PRUEBA DE ACCESO XPERTGROUP
ING. LUIS MOSQUERA

1. CÓDIGO
   a. La aplicación se encuentra divida en un componente backend y frontend, el back se implementa en dos capas, la capa de persistencias y controlador o aplicación; a la capa de persistencia pertenecen las clases: Point3D.java, Point3DRepository.java, StoreData.java, StoreDataRepository.java y LoadDataBases.java; y a la capa de aplicación pertenecen: CubeSumController.java, CubeSumNotFoundException.java, CubeSumNotFoundAdvice.java, Operation.java y CubeSumApplication.java. Por otro lado, el frontend hace las veces de la capa de vista siendo independiente del back e implementándose en angular, se estructuro con un solo componente llamado app y el cual define toda la vista de la aplicación.

   b. Responsabilidad de clases:
      i. ***Point3D.java:*** Clase que permite crear la entidad Point3D a partir de la cual se define la matriz 3D.
      ii. ***Point3DRepository.java:*** Interfaz que extiende un CRUD para la entidad Point3D.
      iii. ***StoreData.java:*** Clase que permite crear la entidad StoreData la cual almacena los datos para la ejecución se la suma de cubos.
      iv. ***StoreDataRepository.java:*** Interfaz que extiende un CRUD para la entidad StoreData.
      v. ***LoadDataBases.java:*** Clase que inicializa los repositorios o almacenamiento de las entidades anteriormente nombradas.
      vi. ***CubeSumController.java:*** Clase que implementa un Api REST mediante la cual de interactua con las entidades de la aplicación.
      vii. ***CubeSumNotFoundException.java:*** Clase que extiende RuntimeException para crear la excepción cuando no se encuentra la información de las entidades.
      viii. ***CubeSumNotFoundAdvice.java:*** Clase que devulve el mensaje de error cuando la excepción de no encontrado se dispara.
      ix. ***Operation.java:*** Clase complementaria que describe el objeto que se recibe al ejecutar una operación (UPDATE – QUERY) de la prueba.
      x. ***CubeSumApplication.java:*** Clase que ejecuta la aplicación:
      xi. ***app.component***: Componente visual que permite el ingreso y salida de datos de la aplicación.

2. PREGUNTAS
   a. **Responsabilidad Única**: Este principio hace referencia a que cada módulo o clase de un software o aplicación debe cumplir con una sola funcionalidad de este, en otras palabras, cada módulo o clase debe modificar el estado de una sola entidad o modelo.
   b. **Código Limpio**: Este concepto hace referencia a que cada módulo o clase que se implemente debe revelar fácilmente su propósito o intensión dentro del software o aplicación.