* 1. Que es una @Anotación en Java
* 2. Cuáles son las Anotaciones de JUnit
* 3. TDD
* 4. Casos de Pruebas Cuenta Corriente
* 5. Ejercicio TDD
* **Anotaciones Java - @Anotación**
* Todos usamos en el día a día Anotaciones Java, pero no siempre entendemos como el lenguaje Java las procesa.
* Las Anotaciones **toman la forma de una declaración de interfaz con un carácter @ precediéndola, y marcada opcionalmente con meta-anotaciones**, como se ve debajo:
* @Retention(RetentionPolicy.RUNTIME)
* @Target({ElementType.METHOD})
* @Ignore
* @After
* @Before
* @AfterClass
* @BeforeClass
* @Test
* @FixMethodOrder
* Las Anotaciones en Java **permiten al programador declarar en su código fuente cómo debe comportarse el software**.
* En programación, **una Anotación Java** **es una forma de añadir**[**metadatos**](https://es.wikipedia.org/wiki/Metadatos)**al código fuente Java que están disponibles para la aplicación en**[**tiempo de ejecución**](https://es.wikipedia.org/wiki/Tiempo_de_ejecuci%C3%B3n)**o de compilación**.
* Las Anotaciones Java **pueden añadirse a los elementos** de programa tales como **clases, métodos,**[**metadatos**](https://es.wikipedia.org/wiki/Metadatos)**, campos, parámetros, variables locales, y paquetes**. Las Anotaciones Java son completamente accesibles al programador, mientras que el software se ejecuta, usando [reflexión](https://es.wikipedia.org/wiki/Reflexi%C3%B3n_(inform%C3%A1tica)).
* **Procesamiento de Anotaciones Java**: Cuando se compila el código fuente de Java, **el compilador Java almacena los metadatos de la Anotación en los ficheros/archivos de clases**. Posteriormente, **la**[**JVM**](https://es.wikipedia.org/wiki/JVM)**u otros programas pueden buscar los metadatos para determinar cómo interactuar con los elementos del programa o cambiar su comportamiento**.
* **Anotaciones JUnit**
* **JUnit** es un **conjunto de clases (**[**framework**](https://es.wikipedia.org/wiki/Framework)**)** que **permite realizar la ejecución de clases Java de manera controlada, para poder evaluar si el funcionamiento de cada uno de los métodos de la clase se comporta como se espera**.
* Es decir, en función de algún valor de entrada **se evalúa el valor de retorno esperado**; **si la clase cumple** con la especificación, entonces **JUnit devolverá que el método de la clase pasó exitosamente la prueba**; **en caso de que el valor esperado sea diferente** al que regresó el método durante la ejecución, **JUnit devolverá un fallo en el método correspondiente**.
* JUnit es también **un medio de controlar las**[**pruebas de regresión**](https://es.wikipedia.org/wiki/Pruebas_de_regresi%C3%B3n), necesarias cuando una parte del código ha sido modificado y se desea **ver que el nuevo código cumple con los requerimientos anteriores y que no se ha alterado su funcionalidad después de la nueva modificación**.
* @AfterClass: Significa que se ejecutara después de todos los métodos.
* @Before: Significa que se ejecutara antes de cada método.

@Ignore

* @Test: Se usa para identificar que el método será un método a probar.

@Test

@Test

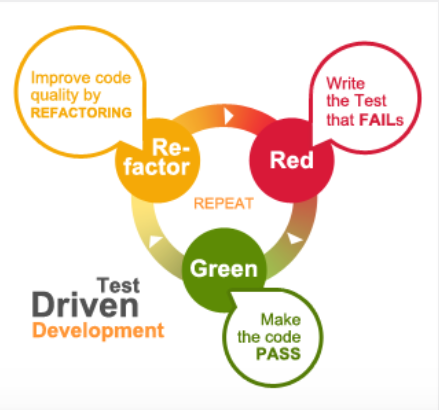
@Test

@Test

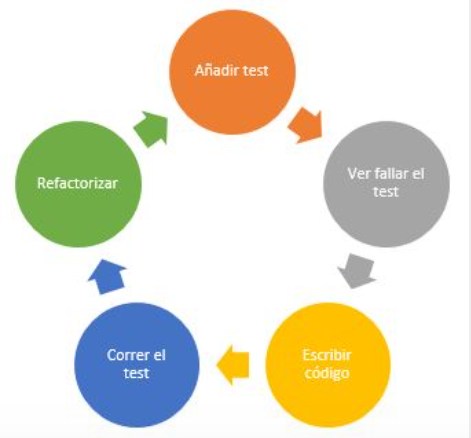
* @After: Significa que se ejecutara después de cada método.
* @BeforeClass: Significa que se ejecutara antes de todos métodos.
* @FixMethodOrder (MethodSorters.NAME\_ASCENDING): Nos ayuda a determinar el orden de ejecución de los Test en orden alfabetico del nombre del método.
* @Ignore: La anotación @Ignore **sirve para avisarle al JUnit que ignore este método**, esta anotación se utiliza generalmente cuando por alguna razón hemos modificado el código y estamos trabajando en el test. Puede ser que todavía nosotros no hayamos incluido código o que dependemos que algún desarrollador termine algo.
* **TDD [Desarrollo dirigido por Pruebas]**
* Es una **metodología que tienen como objetivo asegurar la calidad del software desde la fuente**. Está enfocado en optimizar pequeñas piezas de código; con incrementos de funcionalidad.
* TDD es un proceso iterativo en el cual el desarrollo está guiado por los test. Primero escribimos los test que expresan los requerimientos a cumplir luego desarrollamos para cumplir con dichos requerimientos.
* TDD propone el siguiente proceso:
* 1. **Test-First**: las pruebas se escriben antes del propio código a probar.

Se definen todos los test de la Unida a crear

* 2. **Automatización**: las pruebas del programa deben ser hechas en código, y con la sola ejecución del código de pruebas debemos saber si lo que estamos probando funciona bien o mal.
* 3. **Refactorización** posterior: para mantener la calidad del diseño, se cambia el diseño sin cambiar la funcionalidad para evitar duplicidad.
* 4. Repetir el proceso hasta completar todos los test



* En la práctica se implementan las siguientes acciones:
  + **Se crea primero el test**
  + Se verifica que el test falla
  + Se escribe el código para pasar el test
  + Se verifica que el test pasa
  + Se re-factoriza el código para evitar duplicidad.



* Ejercicio Práctico TDD
* Requisitos
  + Yo como usuario deseo, tener el producto de Cuenta Corriente con el siguiente comportamiento
  + 1. Es una cuenta corriente que tiene un crédito pre aprobado de 500 USD
  + 2. La cuenta acepta depósitos sin límite de valor
  + 3. Puedo realizar compras con la cuenta
  + 4. Para las compras puedo usar el saldo propio de los depósitos
  + 5. para las compra puedo usar el saldo crédito pre-aprobado
  + 6. Cuando se realizan compras con el pre-aprobado se debe validar el tope de crédito
  + 7. Cuando se realiza una compra y el valor del saldo deposito no alcanza a cubrir el valor de la compra se puede usar al mismo tiempo el valor del para-aprobado para completar el valor de la compra.
  + 8. Al hacer la compra primero se usara el valor del depósito en la cuenta, en caso de no ser suficiente se usará el valor del pre-aprobado pudiendo usarlo completamente para el pago.
* Casos de Prueba
  + Cp1 Dado Que Se ha creado una cuenta corriente Cuando realizo un depósito por valor de <valor1> Entonces Puedo ver mi saldo actualizado <saldo-actual> + <valor1>

Pasos:

* + - Crear cuenta
    - Realizar Deposito
    - Consultar Saldo
    - Validar que el Saldo corresponde al valor depositado.
  + Cp2 Dado Que Se ha creado una cuenta corriente Cuando realizo compra con la cuenta Entonces se valida si la cuenta tiene saldo mayor que cero <saldo>

Pasos:

* + - Crear cuenta
    - Realizar compra
    - Validar que pueda realizar compra con el valor depositado.
  + Cp2 Dado Que la cuenta corriente tiene saldo mayor que cero Cuando realizo compra con la cuenta Entonces se valida que valor compra <valor-compra> es inferior al saldo <saldo>
  + Cp3 Dado Que el valor de la compra es inferior al saldo Cuando realizo compra con la cuenta Entonces Se autoriza la compra y valor compra se descuenta de mi saldo. <saldo> - <valor-compra>
  + Cp4 Dado Que el valor de la compra es superior al saldo Cuando realizo compra con la cuenta Entonces Se muestra mensaje: "Fondos insuficientes", se valida historial crediticio, y si aplica, se muestra mensaje de oferta pre-aprobado
  + Cp5 Dado Que Se ha creado una cuenta corriente Cuando consulto el saldo de la cuenta Entonces Puedo ver mi saldo a favor <saldo-a-favor> y la lista de depósitos a mi cuenta <depósitos>
  + Cp6 Dado Que Se tiene crédito pre-aprobado, saldo en cuenta corriente en cero y valor compra inferior al tope pre-aprobado Cuando realizo compra con la cuenta Entonces Mostrar Mensaje solicitando confirmación de uso crédito pre-aprobado
  + Cp7 Dado Que Se tenga la confirmación de uso de crédito pre-aprobado Cuando realizo compra con la cuenta Entonces Autorizar la compra, valor compra descontarlo del crédito pre-aprobado y notificación al área de activos para gestión de cartera. <saldo-crédito> - <valor\_compra>
  + Cp8 Dado Que Se tiene crédito pre-aprobado, saldo en cuenta corriente en cero y valor compra superior al tope pre-aprobado Cuando realizo compra con la cuenta Entonces Mostrar mensaje informando que el valor de la compra supera el tope del crédito pre-aprobado
  + Cp9 Dado Que Se ha ingresado a la aplicación Cuando realizo la autenticación Entonces el sistema muestra mis productos
  + Cp10 Dado Que Se ha creado una cuenta corriente Cuando consulto el saldo de la cuenta Entonces la cuenta debe existir en la base de datos
  + Cp11 Dado Que Se ha creado una cuenta corriente Cuando consulto el saldo de la cuenta Entonces puedo ver que el valor del pre-aprobado es de 500 USD

**Preguntas**:

* + Qué pasa si el valor de la compra supera parcialmente el valor del saldo actual,
  + Se ajusta con el pre-aprobado ? .