**UNIVERSIDAD DE SAN BUENAVENTURA**

**ANALISIS Y DISEÑO DE ALGORITMOS**

**PRUEBA PRÁCTICA - BACKEND DEVELOPER  
PRUEBAS UNITARIAS**

**LUIS FELIPE VELASCO TAO**

**26 DE FEBRERO**

**2020**

Contenido

[Pruebas unitaria proyecto Panadería 5](#_Toc39607610)

[Testeo de conexión a la base de datos 5](#_Toc39607611)

[Testeo 7](#_Toc39607612)

[testConexion 8](#_Toc39607613)

[testConexion 9](#_Toc39607614)

[Testeo de gestión de los productos 10](#_Toc39607615)

[Testeo 15](#_Toc39607616)

[TesteoInicio 15](#_Toc39607617)

[TestBuscarValor 16](#_Toc39607618)

[TestConteo 18](#_Toc39607619)

[Testeo de la modificación y actualización de 21](#_Toc39607620)

[Testeo 25](#_Toc39607621)

[testAnadirProducto 25](#_Toc39607622)

[testModificar 27](#_Toc39607623)

[testEliminar 30](#_Toc39607624)

[Test de búsqueda de productos 33](#_Toc39607625)

[TestBusqueda 35](#_Toc39607626)

[Análisis 37](#_Toc39607627)

[Consideraciones 38](#_Toc39607628)

[Test de fecha 39](#_Toc39607629)

[TestFechas 41](#_Toc39607630)

[Análisis 42](#_Toc39607631)

[Consideraciones 42](#_Toc39607632)

[Test de total de ventas 44](#_Toc39607633)

[Testeo 48](#_Toc39607634)

[TestTotalVentasGeneral 48](#_Toc39607635)

[TestTotalVentasPorDia 50](#_Toc39607636)

[TestTotalVentasFechaNoExitente 52](#_Toc39607637)

[Testeo del DAO de productos 56](#_Toc39607638)

[Testeo 61](#_Toc39607639)

[InsertTest 61](#_Toc39607640)

[DeleteTest 63](#_Toc39607641)

[UpdateTest 64](#_Toc39607642)

[BuscarTodosTest 66](#_Toc39607643)

[ReporteFechaTest 67](#_Toc39607644)

[xBuscarNombreTest 69](#_Toc39607645)

[ReporteModificacionDB 70](#_Toc39607646)

[ANALISIS GENERAL 73](#_Toc39607647)

# Pruebas unitaria proyecto Panadería

Las siguientes pruebas unitarias se realizarán sobre el proyecto creado para la gestión de ventas de una panadería, las pruebas se realizarán sobre los métodos de gestión, tanto de los la manipulación por parte del usuario, como de la manipulación de los datos en la base de datos.

**Las pruebas se realizarán con Junit4 debido a inconvenientes con la versión de Junit5 en Netbeans 8.2, y problemas técnicos con la actualización de los recursos necesarios para la instalación de Junit 5.**

# Testeo de conexión a la base de datos

Se requiere comprobar la correcta conexión del programa a la base de datos, por lo cual, se van a realizar el testeo de los métodos perteneciente a la clase ConexionOracle, los cuales se encargan de validar la conexión a la base de datos por medio dl path de conexión y retornar la conexión creada.

public class ConexionDBTest {

ConexionOracle co;

Connection con;

static Logger logger = null;

/\*\*

\* Inicializacion de los objetos para la conexion a la base de datos

\*/

public ConexionDBTest() {

this.co = new ConexionOracle();

}

/\*\*

\* Anotacion que verifica la iniciacion de la prueba

\*/

@BeforeClass

public static void ejecutarAntes() {

logger = Logger.getLogger("TestAnotacionesJUnit");

logger.info("Se van a ejecutar los Tests…");

}

/\*\*

\* Anotacion que verifica la finalizacion de la prueba

\*/

@AfterClass

public static void ejecutarDespues() {

logger.info("Se han ejecutado los Tests…");

logger = null;

}

/\*\*

\* Metodo de testeo para comprobar la correcta conexion del programa a la

\* base de datos

\*/

@Test

public void testConexion() {

assertTrue(co.conector());

con = co.getCon();

}

/\*\*

\* Metodo de testeo para comprobar la conexion existente del programa con la

\* base de datos

\*/

@Test

public void testObtenerConexion() {

assertTrue(con == co.getCon());

}

}

## Testeo

A continuación, se presentarán los métodos de testeo y sus resultados.

### testConexion

En este método de testeo se corrobora la correcta conexión del programa a la base de datos, por medio de la cual se realizarán sentencias SQL para la extracción de información.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | co.conector()  Este método se encarga de generar el path de conexión a la base de datos realizar la conexión | **Co:** objeto de la clase ConexionOracle  **Conector():** método que retorna un valor booleano que verifica la creación de la conexion |  | True |

#### Análisis

El método de conexión a la base de datos funciona de la forma esperada, sin generar errores en su conexión.

#### Consideraciones

No se requieren modificaciones en el método de conexión.

### testConexion

En este metodo de testeo se pretende corroborar la obtención de la conexión a la base de datos necesria para la ejecución de sentencias SQL sobre esta.

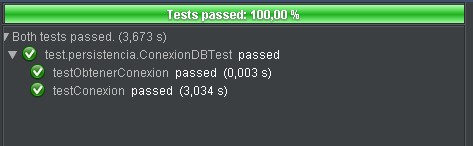
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | con == co.getCon() | **getCon:** metodo de obtención de la conexión generada previamente. | Con: objeto contenedor de la conexión a la base de datos | True |

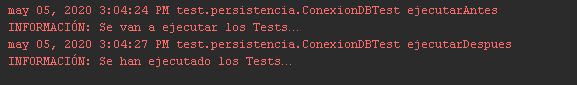
#### Análisis

La conexión a la base de datos obtenida previamente se corrobora que no está vacía y que es igual a la contenida en la clase.

#### Consideraciones

No se requieren modificaciones en los métodos testeados.





# Testeo de gestión de los productos

En este conjunto de pruebas se planea comprobar la correcta indexación y eliminación de un objeto tipo Producto en el sistema. Para la cual se requiere verificar las siguientes acciones:

1. Añadir el producto al sistema
2. Buscar el producto en el sistema
3. Listar los productos existentes
4. Contabilizar los productos del sistema
5. Eliminar el producto añadido

Por lo cual se estructura de la siguiente forma el test:

public class AnadirBuscarEliminarTest {

Gestion\_Producto gp;

Producto p;

/\*\*

\* Inicialización de los objetos

\* @param Gestion\_Product gp = objeto con todos los métodos de acceso al

\* sistema

\* @param Producto p = producto con valores aleatorios para el ejercicio

\*/

public AnadirBuscarEliminarTest() {

this.gp = new Gestion\_Producto();

this.p = new Producto("PAN INTEGRAL", "05/06/2020", 1500.00, 2, "PROPIO", gp.fechaActual(), 3000.00);

}

/\*\*

\* Este método de testeo se ejecutará antes, en el cual se comprobara el

\* correcto indexado del producto al sistema

\*

\* @param Gestion\_Product gp = objeto con todos los metodos de acceso al

\* sistema

\* @param Producto p = producto con valores aleatorios para el ejercio

\*/

@Before

public void testInicio() {

assertTrue("SE AÑADIO EL ELEMENTO", gp.anadirElemento(p));

}

/\*\*

\* Metodo de testeo que se ejecutara despues del metodo de testeto principal

\* en donde se realizará el listado y la eliminacion del producto

\*

\* @param Gestion\_Product gp = objeto con todos los metodos de acceso al

\* sistema

\* @param Producto p = producto con valores aleatorios para el ejercio

\*/

@After

public void testConteo() {

System.out.println(" - " + gp.tamanio());

gp.listar();

assertTrue(gp.tamanio() == 8);

assertTrue("SE ELIMINO EL ELEMENTO", gp.retirarElemento(p.getNombre(), gp.fechaActual()));

assertTrue(gp.tamanio() == 7);

}

/\*\*

\* Método principal donde se comprobará la búsqueda del producto ingresado

\* al sistema

\*

\* @param Gestion\_Product gp = objeto con todos los métodos de acceso al

\* sistema

\* @param Producto p = producto con valores aleatorios para el ejercio

\* @param Producto pb = objeto donde se almacenará la busqueda del producto

\* indexado

\*/

@Test

public void testBuscarValor() {

Producto pb = gp.buscarProducto(p.getNombre());

assertNotNull(pb);

assertTrue(pb.getNombre().equals(p.getNombre())

&& pb.getFechaC().equals(p.getFechaC())

&& pb.getFechaVencimiento().equals(p.getFechaVencimiento()));

}

}

## Testeo

A continuación, se encontrará el análisis de los métodos de testeo

### TesteoInicio

En este método de testeo se pretende comprobar el correcto funcionamiento del método de añadir productos al sistema.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | gp.anadirElemento(p)  Este es un método que retorna un valor booleano que comprueba el correcto indexado del producto al sistema | Gp: clase de gestión de los productos  anadirElemento: Metodo en donde se valida la no existencia del producto para ser indexado en el sistema  p: producto a ser indexado | new Producto ("PAN INTEGRAL", "05/06/2020", 1500.00, 2, "PROPIO", gp.fechaActual(), 3000.00) | True |
| 2 | !(gp.anadirElemento(p)) |  | new Producto ("PAN INTEGRAL", "05/06/2020", 1500.00, 2, "PROPIO", gp.fechaActual(), 3000.00) | True |

#### Análisis

El método para añadir los productos al sistema funciona perfectamente, comprobando que este valida la existencia del producto en el sistema ya que evita los registros duplicados de productos.

#### Consideraciones

No se requieren cambios, ya que el método de añadir productos funciona tal como se esperaba.

### TestBuscarValor

En este método de testeo se pretende comprobar el correcto funcionamiento del método de búsqueda de productos en la lista.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | NotNull(p) | **pb**: producto en donde se almacena la búsqueda del producto anteriormente agregado | new Producto ("PAN INTEGRAL", "05/06/2020", 1500.00, 2, "PROPIO", gp.fechaActual(), 3000.00) | True |
| 2 | pb.getNombre().equals(p.getNombre())  && pb.getFechaC().equals(p.getFechaC())  && pb.getFechaVencimiento().equals(p.getFechaVencimiento()) | Pb:producto encontrado previamente  P: producto ingresado al sistema | new Producto ("PAN INTEGRAL", "05/06/2020", 1500.00, 2, "PROPIO", gp.fechaActual(), 3000.00) | True |

#### Análisis

En esta prueba se comprueba la correcta creación de los objetos y que los atributos de este coincidan con los esperados por el usuario.

#### Consideraciones

No se requiere realizar modificaciones a los métodos involucrados.

### TestConteo

En este método de testeo se pretende comprobar el correcto conteo de los productos indexados antes y después de la eliminación de uno de estos.

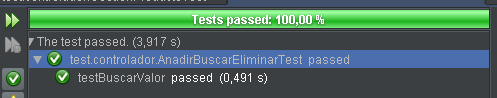
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | gp.tamanio() == 8 | **Tamanio():**  método que retorna la cantidad de elementos en la lista |  | True |
| 2 | gp.retirarElemento(p.getNombre(), gp.fechaActual())  Metodo que retorna un valor booleano para verificar la eliminación de un objeto del sistema | **p**: producto usado en la prueba | new Producto ("PAN INTEGRAL", "05/06/2020", 1500.00, 2, "PROPIO", gp.fechaActual(), 3000.00) | True |
| 3 | gp.tamanio() == 7 | **Tamanio():**  método que retorna la cantidad de elementos en la lista |  | True |

#### Análisis

Al realizar la verificación de los valores esperados, antes y después de la eliminación del producto del sistema, se obtienen los valores tanto que hacen relación con la base de datos con el listado de elementos, además de que se puede corroborar la correcta eliminación de los objetos del sistema.

#### Consideraciones

No se requieren hacer modificaciones.



# 

# Testeo de la modificación y actualización de

En este conjunto de pruebas se planea comprobar la correcta modificación de un objeto tipo Producto en el sistema. Para la cual se requiere verificar las siguientes acciones:

1. Añadir el producto al sistema
2. Comprobar que no se añadan elementos vacíos al sistema
3. Modificar el producto añadido
4. Listar los productos existentes
5. Eliminar el producto añadido

Por lo cual se estructura de la siguiente forma el test:

public class AnadirModificarListarTest {

Gestion\_Producto gp;

Producto p, pb;

static Logger logger = null;

/\*\*

\*Se inicializan los objetos requeridos para la prueba

\* @param gp = objeto de gestion de los productos del sistema

\* @param p,pb = productos que se usaran para la prueba

\* @see El objeto logger se inicializa afuera debido a que se requiere que sea de tipo estatico

\*/

public AnadirModificarListarTest() {

this.gp = new Gestion\_Producto();

this.p = new Producto("PAN INTEGRAL", "05/06/2020", 1500.00, 2, "PROPIO", gp.fechaActual(), 3000.00);

this.pb = null;

}

/\*\*

\*Anotacion que verifica la iniciacion de la prueba

\*/

@BeforeClass

public static void ejecutarAntes() {

logger = Logger.getLogger("TestAnotacionesJUnit");

logger.info("Se van a ejecutar los Tests…");

}

/\*\*

\*Anotacion que verifica la finalizacion de la prueba

\*/

@AfterClass

public static void ejecutarDespues() {

logger.info("Se han ejecutado los Tests…");

logger = null;

}

/\*\*

\*En esta prueba que se realiza previamente

\* se valida la correcta indexacion

\* de los valores al sistema, y que no se puedan añadir objetos vacios

\*/

@Before

public void testAnadirProducto() {

assertTrue(gp.anadirElemento(p));

assertTrue(!gp.anadirElemento(pb));

}

/\*\*

\*

\*/

@After

public void testEliminar() {

System.out.println("Antes de Eliminar");

gp.listar();

assertTrue(gp.retirarElemento(pb.getNombre(), "05/05/2020"));

System.out.println("Despues de Eliminar");

gp.listar();

System.out.println("\*\*" + gp.getpEliminado().toString());

assertTrue(gp.getpEliminado().toString().equals(pb.toString()));

}

/\*\*

\* Metodo de testeo en donde se evalua

\* la correcta modifiacion del objeto previamente añadido

\*/

@Test

public void testModificar() {

String nombreM = "PAN ARABE";

String fechaVM = "10/07/2020";

double costoM = 2550.00;

int cantidadM = 3;

double totalM = costoM \* cantidadM;

pb = new Producto(nombreM, fechaVM, costoM, cantidadM, "BIMBO", gp.fechaActual(), totalM);

assertNotSame(pb, p);

assertTrue(gp.modificar(pb, p.getNombre()));

}

}

## Testeo

A continuación, se encontrará el análisis de los métodos de testeo

### testAnadirProducto

En este método de testeo se pretende comprobar el correcto funcionamiento del método de añadir productos al sistema.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | gp.anadirElemento(p)  Este es un método que retorna un valor booleano que comprueba el correcto indexado del producto al sistema | **Gp**: clase de gestión de los productos  **anadirElemento**: Metodo en donde se valida la no existencia del producto para ser indexado en el sistema  **p**: producto a ser indexado | new Producto ("PAN INTEGRAL", "05/06/2020", 1500.00, 2, "PROPIO", gp.fechaActual(), 3000.00) | True |
| 2 | !(gp.anadirElemento(pb)) | **Gp**: clase de gestión de los productos  **anadirElemento**: Metodo en donde se valida la no existencia del producto para ser indexado en el sistema  **pb**: objeto vacio | null | True |

#### Análisis

El método para añadir los productos al sistema funciona perfectamente, comprobando que este valida la existencia del producto en el sistema ya que evita los registros duplicados de productos, además de evitar los objetos vacíos.

#### Consideraciones

No se requieren cambios, ya que el método de añadir productos funciona tal como se esperaba.

### testModificar

En este método de testeo se pretende comprobar el correcto listado, y eliminación de los objetos, además de corroborar el almacenamiento temporal del objeto eliminado del sistema.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | assertNotSame(pb, p)  se valida que los objetos no son iguales, para continuar con la modificacion | **P:** producto original  **pb**: producto modificado | new Producto("PAN INTEGRAL", "05/06/2020", 1500.00, 2, "PROPIO", gp.fechaActual(), 3000.00);  double costoM = 2550.00;  int cantidadM = 3;  double totalM = costoM \* cantidadM;  pb = new Producto("PAN ARABE", "10/07/2020", costoM, cantidadM, "BIMBO", gp.fechaActual(), totalM); | True |
| 2 | gp.modificar(pb, p.getNombre()) | **Gp**: clase de gestión de los productos  **modificar**: En este método se comprueba la correcta modificacion del producto asociado con el nombre del producto original  **pb**: objeto modificado | double costoM = 2550.00;  int cantidadM = 3;  double totalM = costoM \* cantidadM;  pb = new Producto("PAN ARABE", "10/07/2020", costoM, cantidadM, "BIMBO", gp.fechaActual(), totalM); | True |

#### Análisis

En este método se comprueba que se realiza la correcta modificación de los objetos almacenados.

#### Consideraciones

No se requieren cambios, los métodos de modificación funcionan de la forma esperada.

### testEliminar

En este método de testeo se pretende comprobar el correcto listado, y eliminación de los objetos, además de corroborar el almacenamiento temporal del objeto eliminado del sistema.

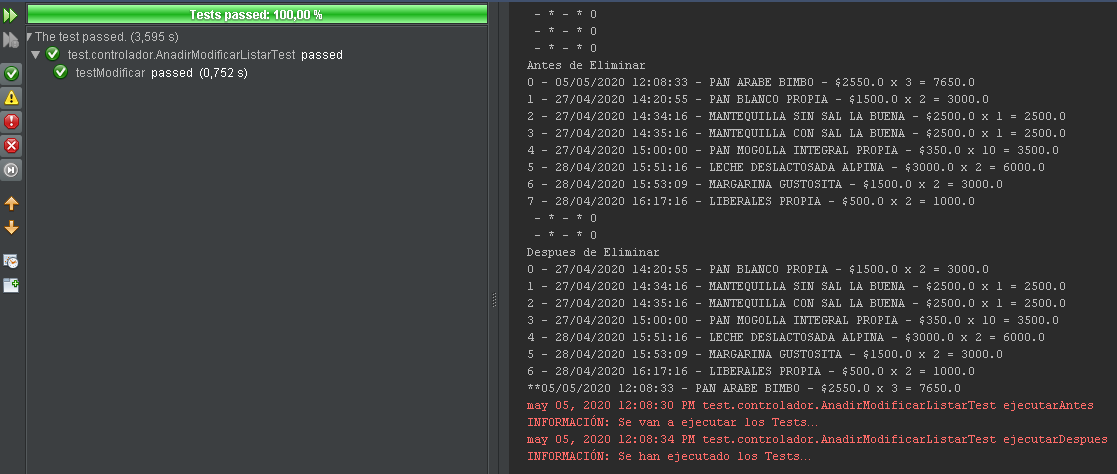
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | gp.retirarElemento(pb.getNombre(), "05/05/2020")  Este método retorna un valor booleano para la verificación de la eliminación de los productos basados en el nombre y la fecha en que se realizado la venta | **Gp**: clase de gestión de los productos  **retirarElemento**: método en donde se realiza la búsqueda y eliminacion de los objetos tipo producto del sistema  **pb**: producto modificado | double costoM = 2550.00;  int cantidadM = 3;  double totalM = costoM \* cantidadM;  pb = new Producto("PAN ARABE", "10/07/2020", costoM, cantidadM, "BIMBO", gp.fechaActual(), totalM); | True |
| 2 | gp.getpEliminado().toString().equals(pb.toString()) | **Gp**: clase de gestión de los productos  **getpEliminado**: Metodo en donde se almacena el ultimo producto eliminado  **pb**: objeto modificado |  | True |

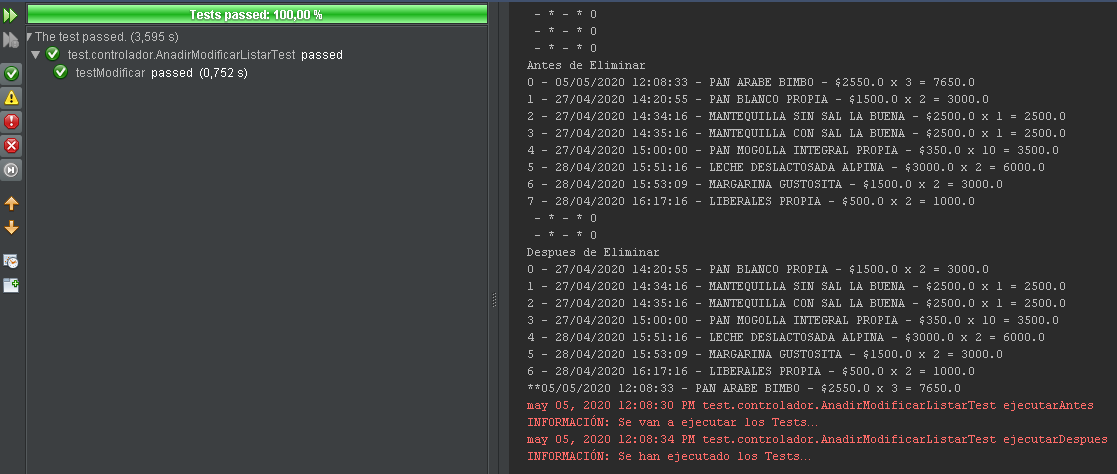
#### Análisis

En este método se comprueba que se realiza la eliminación del objeto indicado, con los parámetros ingresados y validando el almacenamiento del objeto correctamente.

#### Consideraciones

No se requieren cambios, ya que el método de eliminación de productos y listado de estos funciona tal como se esperaba.





# Test de búsqueda de productos

En este test se pretende corroborar el correcto funcionamiento del método de obtención de los productos, los cuales se encuentran almacenados en una lista, por lo cual se accede a estos por medio de su ubicación.

Para esto se pretende observar como se encuentra la lista y validar los valores entregados con los deseados.

public class ObtenerElementoTest {

Gestion\_Producto gp;

static Logger logger = null;

/\*\*

\* Se inicializa la clase de gestion de productos

\*/

public ObtenerElementoTest() {

this.gp = new Gestion\_Producto();

}

/\*\*

\*Anotacion que verifica la iniciacion de la prueba

\*/

@BeforeClass

public static void ejecutarAntes() {

logger = Logger.getLogger("TestAnotacionesJUnit");

logger.info("Se van a ejecutar los Tests…");

}

/\*\*

\*Anotacion que verifica la finalizacion de la prueba

\*/

@AfterClass

public static void ejecutarDespues() {

logger.info("Se han ejecutado los Tests…");

logger = null;

}

/\*\*

\* Se imprime la lista para verificar la busqueda

\*/

@Before

public void setUp() {

System.out.println("Listado elementos");

gp.listar();

}

/\*\*

\* En este metodo se valida que los productos en las posiciones 3 y 5 corresponden a los deseados

\*/

@Test

public void testBusqueda() {

assertTrue(gp.getProducto(3).toString().contains("PAN MOGOLLA INTEGRAL PROPIA"));

assertTrue(gp.getProducto(5).toString().contains("MARGARINA GUSTOSITA"));

}

}

## TestBusqueda

En este teste se van a relacionar los valores deseados con los arrojados por el método getProducto(), el cual requiere la posición que se desea obtener.

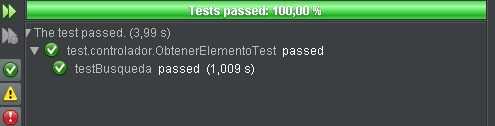
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | gp.getProducto(3).toString().contains("PAN MOGOLLA INTEGRAL PROPIA") | **Gp**: clase de gestión de los productos  **getProducto**: este método se encargará de retornar los productos contenidos en la lista basados en su posicion | 3 : posición deseada | True |
| 2 | gp.getProducto(5).toString().contains("MARGARINA GUSTOSITA") | **Gp**: clase de gestión de los productos  **getProducto**: este método se encargará de retornar los productos contenidos en la lista basados en su posicion | 5: posición deseada | True |
| 3 | gp.getProducto(8) == null | **Gp**: clase de gestión de los productos  **getProducto**: este método se encargará de retornar los productos contenidos en la lista basados en su posición | 8: posicion deseada  Esta posicion esta fuera del rango | true |

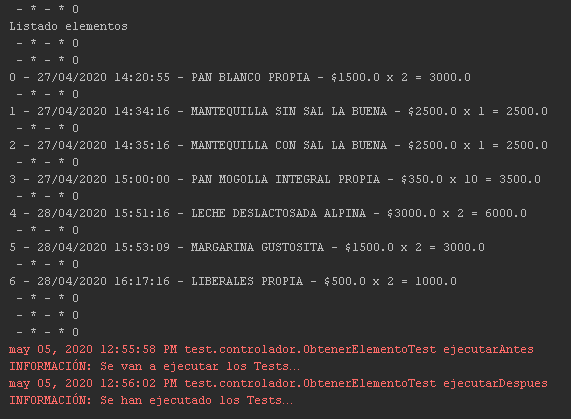
### Análisis

El testeo de los valores contendidos en la lista arroja los valores deseados por el usuario, además de validar el rango de la lista.

### Consideraciones

El método de obtención no requiere modificaciones





# Test de fecha

En este test se pretende comprobar el correcto funcionamiento del método de obtención de fecha y hora actual y que este entregue la fecha y hora en el formato de fecha y hora esperado.

Para esto se pretende corroborar que la fecha y hora entregada corresponda con la actual, y observar el formato en que se entrega la fecha y hora actúa.

public class FechasTest {

Gestion\_Producto gp;

static Logger logger = null;

public FechasTest() {

this.gp = new Gestion\_Producto();

}

/\*\*

\* Anotacion que verifica la iniciacion de la prueba

\*/

@BeforeClass

public static void ejecutarAntes() {

logger = Logger.getLogger("TestAnotacionesJUnit");

logger.info("Se van a ejecutar los Tests…");

}

/\*\*

\*Anotacion que verifica la finalizacion de la prueba

\*/

@AfterClass

public static void ejecutarDespues() {

logger.info("Se han ejecutado los Tests…");

logger = null;

}

/\*\*

\* En este metodo se comprueba el correcto formateo de la fecha y hora actual

\* requerida en la indexacion de los productos

\*/

@Test

public void testFechas(){

String fecha = "05/05/2020";

String hora = "13";

String fechaHActual = gp.fechaActual();

assertTrue(fechaHActual.contains(fecha) && fechaHActual.contains(hora));

}

}

## TestFechas

En este test se corrobora el funcionamiento del método fechaActual, encargado darles a los productos la fecha en que fueron vendidos.

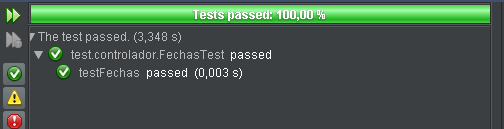
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | fechaHActual.contains(fecha) && fechaHActual.contains(hora | **Gp**: clase de gestión de los productos  **getProducto**: este método se encargará de retornar los productos contenidos en la lista basados en su posicion | String fecha = "05/05/2020";  String hora = "13";  String fechaHActual = gp.fechaActual();  05/05/2020 13:28:17 | True |

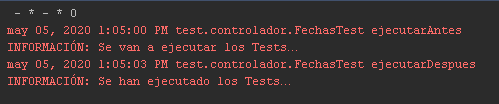
### Análisis

El método para la obtención de la fecha funciona acorde a los parámetros del formato de fecha y hora establecidos, entregando la fecha en formato DD/MM/AAAA y la hora HH:mm:ss (hora en formato 24 horas)

### Consideraciones

Este método no requiere correcciones.





# Test de total de ventas

En este test se pretende corroborar que se entregue un valor correspondiente al total de ventas y que este se acondicione a la lista que se le entregue, por lo tanto, se realizara un testeo general a los valores generales de ventas y uno enfocado a una lista de los valores de ventas de un a fecha específica.

public class TotalVentasTest {

Gestion\_Producto gp;

static Logger logger = null;

double total = 0.0;

/\*\*

\*

\*/

public TotalVentasTest() {

this.gp = new Gestion\_Producto();

}

/\*\*

\* Anotacion que verifica la iniciacion de la prueba

\*/

@BeforeClass

public static void ejecutarAntes() {

logger = Logger.getLogger("TestAnotacionesJUnit");

logger.info("Se van a ejecutar los Tests…");

}

/\*\*

\* Anotacion que verifica la finalizacion de la prueba

\*/

@AfterClass

public static void ejecutarDespues() {

logger.info("Se han ejecutado los Tests…");

logger = null;

}

/\*\*

\* antes de la ejecucion del test se obtiene la lista para relacionarla con

\* el valor del total

\*/

@Before

public void obtenerTotal() {

gp.listar(gp.getLita());

total = 21500.0;

}

/\*\*

\* Metodo de testeo en donde se relaciona el valor esperado deltotal con el

\* entregado por el metodo totalPagos

\*/

@Test

public void testTotalVentasGeneral() {

System.out.println("----------" + total + "\n" + gp.totalPagos(gp.getLita()));

assertTrue(total == gp.totalPagos(gp.getLita()));

}

/\*\*

\* Metodo de testeo posterior al pincipal en donde se relaciona las ventas

\* de un dia, dando la fecha con en valor esperado

\*/

@Test

public void testTotalVentasPorDia() {

gp.listar(gp.reportePorDia("28/04/2020"));

total = 10000.0;

System.out.println("----------" + total + "\n" + gp.totalPagos(gp.reportePorDia("28/04/2020")));

assertTrue(total == gp.totalPagos(gp.reportePorDia("28/04/2020")));

}

/\*\*

\* Metodo de testeo para las ventas de una fecha no existente

\* en el sistema

\*/

@Test

public void testTotalVentasFechaNoExitente() {

assertTrue(0.0 == gp.totalPagos(gp.reportePorDia("30/04/2020")));

}

}

## Testeo

A continuación, se mostrarán las pruebas realizadas sobre los métodos de totalPagos y reporteDelDia, los cuales sirven para entregar el total de ventas en una fecha especifica y obtener la lista de ventas de una fecha determinada.

### TestTotalVentasGeneral

En este método de testeo se va a corroborar que el total de ventas esperado sea el mismo que entregue el sistema.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | total == gp.totalPagos(gp.getLita()) | **Gp**: clase de gestión de los productos  **totalPagos**: método que retorna un valor de coma flotante con la respectiva suma de ventas de una lista ingresada de productos vendidos.  **getLista**: método que retorna la lista total de ventas existente. | Total = 21500.0 | True |

#### Análisis

Los resultados del método arrojan que el método de totalPagos funciona como se esperaba, ya que entrega el valor esperado del total de ventas.

#### Consideraciones

No se requieren modificaciones en los métodos involucrados.

### TestTotalVentasPorDia

En este método de testeo se pretende comprobar que los métodos de totalPagos y reportePorDia funcionan correctamente, enfrentándolo con los valores esperados extraídos de la base de datos.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | total == gp.totalPagos(gp.reportePorDia(fecha)) | **Gp**: clase de gestión de los productos  **totalPagos**: método que retorna un valor de coma flotante con la respectiva suma de ventas de una lista ingresada de productos vendidos.  **reportePorDia**: método encargado de retorna una lista con las ventas de una fecha en especifico entregadas por el usuario. | String fecha = "28/04/2020";  Double total = 10000.0 | True |

#### Análisis

Los resultados del método de testeo corresponden con los esperados, viendo cómo responde el método de totalPagos responde a la lista de productos entregada por el método reportePorDia.

#### Consideraciones

No se requiere realizar modificación a los métodos testeados.

### TestTotalVentasFechaNoExitente

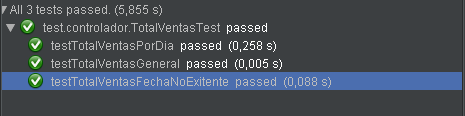
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | 0.0 == gp.totalPagos(gp.reportePorDia("30/04/2020")) | **Gp**: clase de gestión de los productos  **totalPagos**: método que retorna un valor de coma flotante con la respectiva suma de ventas de una lista ingresada de productos vendidos.  **reportePorDia**: método encargado de retorna una lista con las ventas de una fecha en específico entregadas por el usuario. | String fecha = "30/04/2020" | True |

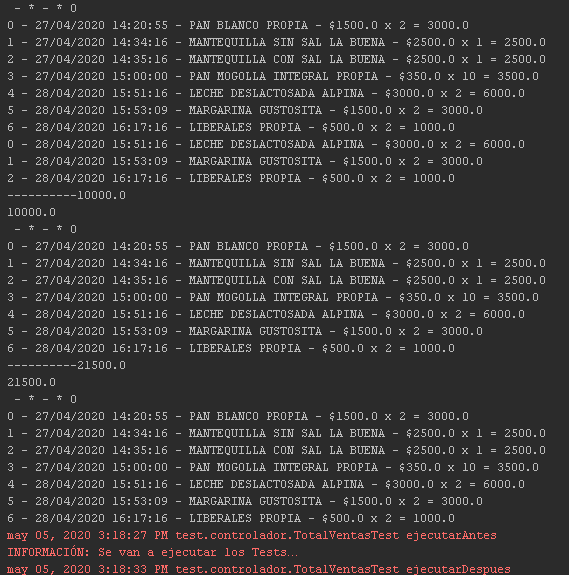
#### Análisis

Ya que la fecha ingresada no se relaciona con ninguna venta realizada, se corrobora que, tanto el método de reportePorDia retorna una lista vacía sin elementos, correspondiente a que ese día no se realizaron ventas, y que el método totalPagos reacciona de forma correcta con una lista vacía, retornado como total 0.0.

#### Consideraciones

No se requieren cambios en los métodos involucrados.





# Testeo del DAO de productos

Se requiere comprobar el correcto funcionamiento de los métodos ejecutores de las sentencias SQL para la extracción y gestión de los datos contenidos en la base de datos.

Para esto se requiere comprobar las siguientes actividades:

* Inserción de productos en la base de datos
* Eliminación de productos de la base de datos
* Modificación o actualización de los datos de los productos en la base de datos.
* Búsqueda de todos los registros de los productos en la base de datos.
* Búsqueda de los registros de venta de productos de una fecha en específico.
* Búsqueda de las ventas de un producto por su nombre

public class DaoProductoTest {

static Logger logger = null;

DAO\_Producto dp;

Producto pr, pm;

Gestion\_Producto gp;

String busqueda;

/\*\*

\* Inicializacion de los objetos necesarios para la prueba

\*

\* @param dp: objecto con los metodos para la manipulacion de la base de

\* datos

\* @param gp: objeto con metodos de gestion de los productos

\* @param pr: objecto tipo producto el cual tendra los datos para ser

\* manipulados

\* @param pm: objeto clon de pr usado para la mosificacion.

\* @param busqueda: cadena de texto con el nombre del producto para ser

\* manipulado

\*/

public DaoProductoTest() {

this.dp = new DAO\_Producto();

this.gp = new Gestion\_Producto();

this.pr = new Producto("PAN INTEGRAL", "05/06/2020", 1500.00, 2, "PROPIO", gp.fechaActual(), 3000.00);

this.pm = this.pr;

this.busqueda = pm.getNombre();

}

/\*\*

\* Metodo de testeo de la insercion de datos a la base de datos

\*/

@Test

public void insertTest() {

assertTrue(dp.crearProducto((Producto) pr));

assertTrue(!dp.crearProducto((Producto) pr));

}

/\*\*

\* metodo de testeo de la eliminacion de un producto de la base de datos

\*/

@Test

public void deleteTest() {

assertTrue(dp.eliminarProducto(pr.getNombre(), "05/05/2020"));

}

/\*\*

\* Metodo de testeo de la modificacion de productos de la base de datos

\*/

@Test

public void updateTest() {

String nombre = "Pan Arabe", fechaV = "12/05/2020";

double valor = 2500.00;

pm.setNombre(nombre.toUpperCase());

pm.setCosto(valor);

pm.setFechaVencimiento(fechaV);

pm.setTotal(valor \* pm.getCantidad());

assertTrue(dp.editarProducto(pm, busqueda));

}

/\*\*

\* Metodo de testeo de la busqueda de los objetos en la base de datos

\*/

@Test

public void buscarTodosTest() {

assertTrue(!dp.buscarTodosLosProductos().isEmpty());

gp.listar(dp.buscarTodosLosProductos());

}

/\*\*

\* Metodo de testeo del reporte de ventas de productos en una fecha

\* especifica

\*/

@Test

public void reporteFechaTest() {

assertTrue(!dp.buscarVentasDia("28/04/2020").isEmpty());

gp.listar(dp.buscarVentasDia("28/04/2020"));

assertTrue(dp.buscarVentasDia("30/04/2020").isEmpty());

}

/\*\*

\* Metodo de testeo de la busqueda de un producto basandose en su nombre

\*/

@Test

public void xbuscarPorNombreTest() {

assertNotNull(dp.buscarProductoPorNombre(busqueda).get(0));

System.out.println(dp.buscarProductoPorNombre(busqueda).get(0).toString());

}

/\*\*

\* Metodo de testeo por medio del cual se realiza una consulta a la tabla

\* log en la base de datos, por medio de la busqueda del ultimo registro

\* ingresado en la tabla por medio del trigger

\*/

@After

public void ReporteModificacionDB() {

String acceso = dp.accesosALaBase();

System.out.println(acceso);

assertNotNull(acceso);

}

}

## Testeo

A continuación, se presentas los resultados de los diferentes métodos de testeo implementados sobre a clase DAO\_Producto y los métodos en esta capaces de ejecutar sentencias SQL sobre la base de datos creada.

### InsertTest

En este método de testeo del método de inserción de productos en la base de datos, el cual se espera evítelos registros duplicados.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | dp.crearProducto((Producto) pr) | **crearProducto**: este metodo se encarga de ejecutar la sentencia SQL para la inserción de productos en la base de datos, evitando los registros duplicados. | Pr = new Producto("PAN INTEGRAL", "05/06/2020", 1500.00, 2, "PROPIO", gp.fechaActual(), 3000.00); | True |
| 2 | ¡dp.crearProducto((Producto) pr) |  |  | True |

#### Análisis

El método reacciono de la forma esperada, evitando los registros duplicados, basándose en la estructura de claves foráneas especificada en la base de datos.

#### Consideraciones

No se requieren modificaciones en el método de inserción.

### DeleteTest

En este método se buscará corroborar la correcta eliminación del producto previamente indexado en la base de datos.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | dp.eliminarProducto(pr.getNombre(), "05/05/2020") | **eliminarProducto**: este metodo se encargará de la ejecución de la sentencia SQL para la eliminación de un producto basándose en la fecha de venta y el nombre del producto | Búsqueda =“PAN INTEGRAL"  Fecha = “05/05/2020” | True |
| 2 | ¡dp.eliminarProducto(pr.getNombre(), "05/05/2020") |  |  | True |

#### Análisis

El método se comportó de la forma esperada en donde se eliminan los registros relacionados con el nombre y la fecha ingresada, y valida la no existencia de productos asociados con dicho nombre y fecha si es requerido.

#### Consideraciones

No se requieren mejoras del método testeado.

### UpdateTest

En este método de testeo se buscará la correcta modificación del producto previamente indexado a la base de datos.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | dp.editarProducto(pm, busqueda) | **editarProducto**: En este metodo de retorno booleano se realiza la sentencia SQL para la modificación de los productos basándose en el nombre del producto | Búsqueda =“PAN INTEGRAL"  String nombre = "Pan Arabe", fechaV = "12/05/2020";  double valor = 2500.00; pm.setNombre(nombre.toUpperCase()); pm.setCosto(valor); pm.setFechaVencimiento(fechaV);  pm.setTotal(valor \* pm.getCantidad()); | True |

#### Análisis

La modificación del producto previamente indexado se realizo de la forma esperada acorde a los parámetros ingresados sin presentar ningún conflicto con la base de datos.

#### Consideraciones

No se requieren modificaciones al método testeado.

### BuscarTodosTest

En este método de testeo se pretende validar el correcto funcionamiento del método de extracción de todos los productos en la base de datos

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | !dp.buscarTodosLosProductos().isEmpty() && dp.buscarTodosLosProductos().size() == 7 | **buscarTodosLosProductos**: Este método se encarga de la extracción de todos los productos contenidos en la base de datos. | El método retorna una lista de objetos tipo Producto  También se valida el tamaño de esta lista | True |

#### Análisis

Se corroboro el correcto funcionamiento del método, además de la extensión de la lista esperada.

#### Consideraciones

No se requieren modificaciones a este método.

### ReporteFechaTest

Este método de testeo se encargará de corroborar que se genere el reporte de ventas de una fecha en específico, además de corroborar que no exista conflicto con la fecha ingresada y se retorne una lista errónea de ventas.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | !dp.buscarVentasDia("28/04/2020").isEmpty() | **buscarVentasDia**: Este metodo se encargará de retornar una lista de ventas relacionadas con una fecha | El método retorna una lista de objetos tipo Producto  En este caso se usa una fecha que corresponde a 3 ventas en la base de datos.  También se valida que la lista no esté vacía | True |
| 2 | dp.buscarVentasDia("30/04/2020").isEmpty() |  | En este caso se usa una fecha la cual no corresponde con ningún registro en la base de datos |  |

#### Análisis

Se corroboro que el método testeado reacciono de la forma esperada ya que las fechas usadas reaccionaron de la forma esperada,

#### Consideraciones

No se requieren modificaciones al método testeado.

### xBuscarNombreTest

En este método de testeo se pretende corroborar el correcto funcionamiento del método para obtener la ultima venta de un producto.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | dp.buscarProductoPorNombre(busqueda).get(0) | **buscarProductoPorNombre**: Este método se encarga de la extracción de la ultima venta realizada de un producto basándose en su nombre | El método retorna una lista de objetos tipo Producto  Utilizando solo el primer valor de la lista | True |

#### Análisis

El método reacciono de la forma esperada, retornando el objeto que se esperaba mediante el uso de la cadena de texto.

#### Consideraciones

No se requieren modificaciones al método

### ReporteModificacionDB

Este método de testeo que se ejecutara después de cada sentencia sobre la base de datos, por lo tanto, se espera que se le comunique al usuario la manipulación a la base de datos.

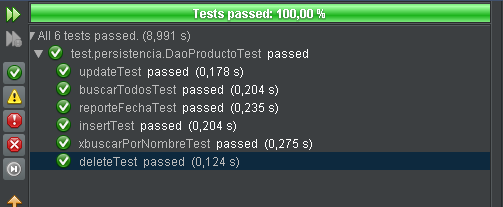
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba | Condición | Parámetros | Objeto | Resultado |
| 1 | dp.accesosALaBase() != null | **accesosALaBase**: Este metodo se encargará de realizar consultas a la tabla log en donde se almacena los reportes de manipulación a la base de datos. | Cadena de testo que retorna la fecha y tipo de modificación a la base de datos | True |
| “Dato eliminado en la tabla productos hecho 05/05/2020 19:38:13” | | | | |
| “Dato eliminado en la tabla productos hecho 05/05/2020 19:38:13” | | | | |
| “Dato eliminado en la tabla productos hecho 05/05/2020 19:38:13” | | | | |
| “Dato ingresado en la tabla productos hecho 05/05/2020 19:58:56” | | | | |
| “Dato ingresado en la tabla productos hecho 05/05/2020 19:58:56” | | | | |
| Dato eliminado en la tabla productos hecho 05/05/2020 19:59:03 | | | | |

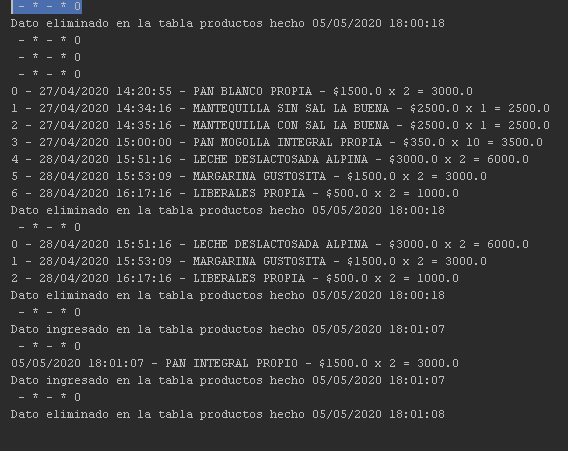
#### Análisis

El método reacciona de la forma esperada, sin generar conflictos al momento de ejecutar sentencias de modificación sobre los datos contenidos en la base de datos.

#### Consideraciones

No se requieren modificaciones al método testeado





# ANALISIS GENERAL

Al realizar los distintos tests sobre los métodos encargados de la gestión de los datos, tanto para su uso en la parte de visualización del usuario como para la gestión de estos en la base de datos, se observa que todos los métodos funcionan de la forma correcta, relacionándose de la forma indicada con la base de datos, la cual es el recurso que generalmente genera más conflictos debido a las sentencias y correcta escritura.