1. Desde el punto de vista del diseño de software, describa qué propósito tiene implementar FIFO y LIFO en estructuras de datos.

Las colas (FIFO) tienen como principal propósito el implementar prioridad a los elementos que se desean acceder, dando énfasis a el primero que entro a esa estructura, por otra parte, en las pilas (LIFO) esta prioridad radica en el último elemento que entro a la estructura. Por ejemplo, tal vez se desea que los datos respeten el orden que llegaron para ser leídos (usaremos una cola), o, se requieren que los datos leídos por un método recursivo sean usados según el último que salió.

2. Explique, ¿Qué utilidad o beneficio se obtiene al definir estructuras de datos genéricas?

Si se designa una estructura con un dato genérico, esta puede ser utilizada para información diversa y por ende, poder ser usada en más situaciones.

3. Explique, ¿Cuál es el propósito de usar abstracciones en la programación orientada a objetos?

Abstracción es encontrar patrones comunes entre elementos que, en este caso, interactuarán entre ellos. Al abstraerlos se podrá crear esas relaciones en sus estados más simples, atacar un problema complejo con elementos sencillos y ver si la solución planteada es plausible.7

4. Explique, ¿Qué diferencia conceptual existe entre los conceptos Clase Abstracta e Interface?

El objetivo de una interfaz es de dar las principales funciones de una clase que se requiere que haga ciertas cosas. Mientras que una Clase abstracta se define como la Generalización máxima que como clase, no será usada por si misma, mas da la base a sus clases hijas. Aparte, en su ejecución en Java, las clases solo poseer una clase abstracta como “Clase padre”, contrapuesto a que pueden implementar muchas interfaces simultaneas.

5. Explique, ¿De qué manera es posible instanciar una Interface?

Siendo estrictos en la definición de instancia, no se pueden instanciar una interfaz. Lo que se puede hacer es una realización de la interfaz, la cual estaremos instanciando alguna clase que la implemente.

6. Explique, ¿A qué refiere el concepto “Diseño de Software”?

Se entiende por diseño, al conjunto de decisiones que se toman en pro de la evasión de errores, idealización de la solución, efectiva al problema planteado, y en general a como debe ser y funcionar el software que se quiere crear.

7. Explique, ¿A qué refiere el concepto “Refactorización”?

Refactorización, palabra que se compone del prefijo re, que se atribuye realizar otra vez y la palabra factorización, la cual, entendida de fábrica, es la creación de algo. En este contexto, refactorización, hace referencia a la fase final del proceso de desarrollo de software, el cual se encarga de mejorar el código escrito, para hacerlo más eficaz, agregar nuevas funcionalidades, eliminar errores, entre otros.

8. Explique, ¿a qué refiere los conceptos “Patrón y Antipatrón de Diseño”?

Llamamos patrón a un conjunto de características que se repite, en este caso en el proceso de diseño, las cuales se consideran buenas practicas y de gran recomendación su seguimiento. Por otro lado, antipatrón, hace alusión a los errores comunes realizados al ejecutar la fase de diseño de software, los cual, por obvias razón (morosidad en la terminación del proyecto, falta de eficiencia de código, errores denominados “bugs”, entre otros) deben ser evitados.

9. Explique, ¿Qué elementos de diseño de software justifican el interés por implementar “Java Collection Framework”?

El uso de esta forma de organizar aquellas clases encargadas de guardar grupos de datos, es la de tener los métodos mínimos y recomendados para que este tipo de clase funcionen correctamente, reducir el esfuerzo al programar, hace que el código se compatible con otras bibliotecas, entre otros.

10. Describa, ¿Cuáles son los elementos centrales que conforman “Java Collection Framework”?

Los principales elementos son subinterfaces (List, Queue,Set), clases (ArrayList, LinkedList, Stack, HashSet, LinkedHashSet,) y métodos (add, remove, size, clear, contains, equals) enfocados al almacenamiento y manipulación de conjuntos de datos.

11. Explique conceptualmente, ¿A qué refiere Iterator?

Un iterador es un objeto que permite recorrer una lista de objetos uno por uno.

12. Explique, ¿Qué rol y propósito tiene la utilización de un hashcode en una estructura de datos?

Un hashcode es una especie de llave, la cual tiene como propósito hacer la búsqueda de una objeto con esa llave, mucho más rápida y con coste muy bajo, ya que entrega con mucha precisión la ubicación en que debería estar, si es que esta.

13. Explique y describa, ¿qué mecanismo(s) permite resolver posibles colisiones en la agregación de datos en un HashSet?

Lo primero es que es una estructura de set, que es un conjunto matemático que no admite elemento repetidos, por ello se logran evitar colisiones con datos idénticos. Segundo, si dos datos distintos, comparten hashcode, estos serán almacenados en la misma y para no sobreponerlos se utiliza un proceso llamado Separate Chaning, el cual consisten en crear una estructura, mayormente lineal, que guarde ambos objetos uno detrás de otro en el mismo espacio. Tercero, para evitar que el proceso anterior dificulte la eficiencia de la búsqueda, existe un coeficiente limite que determina cuando es necesario que el hashset trabaje con más hashcodes.

14. Explique, ¿Qué operaciones tienen las siguientes estructuras de datos en común: Set, List, Queue y Map?

Por lo general poseen un método para introducir un elemento nuevo, buscar un elemento existente, extraer un elemento presente y determinar si están vacías, cada uno variando la forma en que lo ejecuta.

15. Explique, ¿Qué tipo de información provee la notación Big-O para el diseño de software?

La notación Big-O es la forma de decir cuantas operaciones realiza un código y se usa para determinar su eficiencia en relación a otros de su misma índole funcional. Esto valores generalmente se toman pensando en que el código se ejecutara infinitas veces, generando funciones que predicen el total de operaciones realizadas (para estimar generalmente el tiempo de ejecución).

16. Explique, ¿De qué manera la incorrecta selección de una estructura de datos podría impactar el desarrollo de software?

Como principal, este error provoca la no posible ejecución de alguna funcionalidad requerida por el software, la poca eficiencia en que se ejecutan algunos procesos que si se lograron, el coste temporal para lidiar con la forma no trivial de resolver el requerimiento, entre otros.

17. En el marco de la utilización de los principios de ocultamiento de información, explique ¿de qué manera es posible implementar estos principios cuando se desarrolla un programa que utiliza el lenguaje de programación Java?.

Para el encapsulamiento de información que pueda provocar errores, se puede colocar las variables entre claves “{ }”, la cual ocultara su existencia fuera de estos márgenes. Tambíen, existen los modificadores de acceso, los cuales limitan las clases, por ende objetos, que pueden acceder a esos valores y métodos.

18. Explique, ¿A qué refiere el concepto “Primitive Obsession”?

Este termino hace referencia a un antipatrón de diseño, que consiste en llevar todas las variables, entendido como elementos del problema, a datos primitivos, es decir, ints, chars, booleans, doubles, entre otros.