# **PROGRAMACIÓN ORIENTADA A OBJETOS**

**Herencia e interfaces ADEMAS Java desde consola 2024-1**

**Laboratorio 3/6**

**Integrantes: Daniel Aldana, Santiago Avellaneda**

# **OBJETIVOS**

Desarrollar competencias básicas para:

1. Aprovechar los mecanismos de la herencia y el uso de interfaces.
2. Organizar las fuentes en paquetes.
3. Usar la utilidad jar de java para entregar una aplicación.
4. Extender una aplicación cumpliendo especificaciones de diseño, estándares y verificando su corrección.
5. Vivenciar las prácticas XP : Code must be written to agreed [standards](http://www.extremeprogramming.org/rules/standards.html). Code the [unit test first](http://www.extremeprogramming.org/rules/testfirst.html).
6. Utilizar los programas básicos de java (javac, java, javadoc, jar), desde la consola.

# **ENTREGA**

* 1. Incluyan en un archivo .zip los archivos correspondientes al laboratorio. El nombre debe ser los dos apellidos de los miembros del equipo ordenados alfabéticamente.
  2. Deben publicar el avance al final de la sesión y la versión definitiva en la fecha indicada en los espacios preparados para tal fin.

**DESARROLLO**

## **Contexto**

Un sistema complejo es un sistema compuesto de agentes interrelacionadas que como un conjunto exhiben propiedades y comportamientos no evidentes a partir de la suma de las partes individuales. Las dinámicas del desarrollo son llenos de complejidades. Las características de los sistemas complejos (como la interdependencia, la diversidad y la adaptabilidad de los agentes), desafían los supuestos básicos de las teorías tradicionales. En este laboratorio vamos a explorar un jardín complejo.

**Conociendo** [En lab03.doc y garden.asta ]

1. En el directorio descarguen los archivos contenidos en garden.zip. Revisen el código: a) ¿Cuántos paquetes tiene? d) ¿Cuál es el propósito del paquete presentación? e) ¿Cuál es el propósito del paquete dominio?

RTA//: -Se tienen 2 paquetes, uno llamado “presentation” y el otro “domain”.

-El propósito del paquete presentación es mostrar una interfaz gráfica que permita simular un jardín.

-El propósito del paquete dominio es poder proporcionar una estructura de clases e interfaces las cuales nos permitan modelar y simular un sistema en el que en nustro contexto se trata de un jardín.

1. Revisen el paquete de dominio, a)¿Cuáles son los diferentes tipos de componentes de este paquete? b) ¿Qué implica cada uno de estos tipos de componentes?

RTA//: - Tenemos en el paquete Dominio:

* + como interfaces: Thing
  + como clases absrtactas: Agent
  + como clases concretas: Flower, Water, Garden

1. Revisen el paquete de presentación, a) ¿Cuántos componentes tiene? b) ¿Cuántos métodos públicos propios (no heredados) ofrece?

RTA//: Para el paquete Presentación:

* como clase concreta: GardenGUI
* Tiene los siguientes metodos propios:
* getGarden
* main
* Para la clase concreta: PhotoGarden
* PhotoGarden
* paintComponent

1. Para ejecutar un programa en java, ¿Qué método se debe ejecutar? ¿En qué clase se encuentra?

Rta//: Para poder ejecutar el programa, nos debemos ubicar en el paquete presentación, en la clase GardenGUI y ejecutar el método main.

1. Ejecuten el programa. ¿Qué funcionalidades ofrece? ¿Qué hace actualmente? ¿Por qué?

Rta:// Ofrece un botón llamado Tic-tac el cual todavía no está implementado, lo que hace hasta el momento es permitir visualizar una matriz.

## **(Deben ejecutar la aplicación java, no crear un objeto como lo veníamos haciendo)**

**Arquitectura general.** [En lab03.doc y garden.asta]

1. Consulte el significado de las palabras package e import de java. ¿Qué es un paquete? ¿Para qué sirve? ¿Para qué se importa? Explique su uso en este programa.

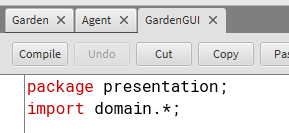
Package: Se usa para decir que un componente pertenece a un paquete.

Import: Sirve para poder importar componentes pertenecientes a un paquete.

Un paquete es un contenedor donde permite guardar diferentes componentes software, como clases concretas, clases abstractas, interfaces, con relación entre sí.

Su funcionalidad es modular y organizar la estructura de un proyecto de manera jerárquica.

Para acceder a los elementos, que pueden estar en otro paquete.

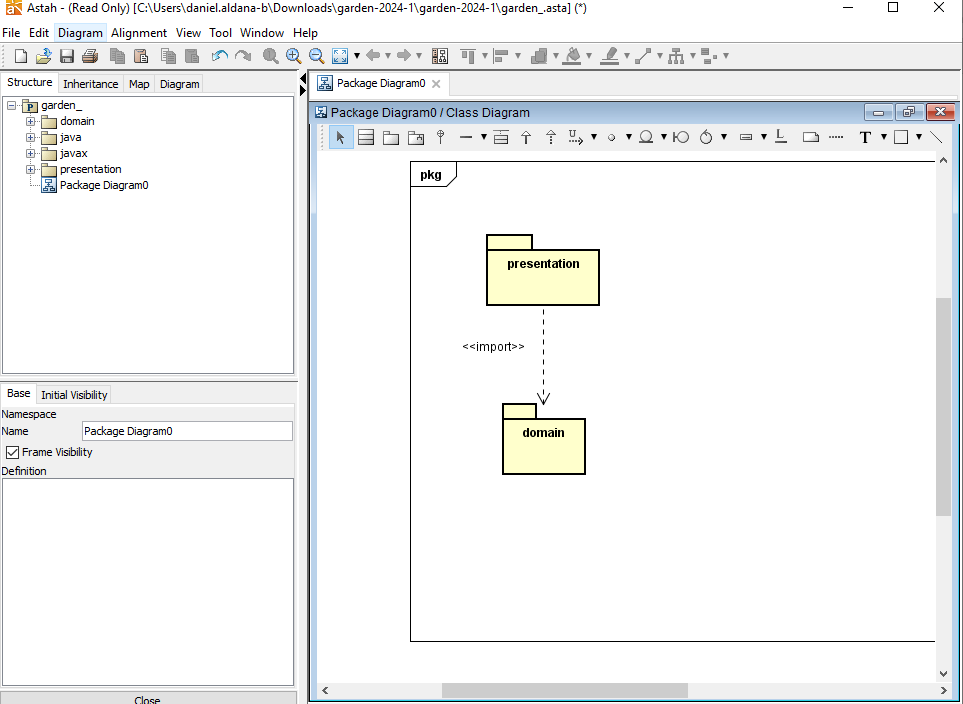


En el programa se puede evidenciar la funcionalidad tanto de los package e import, en este caso nos ubicamos en la clase GardenGui en el que vemos como se dice que esta clase va a pertenecer a el paquete “presentation” y vemos que importa todos los componentes del paquete domain.

1. Revise el contenido del directorio de trabajo y sus subdirectorios. Describa su contenido. ¿Qué coincidencia hay entre paquetes y directorios?

En el directorio, podemos encontrar tres carpetas y el archivo BlueJ, las carpetas son: doc, que se crea por defecto para la documentación del código; presentation y domain, ambas haciendo referencia a los dos paquetes creados en código, usando el mismo nombre. Allí dentro se encuentran los componentes de cada paquete respectivamente.

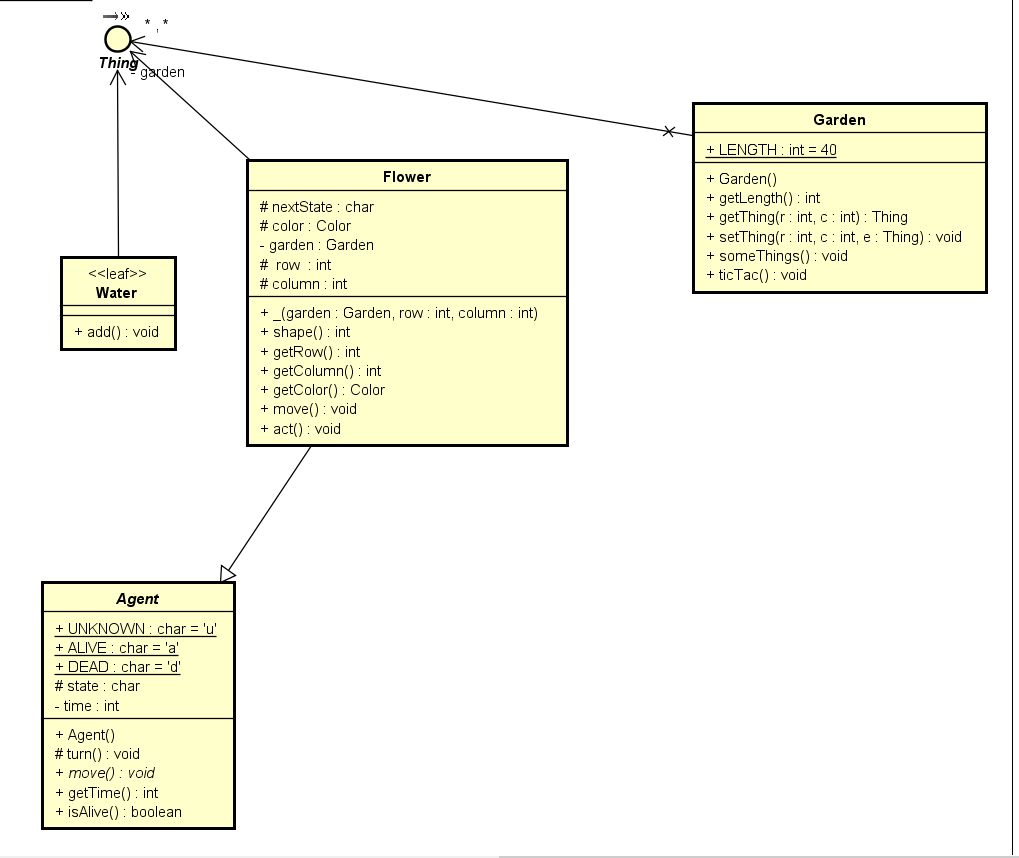
1. Adicione al diseño la arquitectura general con un diagrama de paquetes en el que se presente los paquetes y las relaciones entre ellos. Consulte la referencia en moodle. **En astah, crear un diagrama de clases (cambiar el nombre por Package Diagram0)**



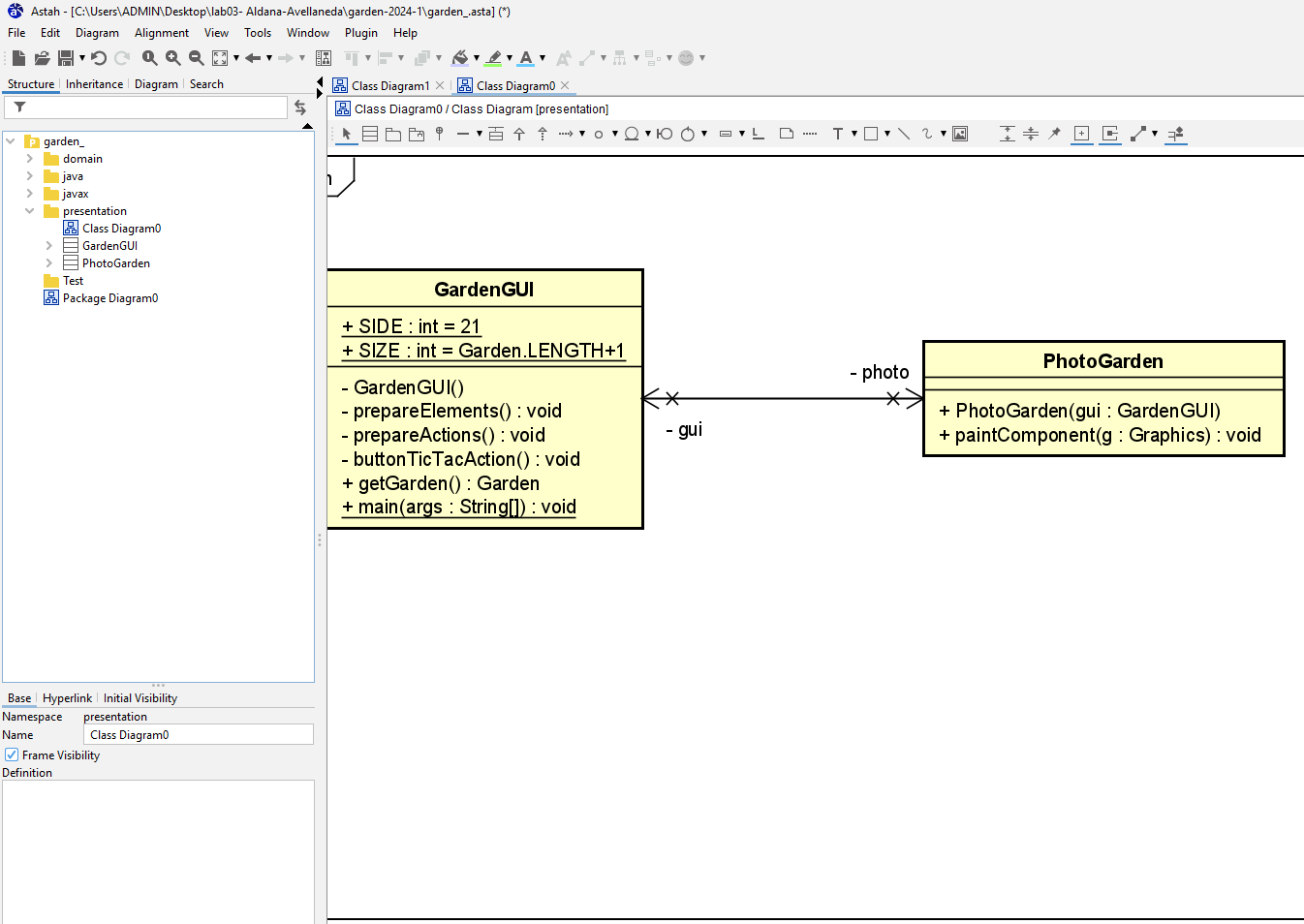
**Arquitectura detallada.** [En lab03.doc y gardenasta]

* 1. Para preparar el poyecto para **BDD.** Completen el diseño detallado del paquete de dominio. Adicionen el diagrama de clases en el paquete correspondiente. a) ¿Qué componentes hacían falta?

Rta//: Hacian falta las clases concretas Water y Flower.



* 1. Completen el diseño detallado del paquete de presentación. Adicionen el diagrama de clases al paquete correspondiente. a) ¿Por qué hay dos clases y un archivo .java?



* 1. Adicione la clase de pruebas unitarias necesaria para **BDD** en un paquete independiente de test**.** (No lo adicione al diagrama de clases) ¿Qué paquete debe usar? ¿Por qué? ¿Asociado a qué clase? ¿Por qué?

Rta//:

El paquete que vamos a usar es domain, debido a que esta relacionado con lo que se debe permitir y que no, mientras que presentation es mas para poder visualizar, teniendo en cuenta eso estaría asociada a la clase “Garden” esto porque es la clase principal del paquete.

Diagrama

Descripción generada automáticamente

**Ciclo 1. Iniciando con las flores normales [En lab03.doc y \*.java]**

# **(NO OLVIDE BDD – MDD)**

1. Estudie la clase garden ¿Qué tipo de colección usa para albergar cosas? ¿Puede recibir flores? ¿Por qué?

En la clase “Garden”, el tipo de colección que se esta utilizando para albergar cosas es, una matriz bidimensional de tipo Thing. Si revisamos un poco del código de las clases del paquete domain, podemos ver que hay 2 clases que implementan la interfaz Thing, por lo que en el contenedor se puede almacenar tanto objetos Water como de Flower, por lo que si puede recibir flores.

1. Estudie el código asociado a la clase Flower, ¿en qué estado se crea? ¿qué forma usa para pintarse? ¿cuándo aumenta su tiempo? ¿qué clases definen la clase Flower ? Justifique sus respuestas.

La clase Flower se crea con el estado inicial de nextState = Agent.Alive;, flower al heredar de Agent, lo que significa que al crear una instancia de FLOWER, se considera que esta viva.

1. Flower por ser un Agent, ¿atributos tiene? ¿qué puede hacer (métodos)? ¿qué decide hacer distinto? ¿qué no puede hacer distinto a todos los agentes? ¿qué debe aprendar a hacer? Justifique sus respuestas.

Protected permite que todas las subclases hereden los atributos de la clase padre, por lo que estaría heredando el atributo de protected char state;, heredado de su padre.

Los metodos que puede hacer es protected void turn(), public abstract void move(), public final int getTime(), public final Boolean isAlive().

No sobrescribe ningun metodo de su padre por lo que no decide hacer algo distinto.

No puede sobreescribir tanto los métodos de getTime y isAlive, ya que son final.

Al heredar de una clase abstracta y no definirse como clase abstracta entonces debe implementar todos los métodos abstractos de su padre por lo que en este caso debe aprender a hacer move.

1. Por comportarse como un Thing, ¿qué sabe hacer? ¿qué decide hacer distinto? ¿qué no puede hacer distinto? ¿qué debe aprender a hacer? Justifique sus respuestas.

Puede usar los metodos de shape(), getColor() e is().

Decide hacer distinto su getColor, ya que en el codigo se ve que no hace referencia a que va a utilizar el definido por la interfaz.

Debe implementar el metodo act(), ya que en la interfaz no esta implementada.

1. Ahora vamos a crear dos flores en diferentes posiciones (10,10) (15,15) llámelas rose y violet usando el método someThings() . Ejecuten el programa, ¿Qué pasa con las flores?¿Por qué? Capturen una palla significativa.

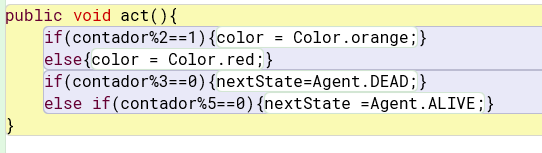
Aplicación, Tabla

Descripción generada automáticamente con confianza media

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

1. Diseñen, construyan y prueben el método llamado act() de la clase Flower. La flor cambia de color de rojo a naranja, se marchita después de 3 tic-tac y revive después de 5 tic-tac.



1. Diseñen, construyan y prueben el método llamado ticTac() de la clase garden.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

1. ¿Cómo quedarían rose y violet después de uno, dos, cuatro y seis **Tic-tac**? Ejecuten el programa. Capturen pantallas significativas en momentos correspondientes. ¿Es correcto?

Uno:

Gráfico

Descripción generada automáticamente con confianza media

Dos:

Una captura de pantalla de un celular

Descripción generada automáticamente con confianza media

Cuatro:

Una captura de pantalla de un celular con texto e imagen

Descripción generada automáticamente con confianza baja

Seis:

Gráfico, Gráfico de dispersión

Descripción generada automáticamente

**Ciclo 2. Incluyendo a las flores carnivoras** [En lab03.doc y gardenasta]

# **(NO OLVIDE BDD – MDD)**

El objetivo de este punto es permitir recibir en el jardín forles carnivoras . Ellas (i) son de color azul; (ii) se mueven hacia la flor más cercana, (iii) si pueden se la comen (ocupan su posición)

1. Para implementar esta nueva flor Carnivorous ¿cuáles métodos se sobre-escriben (overriding)?

Seria act, debido a que cada que se hagan click sobre el boton, esta nueva flor tendra otro comportamiento.

1. Diseñen, construyan y prueben esta nueva clase.
2. Adicione una pareja de inquietas, llámelas venus y sundeuos, (a) ¿Cómo quedarían después de tres **Tic-tac**? Ejecuten el programa y hagan tres clics en el botón. Capturen una palla significativa. (b) ¿Es correcto?

**Ciclo 3. Adicionando arena** [En lab03.doc, garden.asta y \*.java]

El objetivo de este punto es incluir en el garden zonas de arena (sólo vamos a permitir el tipo básico de arena) llos bancos de arena inician grises y a medida que pasa el tiempo van perdiendo su color.

# **(NO OLVIDE BDD – MDD)**

1. Construyan la clase Sand para poder adicionaría en el garden? ¿debe cambiar en el código del garden en algo? ¿por qué?
2. Diseñen , construyan el método y prueben esta nueva clase.
3. Adicionen dos parches de arena en las esquinas superiores del garden, llámenlos tatacoa y sahara, (a) ¿Cómo quedarían después de cuatro **Tic-tac**? Ejecuten el programa y hagan cuatro clics en el botón. Capturen una palla significativa. (b) ¿Es correcto?

## **Ciclo 4. Nueva flor: Proponiendo y diseñando**

El objetivo de este punto es permitir recibir en un nuevo tipo de Entity.

# **(NO OLVIDE BDD – MDD)**

1. Propongan, describan e Implementen el nuevo tipo de flores. (Mínimo dos pruebas de unidad)
2. Considerando una pareja de ellas con el nombre de ustedes. (a) Piensen una prueba significativa y expliquen la intención. (b) Codifiquen la pruebas de unidad correspondiente y capturen la palla de resultados. (c) Ejecuten el programa con esa prueba para capturar las pallas correspondientes.

## **Ciclo 5. Nueva entidad: Proponiendo y diseñando**

El objetivo de este punto es permitir recibir en una nueva entidad (no hormiga) en el

garden.

# **(NO OLVIDE BDD – MDD)**

1. Propongan, describan e Implementen un nuevo tipo de Entity. (Mínimo dos pruebas de unidad)
2. Considerando un par de ellos con los nombres de ustedes . (a) Piensen una prueba significativa y expliquen la intención. (b) Codifiquen la pruebas de unidad correspondiente y capturen la palla de resultados. (c) Ejecuten el programa con esa prueba para capturar las pallas correspondientes.

## **Ciclo 6. BONO. Flor Conway**

La flor Conway se rigue por las siguientes reglas:

* 1. Una flor muerta con exactamente 3 células vecinas vivas "revive" (al tiempo siguiente estará viva).
  2. Una flor viva con 2 ó 3 células vecinas vivas sigue viva.
  3. Si la flor tiene menos de dos o más de tres vecinas vivas muere o permanece muerta por "soledad" o superpoblación".
  4. Si en el vecindario, hay una celda vacia rodeada por 3 células vivas “nace” una

nueva cédula (al tiempo siguiente estará viva).

Primero todas las flores toman la decisión de lo que pasará en el tiempo siguiente y luego la realizan. Para probarla, verifique que esta flor puede genere un patrón que parpadea.

X X

XXX X XXX X

x X

**Empaquetando la versión final para el usuario.** [En lab03.doc, garden.asta , \*.java, garden.jar]

1. Revise las opciones de BlueJ para empaquetar su programa entregable en un archivo .jar. Genere el archivo correspondiente.
2. Consulte el comando java para ejecutar un archivo jar. ejecutennlo ¿qué pasa?
3. ¿Qué ventajas tiene esta forma de entregar los proyectos? Explique claramente.

# **DE BLUEJ A CONSOLA**

En esta sección del laboratorio vamos a aprender a usar java desde consola. Para esto se va a trabajar con el proyecto del punto erior.

**Comandos básicos del sistema operativo** [En lab03.doc]

es de iniciar debemos repasar los comandos básicos del manejo de la consola.

* 1. Investiguen los comandos para moverse en la estructura de directorios: crear, borrar, listar su contenido y copiar o eliminar un archivo.
  2. Organicen un nuevo directorio con la estructura propuesta para probar desde allí su habilidad con los comandos de consola. Consulten y capturen el contenido de su directorio

garden

src

domain presentation test

* 1. En el directorio copien únicamente los archivos \*.java del paquete de aplicación .

Consulte y capture el contenido de src/domain

**Estructura de proyectos java** [En lab03.doc]

En java los proyectos se estructuran considerando tres directorios básicos.

garden

src bin docs

1. Investiguen los archivos que deben quedar en cada una de esas carpetas y la organización interna de cada una de ellas.
2. ¿Qué archivos debería copiar del proyecto original al directorio bin? ¿Por qué? Cópielos y consulte y capture el contenido del directorio que modificó.

**Comandos de java** [En lab03.doc]

1. Consulte para qué sirven cada uno de los siguientes comandos:

javac java javadoc jar

1. Cree una sesión de consola y consulte en línea las opciones de los comandos java y

javac. Capture las pallas.

1. Busque la opción que sirve para conocer la versión a que corresponden estos dos comandos. Documente el resultado.

**Compilando** [En lab03.doc]

1. Utilizando el comando javac, **desde el directorio raiz (desde garden con una sóla instrucción)**, compile el proyecto. ¿Qué instrucción completa tuvo que dar a la consola para compilar TODO el proyecto? Tenga presente que se pide un único comando y que los archivos compilados deben quedar en los directorios respectivos.
2. Revise de nuevo el contenido del directorio de trabajo y sus subdirectorios. ¿Cuáles nuevos archivos aparecen ahora y dónde se ubican?

**Documentando** [En lab03.doc]

1. Utilizando el comando javadoc, desde el directorio raiz, genere la documentación (API) en formato html, en este directorio. ¿cuál es el comando completo para generar esta documentación?
2. ¿Cuál archivo hay qué abrir para empezar a navegar por la documentación? Ábralo y capture la palla.

**Ejecutando** [En lab03.doc]

4. Empleando el comando java, desde el directorio raiz, ejecute el programa. ¿Cómo utilizó este comando?

**Probando** [En lab03.doc]

1. Adicione ahora los archivos del directorio pruebas y trate de compilar nuevamente el programa.Tenga en cuenta que estas clases requieren la librería junit 4.8. ¿Cómo se incluye un paquete para compilar? ¿Qué instrucción completa tuvo que dar a la consola para compilar?
2. Ejecute desde consola las pruebas . ¿Cómo utilizó este comando?. Puede ver ejemplos de cómo ejecutar el“test runner”en [How to run JUnit test cases from the command line](https://stackoverflow.com/questions/2235276/how-to-run-junit-test-cases-from-the-command-line)
3. Pegue en su documento el resultado de las pruebas

**Empaquetando** [En lab03.doc]

1. Consulte como utilizar desde consola el comando jar para empaquetar su programa entregable en un archivo .jar, que contenga los archivos bytecode necesarios (no las fuentes ni las clases de prueba), y que se pueda ejecutar al instalarlo en cualquier directorio, con solo tener la máquina virtual de java y su entorno de ejecución (JRE).

¿Cómo empaquetó jar ?

1. ¿Cómo se ejecuta el proyecto empaquetado?

# **RETROSPECTIVA**

1. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes? (Horas/ Hombre)
2. ¿Cuál es el estado actual de laboratorio? ¿Por qué? (Para cada método incluya su estado)
3. Considerando las prácticas XP del laboratorio de hoy ¿por qué consideran que son importe?
4. ¿Cuál consideran fue su mayor logro? ¿Por qué? ¿Cuál consideran que fue su mayor problema? ¿Qué hicieron para resolverlo?
5. ¿Qué hicieron bien como equipo? ¿Qué se comprometen a hacer para mejorar los resultados?