Programación 3

TEMA 1: Entorno - Java Development Kit (JDK) / IDE ECLIPSE

Práctica nº 1

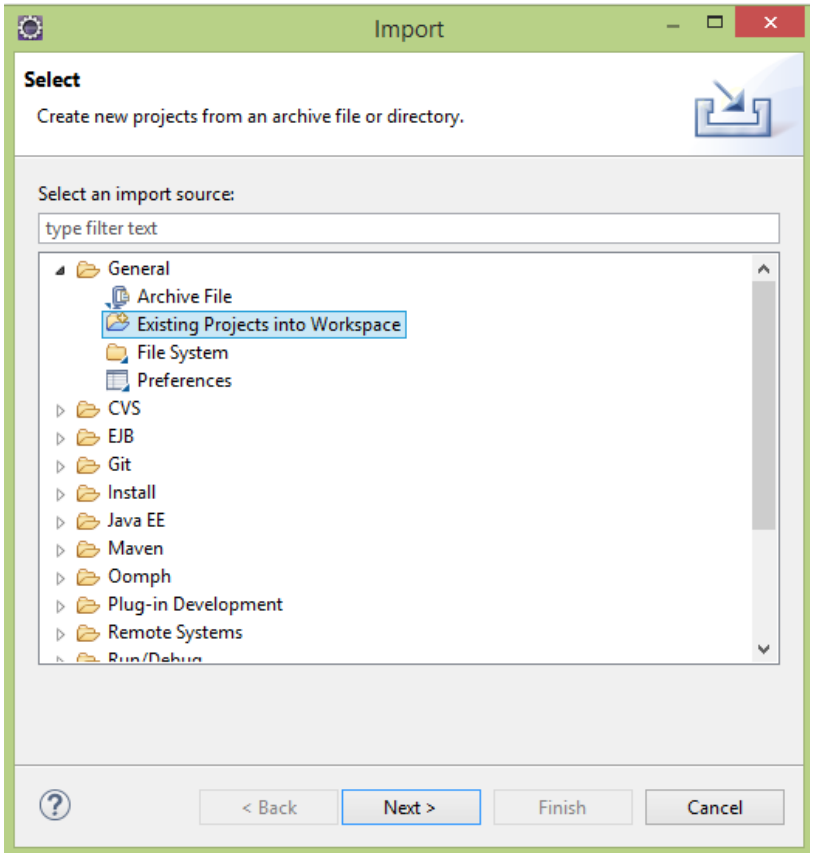
Para iniciar las prácticas debería tener instalado el JDK y el entorno eclipse.

- Descargue la Plataforma JAVA (Edición Estándar o Java SE) desde la URL https://www.oracle.com/java/technologies/javase-downloads.html e instale. NOTA: **sólo** se usarán características estándar del lenguaje, si bien se puede usar JDK 8+.
- Descargue Eclipse desde la URL http://www.eclipse.org/downloads/eclipse-packages/ . Seleccione la versión Eclipse IDE for Java Developers.
- Descomprima el archivo descargado.

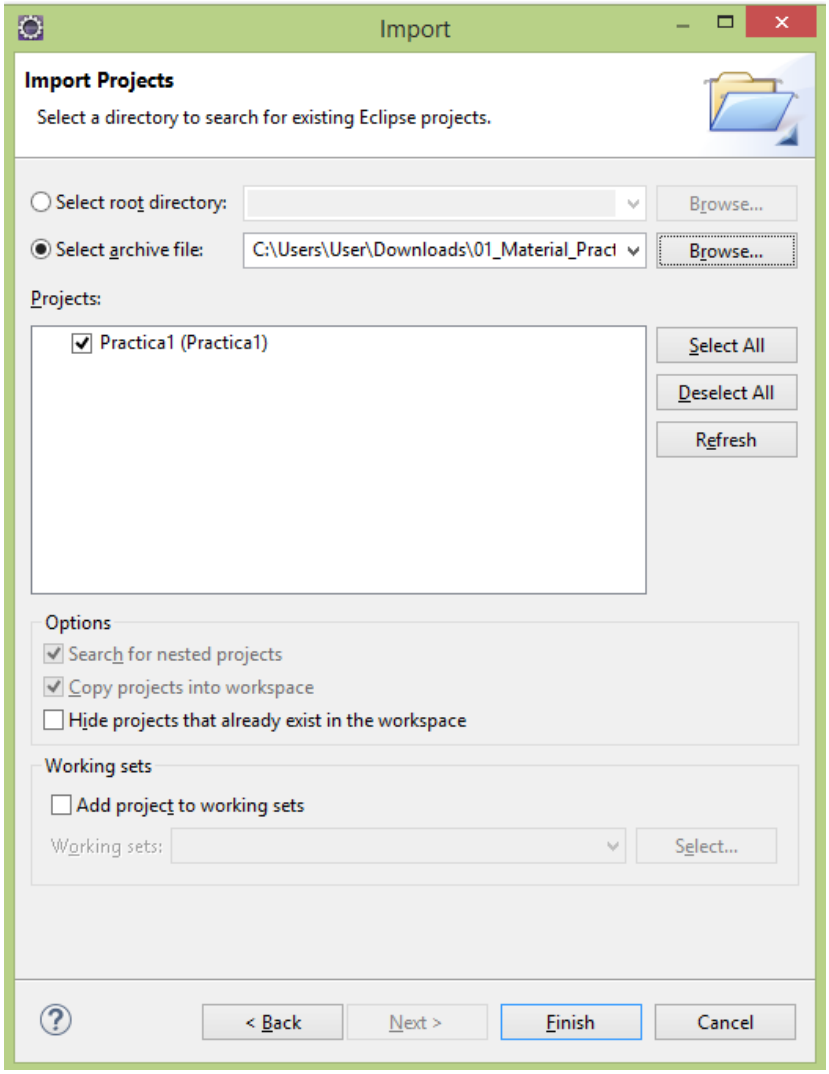
1. Importación de proyectos en eclipse

a. Descargue el archivo 01_Material_Practica_Tutorial_de_Eclipse.zip
b. Desde la barra de menú, seleccione *File -> Import …*
c. Seleccione *Existing Projects into Workspace* y presione *Next*.

*Eclipse organiza el código fuente en **proyectos** con el propósito de facilitar el desarrollo de aplicaciones. Esto **NO** es un requerimiento del lenguaje Java*



i. Luego seleccione la opción "Select Archive File:" e indique la ruta hasta el archivo 01_Material_Practica_Tutorial_de_Eclipse.zip. Presione el botón *Select All* para importar todos los proyectos dentro de ese archivo .zip como muestra la figura y presione *Finish*.



NOTA: si descomprime el archivo de la cátedra, encontrará un archivo .project, este archivo es un descriptor del proyecto y es por este motivo que eclipse permite su importación como "Proyecto Existente".

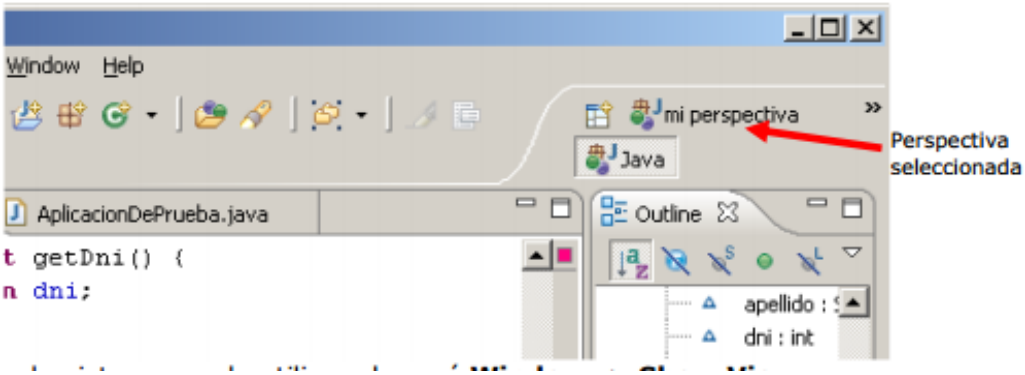
d. Propiedades del PROYECTO
   i. Usando el botón derecho del mouse, y sobre el nombre del proyecto anterior, seleccione la opción "*Properties*"
   ii. Seleccione en el margen izquierdo la opción "Resource".
      1. **Responda:** ¿Cuál es la ubicación de su proyecto Practica1?
   iii. Ahora seleccione en el margen izquierdo la opción "Java Compiler".

1. **Responda:** ¿Qué versiones de compilación tiene disponibles?
2. **Responda:** ¿a qué versión de Java se está compilando este proyecto en eclipse?

iv. Ahora seleccione en el margen izquierdo la opción "Java Build Path". Vaya al TAB "Libraries".

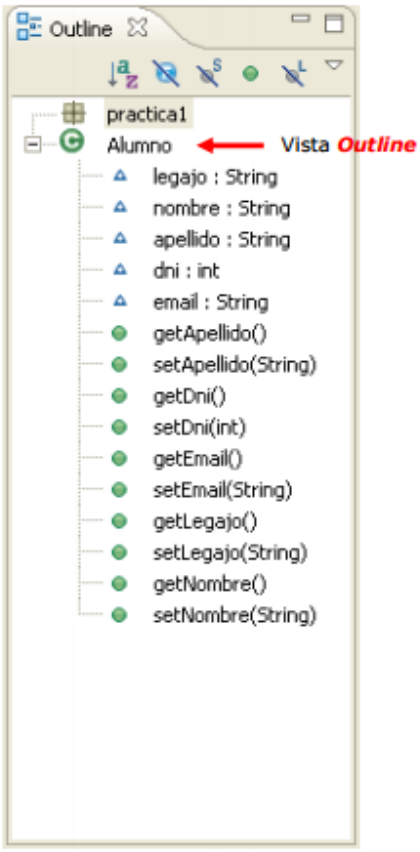
1. **Responda:** ¿Qué librería/s tiene disponible/s?

2. Usando **VISTAS y PERSPECTIVAS.**
Eclipse utiliza el concepto de *vistas* para permitirnos explorar y modificar los programas, examinar resultados, depurar código, etc. Las perspectivas *son organizaciones visuales de una selección de esas vistas*. Existen perspectivas preestablecidas y también es posible crear nuevas personalizadas. Eclipse muestra por defecto la **perspectiva Java**.
Para cambiar de perspectiva se puede utilizar los botones del margen superior derecho, o desde el menú *Window -> Open Perspective*



Para cambiar de vista se puede utilizar el menú *Window -> Show View*

a. **Responda:** ¿Qué otras perspectivas puede abrir?
b. **Responda:** ¿Qué vistas puede ver en la perspectiva Java?
c. **Explorando paquetes y editando código en el entorno**
   i. En la vista **Package Explorer** expanda el paquete *practica1* para inspeccionar su estructura. Cada paquete contiene uno o más archivos Java.
   ii. Busque la clase *Alumno.java* y haga doble click para abrirlo en el editor java. Observe el código fuente: estructura de la clase, palabras claves, etc.
   iii. Seleccione la vista **Outline**. Observe como muestra el paquete al cual pertenece la clase, los paquetes que importa, las declaraciones, miembros, métodos.



iv. El código fuente puede editarse completamente o visualizando sólo un método determinado. Seleccione en el editor la clase **Alumno.java** y seleccione en la vista **Outline** un método determinado, haciendo doble clic sobre el mismo se accede directamente a la definición del mismo. Presione el botón **Sort**, para ver ordenado alfabéticamente los elementos de la vista Outline.

v. Sobre la clase Alumno, presione la combinación de teclas Ctrol+o ¿Qué es lo que visualiza?

vi. Presione la combinación de teclas Ctrol+Shift+t ¿Qué funcionalidad tiene disponible?

vii. Cierre el archivo fuente **Alumno.java** y si hizo cambios **NO** los guarde.

d. Usando el ASISTENTE DE CÓDIGO

i. En la **perspectiva Java**, y desde la vista **Package Explorer** abra el archivo Alumno.java.

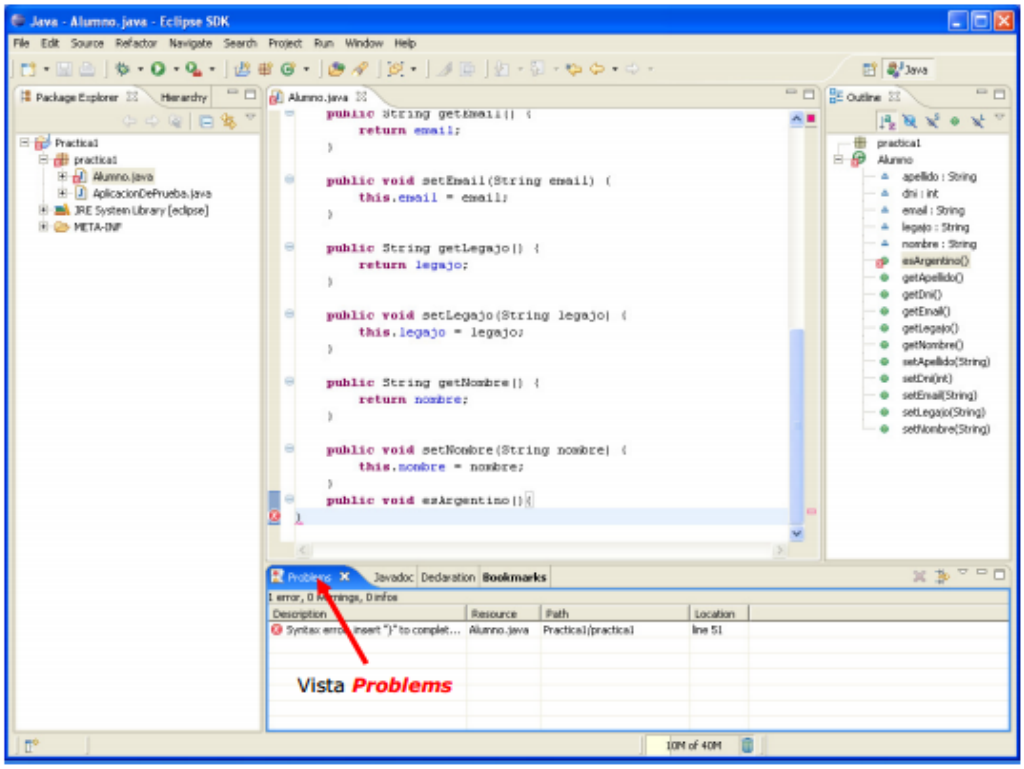
ii. Asegúrese de que el botón **Sort** (de la vista Outline), esté presionado para que muestre la vista ordenada alfabéticamente.

iii. Agregue en el código fuente, al final de la clase el siguiente código:

```
public void esArgentino(){
```

Observe como automáticamente el código es agregado en orden alfabético

iv. Presione el botón **Save**. Eclipse por defecto tiene habilitada la compilación automática cuando se salvan los fuentes. Los errores son mostrados en la vista Problems y en el editor con el símbolo:



e. **Reemplazo de un método desde el LOCAL HISTORY**

i. Continúe trabajando con la clase **Alumno.java**. En la vista **Outline**, seleccione el método *esArgentino()* que acaba de crear, edítelo y agregue una }.

ii. Luego abra el menú contextual y seleccione Delete y confirme.

iii. Agregue nuevamente un método al final de la clase con el siguiente código:

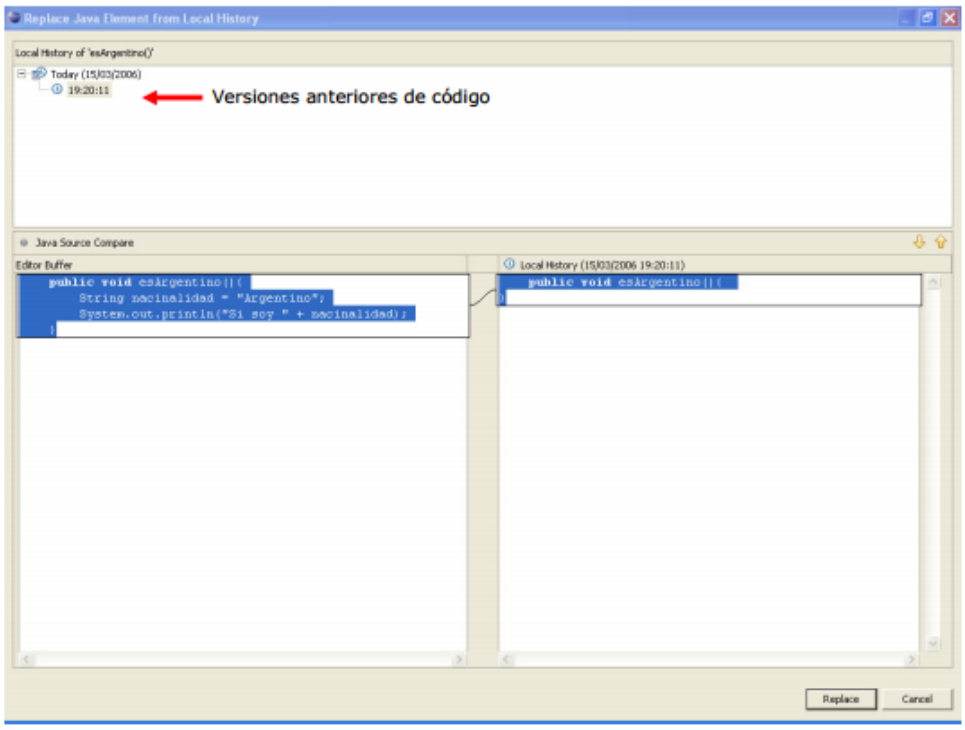
```
public void esArgentino(){
        String nacionalidad = "Argentino";
        System.out.println("Si soy " + nacionalidad);
}
```

iv. Guarde la clase.

v. En la vista **Outline**, seleccione el método *esArgentino()* que acaba de crear, abra el menú contextual y seleccione *Replace with -> Element from Local History…*.
Se abrirá una ventana con las **versiones anteriores** del método. Cada vez que se elige una versión en el panel superior, los paneles

inferiores muestran la versión actual comparándola con la seleccionada en el panel superior.

vi. Acepte presionando el botón **Replace**. El código será reemplazado.

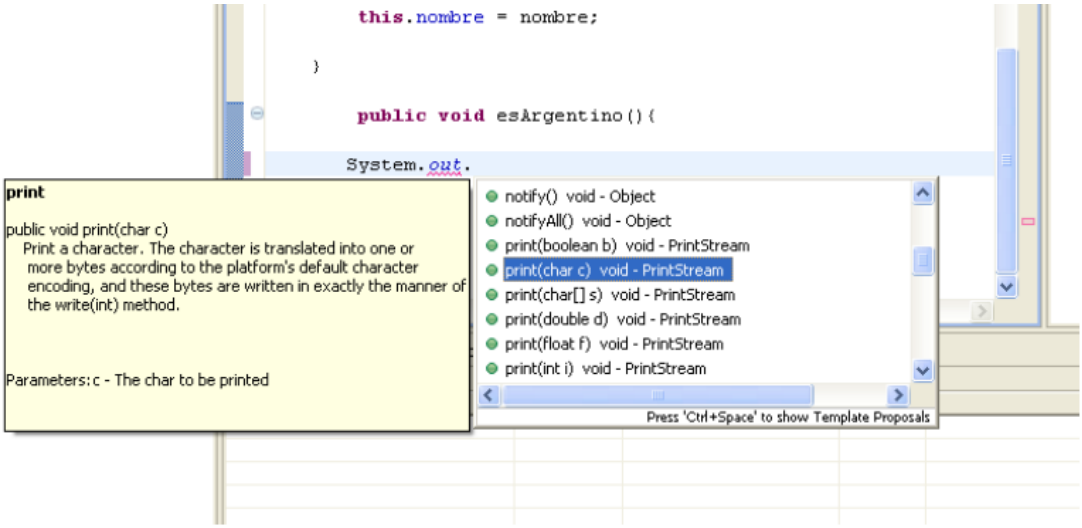


f. **Usando el asistente de contenido**

i. Continúe trabajando con la clase **Alumno**. En la vista **Outline**, seleccione el método esArgentino(). Agregue las siguientes líneas en el método seleccionado:

```
System.out.
```

ii. Con el cursor al final de System.out. presione **CTRL+Barra espaciadora** para activar el asistente de código.



iii. Seleccione el método **print(char c)** y presione **ENTER**. Después de que el código es insertado complételo así:

```
System.out.print('A');
```

También se puede activar este asistente poniendo la referencia del objeto o una clase y ".”

iv. Guarde la clase.

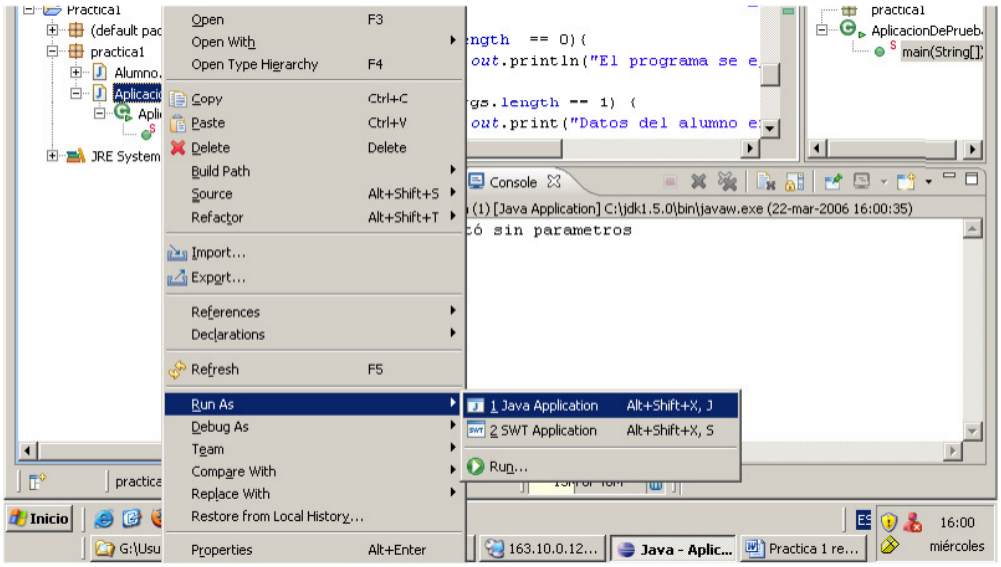
> **NOTA:** el editor autocompleta la instrucción System.out.println si ud. escribe **syso** y a continuación Ctrol + barra espaciadora.

3. Ejecutando un programa JAVA

Para ejecutar un programa Java es necesario que la clase contenga un método llamado **main** cuya declaración sea:

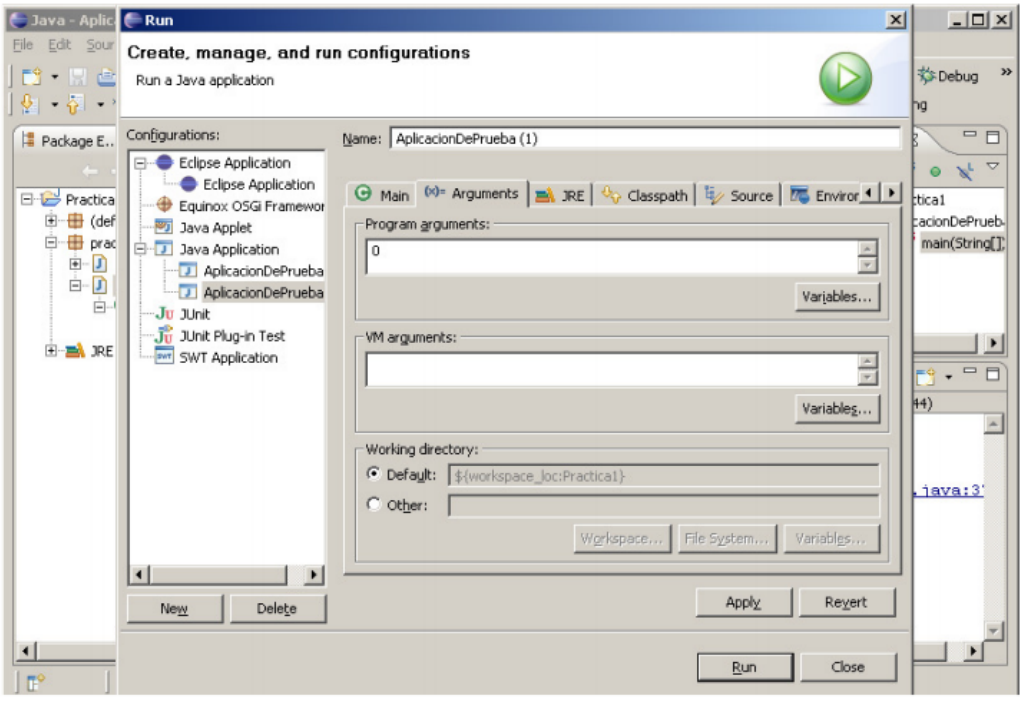
public static void main(String[] args) {}

a. En esta aplicación ejemplo, la clase que posee este método es AplicacionDePrueba. Ejecute la aplicación seleccionando la clase AplicacionDePrueba en la vista **Package Explorer** y con el botón derecho seleccione *Run As -> Java Application*. La aplicación crea 2 alumnos e imprime los datos personales del alumno especificado por un número de orden. Si no se especifica dicho número imprime *"El programa se ejecutó sin parámetros"*.

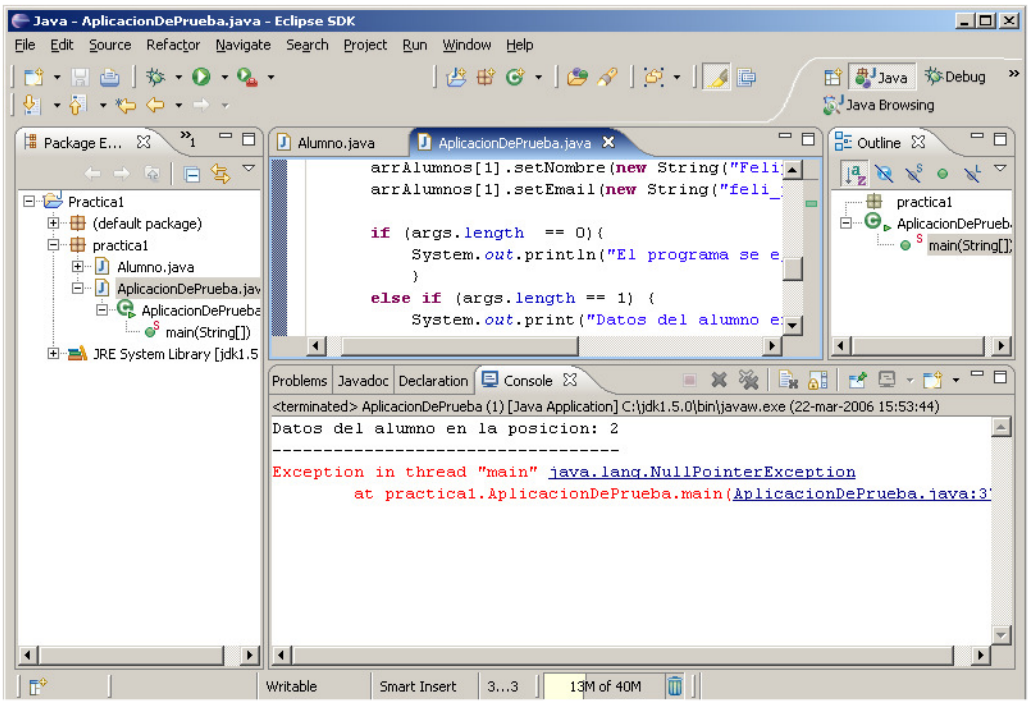


Java utiliza una vista llamada **Console** para dejar mensajes. Esta aplicación hace uso de ella para obtener una salida de forma sencilla.

b. Ejecute la aplicación pasándole como parámetro el número de orden igual a 0 ó 1. Para esto seleccione el menú *Run -> Run…* , luego seleccione la pestaña *Arguments* y en *Program Arguments* especifique 0 ó 1. El programa debería imprimir los datos personales del alumno seleccionado.



c. Pruebe de ejecutar el programa con un número de orden mayor a 2. Esto debería dar un error de ejecución.



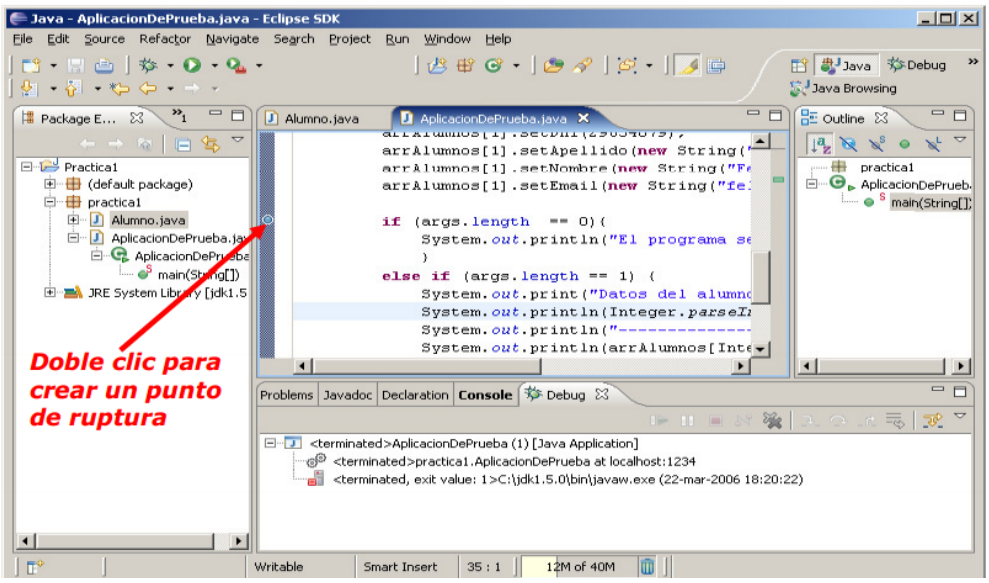
4. Usando el DEBUG

La forma más simple de detectar por qué falla una aplicación es haciendo uso del **DEBUG**. Eclipse provee una perspectiva llamada Debug para encontrar errores que puedan producirse durante la ejecución de un programa. Para esto se pueden establecer puntos de ruptura (Breakpoints) para que a partir de ahí se pueda ejecutar el programa paso a paso y examinar el estado de las variables y el flujo de ejecución que sigue el mismo.

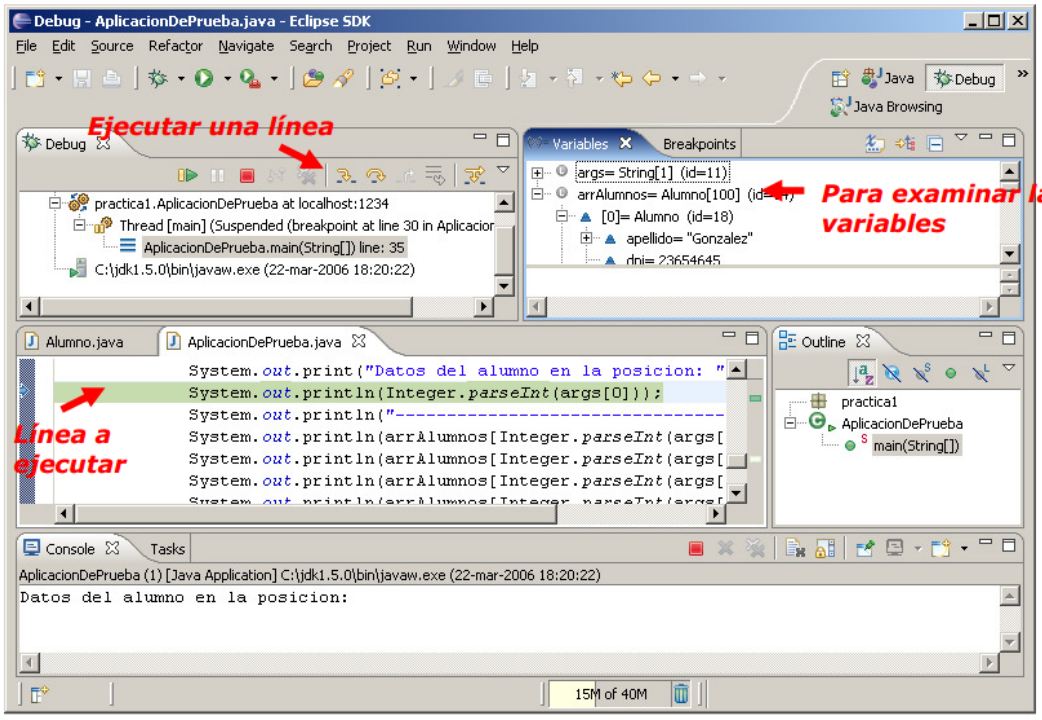
a. En el código de la clase AplicacionDePrueba coloque un punto de ruptura (breakpoint) en la línea que dice:

if (args.length == 0){

haciendo doble click en el margen izquierdo sobre esa línea.



b. Ejecute el debug seleccionando el menú *Run -> Debug…* deberá abrir la **perspectiva Debug**. Examine el contenido de las variables que contienen los datos de los alumnos y ejecute paso por paso el programa.



i. **Responda:** ¿qué tecla le permite realizar la opción "Step Over"?

ii. Vuelva a ejecutar el programa en modo Debug hasta llegar a la línea:

if (args.length == 0){

Observe que si pasa con el mouse y se detiene sobre *args* o *arrAlumnos* automáticamente el eclipse muestra el contenido de las mismas.

iii. Escriba en cualquier parte del editor: arrAlumnos.length (no guarde la clase). Seleccione lo recién escrito y con el botón derecho del mouse, seleccione la opción **Inspect**.¿Qué respuesta obtuvo?

NOTA: El correcto uso del Debug es imprescindible para que pueda detectar las causas de falla en su programa. Si observa las opciones del menú contextual de un breakpoint, puede incluso indicar condiciones asociadas al breakpoint, como por ejemplo: "que sólo se detenga la ejecución cuando una variable adopta el valor NULL"

5. Usando **eclipse**, cree un proyecto Java llamado **Prueba** y defina una clase llamada **Bienvenida** que imprima por consola el mensaje "Hola".
   a. **Responda:** ¿qué programa usó para compilar su código?
   b. **Responda:** ¿dónde se ubican los archivos resultantes de la compilación?
   c. Modifique la firma del método **main** por el siguiente:
      public static void main()
      Responda: ¿puede ejecutar su programa? **JUSTIFIQUE**
   d. Deshaga los cambios de modo que la firma del método main sea la original:
      public static void main(String[] args)
   e. Ahora su programa recibirá dos argumentos: *nombre* y *apellido*. Modifique la clase Bienvenida para que imprima "Hola " seguido por el nombre y apellido ingresados. Verifique que funciona correctamente.

6. Escriba una clase llamada **Temperatura** con un método llamado **calcularCelsius**, que dada una temperatura en Fahrenheit devuelva la misma en Celsius. La firma del método será:

   double calcularCelcius(double enFarenheit)

   Imprima por consola el resultado del método.
   CONVERSIÓN: *Para convertir valores Fahrenheit a Celsius, reste 32 al valor Fahrenheit, multiplique por 5 y divida entre 9. Asegúrese que el resultado obtenido es correcto (por ejemplo realizando los cálculos en forma manual o con una calculadora).*

NOTA: un código correctamente indentado es más legible, para una correcta indentación/formateo de su código, use la combinación de teclas Ctrol + Shift + F