IMPLEMENTACIÓN EXCEPCIONES

PARTE 1:

Implemento primero un método que devuelve un array de cien números aleatorios comprendidos entre el cero y el quinientos. Después creo un método void que ejecuta el array creado a partir del anterior método, generando un índice aleatorio comprendido entre el -50 y el 150 (primero genero un Math.random con máximo 200 y luego le resto 50), después imprime un mensaje con el número aleatorio y el índice utilizado.

Para esta parte utilicé el try, catch. El catch se ejecuta si el índice del array está fuera de los límites.

PARTE 2:

Creo una nueva clase dentro de la clase Triangle llamada TriangleIllegalException que es subclase de Exception. Le creo un atributo String, y un constructor, el cual ejecuta este string (frase de explicación de la excepción).
A continuación asigno los atributos lado1, lado2 y lado3 a Triangle y creo un constructor para la clase.

He decidido utilizar cláusulas guardia para ésta aproximación, donde si uno de los lados es igual o menor a cero o si algún lado es menor a la suma de los otros dos lados, lanzará la excepción personalizada junto a un string de explicación. Si no ocurren errores, los datos del triángulo se inicializan correctamente.

PARTE 3:

Primero permito que las aserciones sean posibles en el proyecto, entrando a las propiedades del proyecto, luego a run, y poniendo el argumento -enableassertments.
Lo siguiente que hago es crear el método, simplemente imprime en pantalla la división de los parámetros int a y int b. Antes de finalizar el método, introduzco las aserciones o postcondiciones. Si b es cero se ejecutará un assertment, si b es menor a 0 se ejecutará otro assertment y si a es menor o igual a 0.

REFLEXIÓN DEL USO DE ASERCIONES

Siento que las aserciones son una gran herramienta para hacer saber al usuario o al desarrollador que una excepción ha ocurrido, y así sea una tarea fácil de resolver, agilizando el trabajo del equipo en conjunto.