modulo JMS, Fábrica de Conexiones, Cola.*
2. *Almacén Persistente, Modulo JMS, Fábrica de Conexiones, Cola Distribuida Uniforme.*
3. *Servidor JMS, Almacén Persistente, Modulo JMS, Despliegue Secundario, Fábrica de Conexiones, Cola Distribuida Uniforme.*
4. *Servidor JMS, Modulo JMS, Fábrica de Conexiones, Cola*
5. *Ninguna de las anteriores.*

**PREGUNTA #9: Para la creación de ORACLE Driver en Weblogic, cuales son los Drivers utilizados tanto para DataSource normales y de tipo XA: (1 punto)**

1. *Oracle’s Driver (Thin) for Service connections, Oracle’s Driver (Thin XA) for Service connections.*
2. *Oracle’s Driver (Thin) for Instance connections, Oracle’s Driver (Thin XA) for Instance connections.*
3. *Oracle’s Driver (Thin) for RAC Service-Instance connections, Oracle’s Driver (Thin XA) for RAC Service-Instance connections.*
4. *Ninguna de las anteriores.*

**PREGUNTA #10: Java al serializar un objeto automáticamente serializa todos los contenidos internos del objeto. Para evitar una serialización específica de un contenido se debe de usar:   
 (1 punto)**

1. *final.*
2. *transient.*
3. *static.*
4. *Ninguna de las anteriores.*

**PREGUNTA #11: El método “*Equals*” en *StringBuffer* y *StringBuilder* no funciona para comparación de cadenas de forma directa (VERDADERO o FALSO). ¿Porque? (1 punto)**

**PREGUNTA #12: *Polimorfismo de Generics*. Mencionar cual está correctamente aplicada y cual no. Más de una pueden ser correctas. (1 punto)**

1. *Object[] miLista01 = new JButon[10].*
2. *Persona[] miLista02= new Empleado[10].*
3. *List<Object> miLista03 = new ArrayList<JButton>().*
4. *List<Persona> miLista04 = new ArrayList<Empleado>().*
5. *Todas.*

**PREGUNTA #13: *Mencionar cual es la diferencia en la comparación usando: == y equals*. En qué casos se utiliza cada uno de ellos. (1 punto)**

**PREGUNTA #14: *Ingresar VERDADERO o FALSO*, para las siguientes afirmaciones relacionadas a INTERFACES y CLASES ABSTRACTAS: (1 punto)**

1. *No todos los métodos de una clase abstracta tienen que ser abstractos.*
2. *Las clases abstractas no pueden tener métodos de tipo private ni static.*
3. *Las clases abstractas permiten declaración de variables.*
4. *Las interfaces no permiten declaración de variables.*
5. *Las interfaces no permiten la declaración de constructores.*
6. *Las interfaces pueden extender a otras interfaces.*

**PREGUNTA #15: *Orden de carga en clases*. Al momento de ejecutarse una clase siempre hay un orden de ejecución. Supongamos que se tiene 2 clases una que hereda de la otra (Animal, Perro) y que poseen constructores, bloques normales, estáticos, etc. Seleccionar cual es el orden de ejecución correcto al ser ejecutada la clase. (1 punto)**

1. *Bloques Estáticos, Bloques Normales, Constructores.*
2. *Constructores, Bloques Estáticos, Bloques Normales.*
3. *Bloques Normales, Constructores, Bloques Estáticos.*
4. *Bloques Normales, Bloques Estáticos, Constructores.*

**PREGUNTA #16: Control de Acceso. *Mencionar la diferencia en la visibilidad de datos en el manejo de los Modificadores de tipo: PRIVATE, PUBLIC, PROTECTED:* (1 punto)**

**PREGUNTA #17: *Ingresar VERDADERO o FALSO*, para las siguientes afirmaciones relacionadas a métodos OVERLOADING y OVERRIDING: (1 punto)**

1. *Los métodos de tipo Override no pueden retornar un argument.*
2. *Los métodos de tipo Override no pueden ser de visibles como private.*
3. *Los métodos de tipo Overload pueden retornar un argument.*
4. *Los métodos de tipo Overload pueden cambiar la forma de método en lo relacionar a arguments.*

**PREGUNTA #18: *Mencione la diferencia entre WRAP y UNWRAP (terminología):* (1 punto)**

**PREGUNTA #19: Pseudocodigo. Escribir un algoritmo que cumpla con el siguiente escenario:**

***“Se tiene como input una trama de tipo cadena conformado por N dígitos del 1-9 (123456789) y lo que se quiere es obtener como output 2 cadenas: la primera cadena que obtengan los números pares y la segunda cadena que obtengan los números impares, luego imprimir cada una en consola”.* No usar las funciones de *java.lang.string*:  
(1 punto)**

**PREGUNTA #20: Cual es el resultado de ejecutar el siguiente trozo de código: (1 punto)**

***int numero=10;  
numero\*=2+5;***