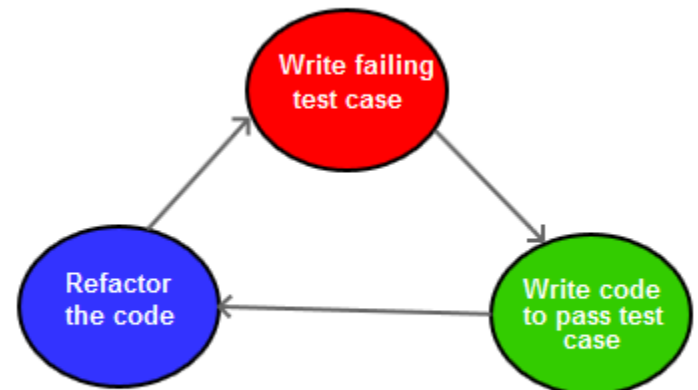





TDD Test Driven Development

TDD

- Test Driven Development es una metodología ágil.
- Sigue el ciclo:
 - Test → Code → Refactor





Ejercicio Guiado

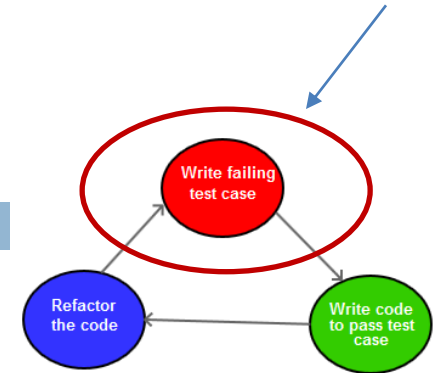
Ejercicio guiado

- Debemos escoger uno de los requisitos aportados por el cliente:
 - Crear un carrito de la compra vacío.
 - Cuando: se crea un carrito vacío.
 - Entonces: el conteo de productos del carrito será 0.
 - Añadir un producto al carrito de la compra:
 - Cuando: añadimos 1 unidad de "H&S Champú" a 5.25€.
 - Entonces: el precio total del carrito será 5.25€ y el total de unidades 1.
 - Añadir diferentes productos al carrito de la compra:
 - Cuando: añadimos 1 unidad de 'Jabón Marsella' a 2.15€.
 - Entonces: el precio total del carrito será 7.40€ y el total de unidades 2.



Primer requisito

Ejercicio Guiado



□ Escogemos el primer requisito:

- Crear un carrito de la compra vacío.
 - Cuando: se crea un carrito vacío.
 - Entonces: el conteo de productos del carrito será 0.

```
package com.tdd.test.shoppingcart;  
import org.junit.Assert;  
import org.junit.Test;
```

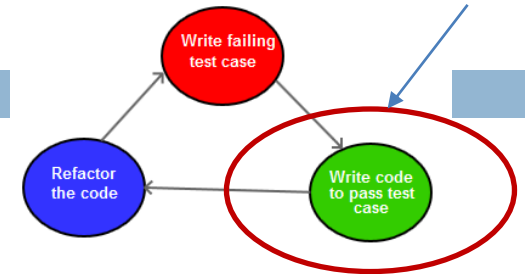
```
public class ShoppingCartAppTest {  
    @Test  
    public void testCreateEmptyShoppingCart() {  
        ShoppingCart cart = new ShoppingCart();  
        Assert.assertEquals(0, cart.getProductCount());  
    }  
}
```

X -> El test falla porque no tenemos código creado.

Ejercicio Guiado

❑ Escogemos el primer requisito:

- ❑ Crear un carrito de la compra vacío.
 - ❑ Cuando: se crea un carrito vacío.
 - ❑ Entonces: el conteo de productos del carrito será 0.

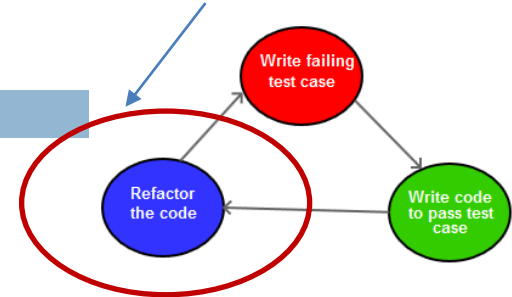


```
package com.tdd.shoppingcart;  
    public class ShoppingCart {  
  
        public int getProductCount() {  
            return 0;  
        }  
    }
```

X -> El test falla porque no tenemos código creado.

Ejercicio Guiado

- ❑ Escogemos el primer requisito:
 - ❑ Crear un carrito de la compra vacío.
 - ❑ Cuando: se crea un carrito vacío.
 - ❑ Entonces: el conteo de productos del carrito será 0.

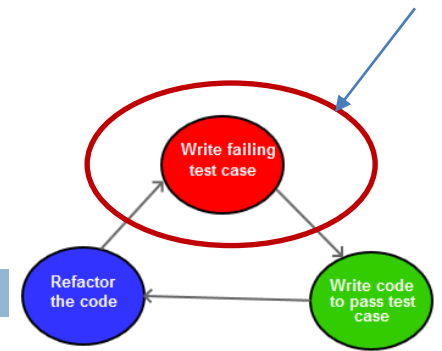


En este caso no es necesario refactorizar.



Segundo requisito

Ejercicio Guiado



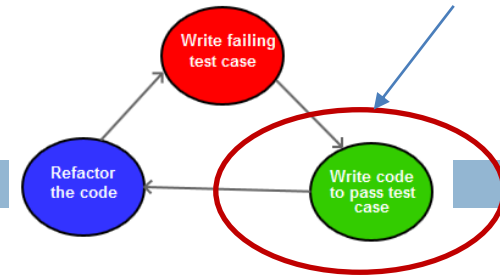
Escogemos el SEGUNDO REQUISITO:

- Añadir un producto al carrito de la compra:
 - Cuando: añadimos 1 unidad de "H&S Champú" a 5.25€.
 - Entonces: el precio total del carrito será 5.25€ y el total de unidades 1.

```
import org.junit.Assert;
import org.junit.Test;
import com.tdd.shoppingcart.Product;
import com.tdd.shoppingcart.ShoppingCart;
public class ShoppingCartAppTest {
    @Test
    public void testCreateEmptyShoppingCart() {
        ShoppingCart cart = new ShoppingCart();
        Assert.assertEquals(0, cart.getProductCount());
    }
    @Test
    public void testAddSingleProductToShoppingCart() {
        ShoppingCart cart = new ShoppingCart();
        Product product = new Product("HYS CHAMPU", 1, 5.25);
        cart.addProduct(product);
        Assert.assertEquals(1, cart.getProductCount());
    }
}
```

X -> El test falla porque no tenemos PRODUCTO creado.

Ejercicio Guiado



Escogemos el **SEGUNDO REQUISITO**:

- Añadir un producto al carrito de la compra:
 - Cuando: añadimos 1 unidad de "H&S Champú" a 5.25€.
 - Entonces: el precio total del carrito será 5.25€ y el total de unidades 1.

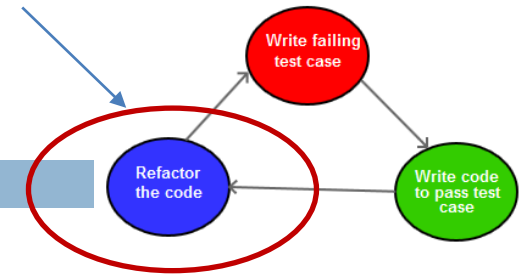
```
package com.tdd.shoppingcart;
public class Product {
    private String productName;
    private int quantity;
    private double totalPrice;

    public Product(String productName, int quantity, double totalPrice) {
        this.productName = productName;
        this.quantity = quantity;
        this.totalPrice = totalPrice;
    }
}
```

```
package com.tdd.shoppingcart;
public class ShoppingCart {
    public int getProductCount() {
        return 0;
    }
    public void addProduct(Product product) {
    }
}
```

X -> Uno de los test falla pero el otro no.

Ejercicio Guiado



Escogemos el SEGUNDO REQUISITO:

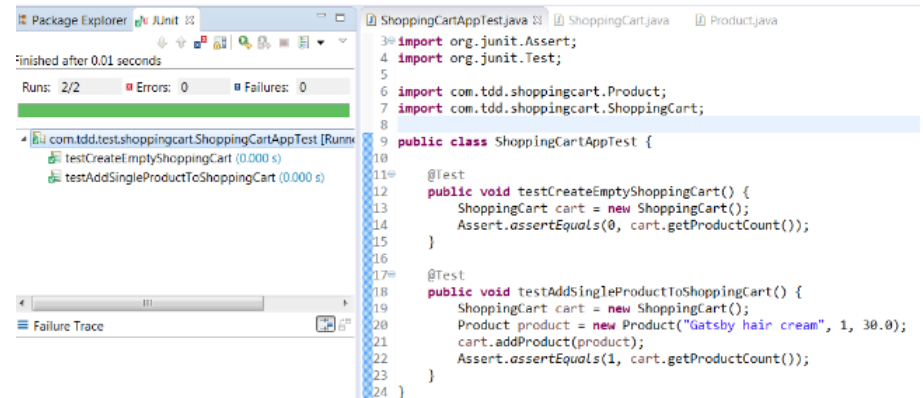
- Añadir un producto al carrito de la compra:
 - Cuando: añadimos 1 unidad de "H&S Champú" a 5.25€.
 - Entonces: el precio total del carrito será 5.25€ y el total de unidades 1.

```
package com.tdd.shoppingcart;
```

```
import java.util.ArrayList;
import java.util.List

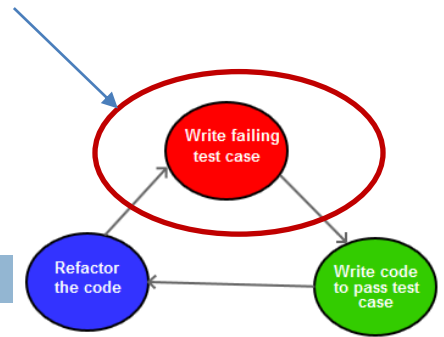
public class ShoppingCart {
    private List<Product> productList = new ArrayList<>();
    public int getProductCount() {
        return productList.size();
    }

    public void addProduct(Product product) {
        productList.add(product);
    }
}
```



Los dos test pasan

Ejercicio Guiado



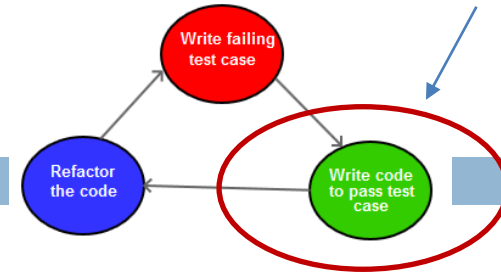
Escogemos el SEGUNDO REQUISITO:

- Añadir un producto al carrito de la compra:
 - Cuando: añadimos 1 unidad de "H&S Champú" a 5.25€.
 - Entonces: el precio total del carrito será 5.25€ y el total de unidades 1.

```
package com.tdd.test.shoppingcart;
import org.junit.Assert;
import org.junit.Test;
import com.tdd.shoppingcart.Product;
import com.tdd.shoppingcart.ShoppingCart;
public class ShoppingCartAppTest {
    @Test
    public void testCreateEmptyShoppingCart() {
        ShoppingCart cart = new ShoppingCart();
        Assert.assertEquals(0, cart.getProductCount());
    }
    @Test
    public void testAddSingleProductToShoppingCart() {
        ShoppingCart cart = new ShoppingCart();
        Product product = new Product("HYS CHAMPU", 1, 5.25);
        cart.addProduct(product);
        Assert.assertEquals(1, cart.getProductCount());
        Assert.assertEquals(5.25, cart.getTotalCartValue());
    }
}
```

X -> El test falla porque no tenemos función para el valor total

Ejercicio Guiado



Escogemos el SEGUNDO REQUISITO:

- Añadir un producto al carrito de la compra:
 - Cuando: añadimos 1 unidad de "H&S Champú" a 5.25€.
 - Entonces: el precio total del carrito será 5.25€ y el total de unidades 1.

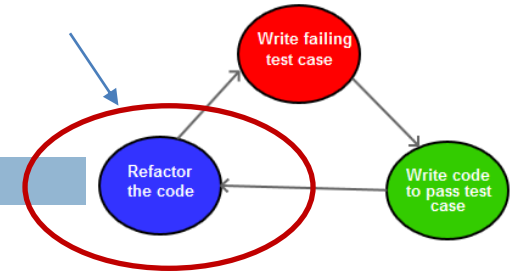
```
package com.tdd.shoppingcart;

import java.util.ArrayList;
import java.util.List
public class ShoppingCart {
    private List<Product> productList = new ArrayList<>();
    public int getProductCount() {
        return productList.size();
    }

    public void addProduct(Product product) {
        productList.add(product);
    }

    public double getTotalCartValue() {
        return 0.0;
    }
}
```

Ejercicio Guiado



Escogemos el SEGUNDO REQUISITO:

- Añadir un producto al carrito de la compra:
 - Cuando: añadimos 1 unidad de "H&S Champú" a 5.25€.
 - Entonces: el precio total del carrito será 5.25€ y el total de unidades 1.

```
package com.tdd.shoppingcart;
import java.util.ArrayList;
import java.util.List

public class ShoppingCart {
    private List<Product> productList = new ArrayList<>();

    public int getProductCount() {
        return productList.size();
    }

    public void addProduct(Product product) {
        productList.add(product);
    }

    public double getTotalCartValue() {
        if (productList.size() > 0) {
            for (Product product : productList) {
                totalCartValue = totalCartValue + product.getTotalPrice();
            }
        }
        return totalCartValue;
    }
}
```

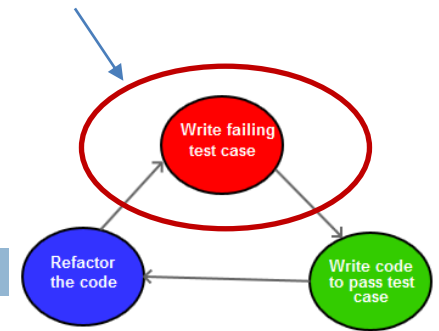
```
ShoppingCartAppTest.java
4 import org.junit.Test;
5
6 import com.tdd.shoppingcart.Product;
7 import com.tdd.shoppingcart.ShoppingCart;
8
9 public class ShoppingCartAppTest {
10
11     @Test
12     public void testCreateEmptyShoppingCart() {
13         ShoppingCart cart = new ShoppingCart();
14         Assert.assertEquals(0, cart.getProductCount());
15     }
16
17     @Test
18     public void testAddSingleProductToShoppingCart() {
19         ShoppingCart cart = new ShoppingCart();
20         Product product = new Product("Gatsby hair cream", 1, 30.0);
21         cart.addProduct(product);
22         Assert.assertEquals(1, cart.getProductCount());
23         Assert.assertEquals(30.0, cart.getTotalCartValue(), 0.0);
24     }
25 }
```

Los dos test pasan



Tercer requisito

Ejercicio Guiado



Escogemos el TERCER REQUISITO:

- Añadir diferentes productos al carrito de la compra:
 - Cuando: añadimos 1 unidad de 'Jabón Marsella' a 2.15€.
 - Entonces: el precio total del carrito será 7.40€ y el total de unidades 2.

```
@Test
public void addDifferentProductsToTheCart(){
    ShoppingCart cart = new ShoppingCart();
    Product gatsByCream = new Product("HYS CHAMPU", 1, 5.25);
    Product bvlgiSoap = new Product("JABON MARSELLA", 1, 7.40);
    cart.addProduct(gatsByCream);
    cart.addProduct(bvlgiSoap);
    Assert.assertEquals(2, cart.getProductCount());
    Assert.assertEquals(12.65, cart.getTotalCartValue(), 0.0);
}
```

El test pasa