**Diferencia entre interfaz y clase abstracta**

La interfaz unicamente contiene la definición de métodos que serán implementados en las clases

La clase abstracta puede contender definición de métodos que serán heredados así como la implementación de lógica en los métodos para rehusar en clases hijas

**Diferencia entre LinkedList y ArrayList**

LinkedList mantiene la colección ordenada en puede hacer recorrido de sus elementos hacia adelante o hacia atrás

ArrayList no garantiza que se mantengan los elementos en el orden en el que fueron ingresados, es mas rápida la inserción y lectura de sus elementos

**Cómo funciona el recolector de basura**

Cuando un elemento en memoria deja de tener una referencia se elimina del Heap.

El Heap se clasifica en 3 espacios de memoria. Cuando un Objeto o variable es declarado se crea en el **Eden**, posteriormente cuando se esta usando esa variable y asignándole un valor o referencia ya en memoria se crea en el espacio de ***sobrevivientes*** y ahí se queda hasta que pierde uso o referencia en memoria, el apartado de ***permanente*** es usado por las variables constantes y objetos que tienen un uso permanente en la aplicación

**Escribe la función de POW sin usar multiplicación/división ni las funciones de las mismas**

...

double numeroDiaAnioProgramador = Math.pow(2, 8);

System.out.println(“Dia del año que pertence el dia del programador: ” + numeroDiaAnioProgramador );

…

**Tenemos un partner el cual nos manda notificación de pago, el endpoint que**

**tenemos con este partner no debe de demorar en responder más de 20 ms esto**

**para cumplir 50 TPS (transaction per seconds) que nos exige el partner, tomando**

**en cuenta que cuando recibimos la notificación de pago, nosotros tenemos que**

**usar unos APIs en JSON que demoran en contestarnos 50ms.**

**1) ¿Cómo podemos solucionar este problema para cumplir con los 20ms?**

**Explica cómo lo harías, qué tecnologías utilizarías y si es posible muestra la**

**parte crucial de este código**

Se asume que las peticiones son de tipo GET entonces se tiene que realizar un cache para que las mismas peticiones no vayan al endpoint del partner

*@Bean*

*public HttpClient httpClient() {*

*return CachingHttpClientBuilder*

*.create()*

*.setCacheConfig(cacheConfig())*

*.build();*

*}*

*@Bean*

*public CacheConfig cacheConfig() {*

*return CacheConfig*

*.custom()*

*.setMaxObjectSize(500000) // 500KB*

*.setMaxCacheEntries(2000)*

*.build();*

*}*

Si las peticiones son de tipo post se debe implementar otras soluciones mas robustas de cache como puede ser hazelcast