NOMBRE Y PREGUNTA A RESPONDER (2 PREGUNTAS POR PERSONA PARA TERMINAR RÁPIDO)

ERICK: 3-4

1. ¿Qué es un miembro estático? ¿Cómo se usa este concepto en Java?

Static

La palabra clave static declara miembros (atributos, métodos y clases) que están asociados a la clase en lugar de a una instancia de la clase.

Utilidad

A veces puede ser útil tener una variable compartida por todos los objetos de la clase (algo parecido a una variable global). En este caso marcaríamos esta variable como static.

para no tener que crear clases a lo wey para usar un metodo (no creamos una clase para crear el main)

2. ¿Qué propiedades tiene un miembro estático?

es un atributo que no esta asociado a una instancia fija de una clase, es decir que es un elemento que comparten varias instancias

un ejemplo, el atributo generacion de becarios es estatico para todos los becarios de la generacion,

- si se modifica su valor se refleja para todas las instancias de la clase
- solo puede ser modificado por metodos estaticos
- las clases estaticas no se pueden instanciar
- los metodos estaticos no requieren de la creacion de una instancia de la clase para ser usados

https://www.adictosaltrabajo.com/tutoriales/la-directiva-static-en-java/

3. Menciona los 4 pilares de la POO

- ABSTRACCIÓN
- **RELACIONES
- HERENCIA

- ENCAPSULAMIENTO
- /* Y el P O L I M O R F I S M O? */

https://yo3783.wordpress.com/2009/03/29/pilares-de-la-orientacion-a-objetos/

4. ¿Qué es encapsulamiento?

Es la característica de la POO que permite el ocultamiento de la complejidad del código, pertenece a la parte privada de la clase y que no puede ser vista desde ningún otro programa.

/*En una clase los programadores definen variables y métodos, los cuales pueden ocultarse para que no puedan ser utilizadas en otras clases, o bien, pueden definirse de diferentes maneras para que puedan ser utilizadas libremente por otras clases y programadores.

Este concepto consiste en la ocultación del estado o de los datos miembro de un objeto, de forma que sólo es posible modificar los mismos mediante los métodos definidos para dicho objeto.*/

5. ¿Cómo se usa el encapsulamiento en Java?

Se usa declarando como private algunos atributos de una clase, estos atributos unicamente serán accesibles a través de métodos (get y set).

tambien el protected ayuda

6. Describe los modificadores de acceso en java

Modificadores de acceso

	La misma clase	Otra clase del mismo paquete	Subclase de otro paquete	Otra clase de otro paquete
public	X	X	X	X
protected	X	X	X	
default	X	X		
private	X			

7. ¿El concepto de encapsulamiento sólo se representa con los modificadores de acceso en java? Justifique su respuesta.

Sí, requerimos del modificador private. Según la definición de encapsulamiento, al usar private indicamos que solo podrán ser accesibles a través de métodos como get y set; sin embargo, el encapsulamiento nos habla también de tratar un objeto por sí solo.

8. ¿Qué es la abstracción?

Una abstracción puede definirse como:
las características especificas de un objeto, aquellas que lo distinguen de los demás tipos de objetos y que logran definir límites conceptuales respecto a quien está haciendo dicha abstracción del objeto.
Una abstracción se enfoca en la visión externa de un objeto, separa el comportamiento específico de un objeto, a esta división que realiza se le conoce como la barrera de abstracción, la cuál se consigue aplicando el principio de mínimo compromiso.

9. ¿Cómo se usa este concepto en java?

Se usa para al crear las clases (para declarar los atributos) y/o interfaces. Se deben expresar las características esenciales de un objeto, las cuales lo distinguen de los demás.

10. ¿Qué es una clase abstracta? ¿Para qué sirve?

Una clase que declara la existencia de métodos pero no la implementación de dichos métodos, se considera una clase abstracta.
https://es.slideshare.net/jcalmeida2/clases-abstractas-e-interfaces-en-java-15719490

/* Una clase abstracta no se puede instanciar(es decir no se pueden volver en objetos)pero si se puede heredar y las clases hijas serán las encargadas de agregar la funcionalidad a los métodos abstractos. */

‘’11. ¿Para qué es útil la herencia con las clases abstractas?

para definir los metodos de las clases hijas

las interfaces son clases meramente abstractas, por lo que en ellas solo se definen los metodos de la clase, esto es util para ahorrar codigo

las clases hijas sobrecargan - override los metodos de las clases padres y definen su propio comportamiento

12. ¿Cómo se castea objetos de clases distintas?

el casteo es para tratar objetos como si fueran instancias de otra clase

```
Object o = "whatever";
String str = (String)o;
```

lo que hacemos es crear una instancia de la clase objeto llamada o asignandole un valor string (hay un constructor modificado que ya recibe esos parametros), luego se crea una referencia de tipo string con un nombre (str) y se realiza el casteo :
se coloca el tipo de datos al que se desea cambiar entre parentesis seguido del nombre del objeto, con la referencia str se puede utilizar al objeto o (originalmente instancia de object) como una instancia de la clase tipo string y utilizar los metodos no estaticos propios de esta clase

13. ¿Para qué sirve la palabra reservada "Implements"?

Para implementar una interfaz, las interfaces permiten la herencia multiple, son muy utiles para reducir el codigo

14. Investigue la clase ArrayList, para que sirve y su diferencia con Arreglos

permite almacenar datos en memoria de forma similar a los Arrays, con la ventaja de que el numero de elementos que almacena, lo hace de forma dinámica, es decir, que no es necesario declarar su tamaño como pasa con los Arrays.

15. Investigue 5 Métodos principales de la Clase ArrayList

OMBRE MÉTODO	PARÁMETROS	DESCRIPCIÓN	VALOR RETORNADO
add	Elemento a insertar.	Añade un elemento a el ArrayList. Añade desde el final.	Booleano, indicando el estado de la operación.
remove	Índice a borrar.	Borra un elemento del ArrayList.	Booleano, indicando el estado de la operación.
clear	Ninguno	Limpia el ArrayList de elementos.	Ninguno
size	Ninguno	Devuelve el número de elementos	Cantidad de elementos en el ArrayList
get	Índice del elemento queremos	Devuelve el elemento en el índice indicado	Elemento en el índice indicado
iterator	Ninguno	Devuelve un iterador para recorrer el ArrayList.	Iterator
isEmpty	Ninguno	Indica si el ArrayList esta o o vacia	Booleano, indicando si esta vacio o no el ArrayList.
indexOf	Elemento	Devuelve la posición del elemento puesto como parámetro.	Posición del elemento en el ArrayList.

16. ¿Cómo se pasan argumentos a un programa desde linea de comandos?

Los valores que se envían se deben escribir a continuación del nombre del programa y separados por un espacio en blanco.
El array **array args** que aparece como argumento del método **main** es el encargado de recoger y almacenar estos valores.

17. Hacer un programa que contenga cada uno de los conceptos que se vieron hoy en clase (Omitiendo Herencia, Polimorfismo y Abstracción)

de volon pinpon

LAS EXTRAS. Investiga qué significa:

18. Casting

procedimiento para transformar una variable primitiva de un tipo a otro, o transformar un objeto de una clase a otra clase siempre y cuando haya una relación de herencia entre ambas

19. Parse

nos permite convertir caracteres numéricos a datos numéricos, es decir, convertir un número almacenado como String a un dato del tipo int, double u otro según se requiera.

20. Boxing

Autoboxing es la conversión automática que el compilador de Java hace que entre los tipos primitivos y sus clases de objetos(conocidos como wrappers). Por ejemplo, la conversión de un int a un Integer, un double con un Double, y así sucesivamente.

21. Unboxing

el unboxing es el opuesto del boxing, es la conversion de tipos de datos de tipo objeto (integer, character) a tipos de dato primitivos (int, char)
los tipos primitidos se tratan como variables, los tipos objeto heredan los metodos de la clase Object y tienen las propiedades de un objeto