MEDRAR SOLUCIONES DE SOFTWARE EMPRESARIAL

CARGO A DESEMPEÑAR: Desarrollador JAVA
FECHA DE INICIACIÓN: Julio 31 de 2019.
CIUDAD DE CONTRATACIÓN: Armenia, Quindío.
TIEMPO PARA EJECUCIÓN: (12) Horas.

PRUEBA TÉCNICA: DESARROLLO DE SOFTWARE JAVA

1. Preguntas conceptuales OOP

1. ¿Qué significa "alta cohesión bajo acoplamiento" en OOP?

R// **Cada módulo debe ser coherente con su funcionalidad y debe hacer únicamente lo que le corresponde, no tener más de una responsabilidad. y debe estar lo más desacoplado posible, es decir depender lo mínimo posible de otros módulos.**

2. ¿En qué caso favorece el uso de la herencia sobre la composición en OOP?

R// Cuando se requiere definir un comportamiento para las clases asociadas.

3. ¿Por qué en OOP la encapsulación es importante?

R// **Me permite proteger los atributos y datos de un objeto de accesos no permitidos.**

4. ¿Qué implicaciones trae en el diseño de una aplicación que respete el principio "cerrado a la modificación abierto a la extensión"?

R// Me permite crecer la aplicación sin arriesgar lo que ya está actualmente construido, debe poder permitir nuevas funcionalidades garantizando que lo que ya está no va ser modificado.

5. ¿Qué significa la palabra clave static, y dónde puede ser usada?

R// La palabra static me carga en memoria el atributo antes de ejecutar el método constructor de la clase, y me permite acceder a él sin necesidad de instanciar la misma.

6. ¿Qué es polimorfismo? Descríbalo con un ejemplo

R// Es la posibilidad de cambiar el comportamiento de un objeto en tiempo de ejecución, un ejemplo es el de las figuras,

Clase Figura ; Clase Triangulo extends Figura ;Clase Cuadrado extends Figura.

Al tener un objeto de tipo Figura, podré transformarlo en cualquier momento a los diferentes tipos de figura (triangulo, cuadrado) y asi los métodos como area(); se comportan de manera diferente en cada llamado.

7 ¿Qué es el garbage collector?

R// Es el encargado de liberar los objetos que ya no están en uso y no se utilizaran en el futuro, limpiando la memoria.

8 ¿Qué hace la palabra clave synchronized?

R// Con esta palabra se indica que un recurso debe ser accedido por un subproceso a la vez, es utilizado mucho cuando se programa con hilos.

9 ¿Cuándo y por qué son los getters y setters importantes?

R// Son los métodos encargados de acceder y/o modificar el valor de mis atributos, son importantes ya que son pieza fundamental del encapsulamiento.

10 ¿Cuáles son las diferencias entre interfaces, clases abstractas, clases e instancias?

R// Interface :

- Debe tener todos sus métodos abstractos, es decir declarados, y su implementación se deberá realizar en las clases que la implementen.
- Todas sus variables deben ser final.
- Una interface no puede ser instanciada.
- Una clase podrá implementar cuantas interfaces requiera.

Clase Abstracta:

- Debe tener al menos un método abstracto.
- No puede ser instanciada.
- Una clase solo podrá extender de una sola clase abstracta.

Clases:

- Es la representación abstracta de un objeto de la vida real, y está compuesta por atributos y métodos los cuales definen las características y el comportamiento del objeto.

Instancias:

- Es la creación de un objeto en memoria

11 ¿Qué es sobrecarga de métodos?

R// Es tener dos o mas métodos con el mismo nombre pero con diferentes parametros.

12 ¿Cómo se maneja o controla una excepción?

R// Se controla con try Catch en donde se debe manejar por orden jerárquico.

2. Preguntas de algoritmos y estructuras de datos

1 ¿Qué diferencias existen entre las estructuras de datos: lista enlazada (linked list), un arreglo y un tabla de hash (hastable/hashmap)?

R//

LinkedList : Es una lista doblemente enlazada en donde cada posición apunta al ítem posterior y al ítem anterior, es más costosa al momento de obtener un item especifico ya que se debe recorrer toda la lista hasta ubicarlo.

Arreglo: Es una estructura de almacenamiento fijo,es menos costosa para obtener un ítem ya que se puede apuntar directamente a la posición.

Tabla Hash: Me permite almacenar información mediante llave valor, no puedo tener items con llaves repetidas, y utiliza los métodos hashCode e iquals() para ordenar y obtener los items del mapa, es por esto que no se pueden utilizar variables primitivas.

2 ¿Bajo qué criterio una función recursiva es más costosa que una función iterativa?

R//

3 ¿Con qué estructura de datos modelaría el sistema de transporte Transmilenio para realizar cálculos sobre las rutas y tiempos totales de cada ruta y encontrar la ruta más rápida de estación a estación?

R// Grafos. Ya que que permite recorrer sin impedimento los diferentes caminos hasta encontrar el más óptimo apoyándome en el peso entre los nodos.

4. ¿Con qué tipo de algoritmo se enfrentaría al problema de distribución de paquetes por peso en camiones de una empresa de mensajería para que se realizara de manera óptima?

R//

3. Ingeniería de software

1. Liste las etapas básicas de la construcción de software, independientes a la metodología de desarrollo y defina qué es calidad de software.

R// Etapas de Construcción de Software:

- Análisis
- Especificación
- Diseño y Arquitectura
- Implementación
- Pruebas

Calidad de software: Es el proceso mediante el cual se garantiza que la aplicación cumpla con los criterios, estándares y buenas prácticas que me permitan garantizar un producto de calidad.

2. Describa los escenarios de prueba que le definiría al proceso de hervir un huevo

R// **Variables**: olla, Agua, Fuego,Huevo y Estufa.

Escenario 1: En caso que falte alguna variable. Salida -> Fallo.

Escenario 2: Agua en la olla, introducir el huevo en la olla con agua , prender la estufa, poner el fuego demasiado alto, lo cual evaporara el agua antes que este cocido el huevo. Salida: Fallo.

Escenario 3: Agua en la olla, introducir el huevo en la olla con agua , prender la estufa, poner el fuego demasiado alto lo cual hará que el hueva se reviente. Salida:Fallo.

Escenario 4: Agua en la olla, introducir el huevo en la olla con agua ,la estufa falla y no prende por ende no es posible, poner el fuego. Salida: Exito.

Escenario 5: Agua en la olla, introducir el huevo en la olla con agua , prender la estufa, poner el fuego a término lento, esperar 15 minutos. Salida: Exito.

3 ¿Cuáles la **diferencia** entre concurrencia, disponibilidad y consistencia?

R// Concurrencia: Capacidad de controlar multiples peticiones al mismo recurso en el mismo momento.
Disponibilidad: Capacidad de dar respuesta en cualquier momento a cada petición.
Consistencia: Datos guardados, coherentes y consistentes a los enviados.

4 Mencione algunos patrones de diseño: ejemplo Singleton

R// Singleton, Facade, DTO, Inversión de Control, Decoretor, Proxy

5 Explique la inyección de dependencia y su relación con la inversión de control.

R// La inyección de dependencias es la implementación del patrón de diseño inversión de control. En donde me permite un bajo acoplamiento haciendo que las clases no dependan de una implementación en concreto.

6 ¿Cuáles la diferencia entre un bloqueo optimista y un bloqueo pesimista?

R//

4. Problemas prácticos

1. Resuelva de manera recursiva los siguientes ejercicios:
   - Encontrar el número mayor en un arreglo de 12 posiciones llenado de manera aleatoria
   - Sumar los dígitos de un número de 6 caracteres dado de forma aleatoria
   - Encontrar el factorial de un número entre 1 y 10 dado de manera aleatoria

NOTA: para estos 3 ejercicios se debe imprimir en consola los datos de entrada y salida.

2. Se pide realizar un desarrollo para un cine (solo de una sala) la cual tiene un conjunto de asientos (8 filas por 9 columnas).

   Del cine nos interesa conocer la película que se está reproduciendo, cuantas sillas están ocupadas, cuántas disponibles y el precio de la entrada en el cine.

   De las películas nos interesa saber el título, duración, edad mínima y director.

   Del espectador, nos interesa saber su nombre, edad y el dinero que tiene.

   Los asientos son etiquetados por una letra (columna) y un número (fila), la fila 1 empieza al final de la matriz como se muestra en la tabla. También deberemos saber si está ocupado o no el asiento.

8 A 8 B 8 C 8 D 8 E 8 F 8 G 8 H 8 I
7 A 7 B 7 C 7 D 7 E 7 F 7 G 7 H 7 I
6 A 6 B 6 C 6 D 6 E 6 F 6 G 6 H 6 I
5 A 5 B 5 C 5 D 5 E 5 F 5 G 5 H 5 I
4 A 4 B 4 C 4 D 4 E 4 F 4 G 4 H 4 I
3 A 3 B 3 C 3 D 3 E 3 F 3 G 3 H 3 I

2 A 2 B 2 C 2 D 2 E 2 F 2 G 2 H 2 I
1 A 1 B 1 C 1 D 1 E 1 F 1 G 1 H 1 I

Se debe realizar una pequeña simulación, en el que se generaran muchos espectadores (más que la disponibilidad de la sala) y se sentaran aleatoriamente (verificar si la silla está ocupada o no).

Solo se podrá sentar si tienen el suficiente dinero, hay espacio libre y tiene edad para ver la película.

Los datos del espectador y la película pueden ser totalmente aleatorios.

NOTA: Se deben imprimir de manera ordenada los datos de todos los espectadores y películas generadas antes de pasar al proceso de selección de película (aleatoria de las generadas) y a sentar y validar los espectadores, esto al final debe ser impreso adicional a la información solicitada inicialmente.

Para las pruebas prácticas se debe entrar los proyectos y/o clases java para su ejecución y pruebas.

Para enviar los entregables de la prueba se debe tener o crear un repositorio en git público en el cual se deben subir el documento resuelto y los proyectos java para ser revisados y ejecutados. Una vez tengan este proyecto en el repositorio, se debe enviar la URL del repositorio al correo talentos@medrarsof.com, para poder clonarlo y hacer su revisión.

- La tecnología para realizar el aplicativo podrá ser elegida por el aspirante.