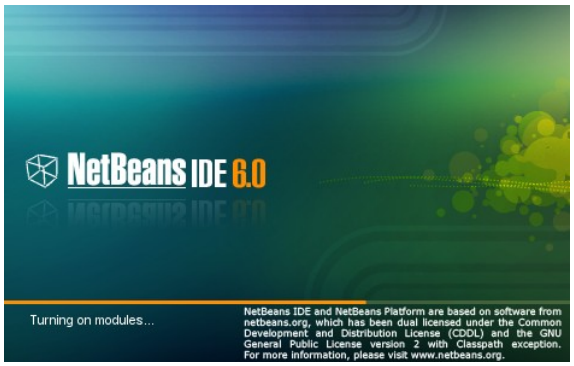
INSTALACIÓN E INTERFAZ GRÁFICA DE LA HERRAMIENTA DE PROGRAMACIÓN

NETBEANS 6.0



INSTALACION DEL NETBEANS 6.0

El IDE de NetBeans es gratuito, y de codigo abierto para desarrolladores de software. En esta version tienes al alcance de tu mano todas las herramientas necesarias para crear aplicaciones profesionales para entornos de escritorio, empresa, web y moviles, ya sea en C/C++, Java e incluso Ruby. El IDE ha sido desarrollado para distintas plataformas como Linux, MacOS X, Solaris y tambien Windows.

Pueden bajar el instalador haciendo click aquí desde la misma pagina web oficial del NetBeans o desde mi espacio en el servidor FTP de la universidad haciendo clickaquí **(Recomendado)**. Para poder llevar acabo la instalación es necesario tener instalado el JDK (Java Development Kit), en el archivo que han bajado desde mi espacio FTP esta incluido este archivo.

PROCESODEINSTALACION

Se necesita tener estos dos (2) programas almacenados en su computadora y realizar los siguientes pasos:

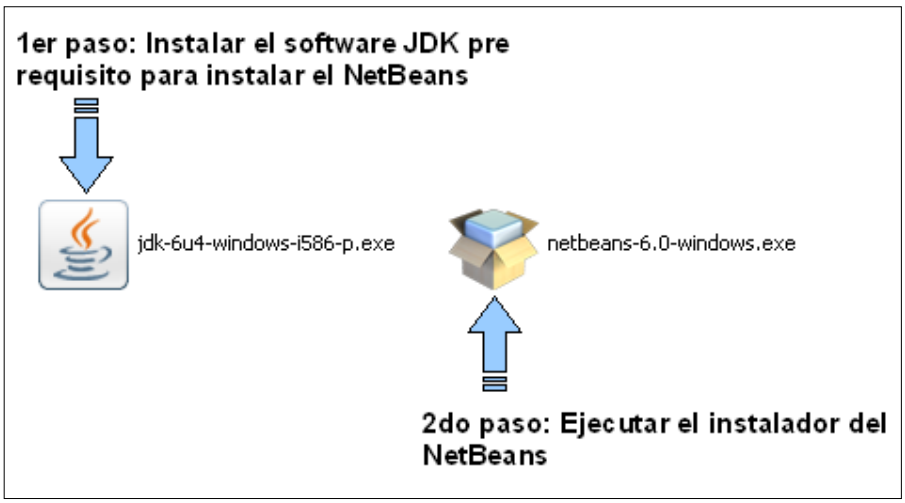


Imagen 1: Pasos para instalar el Netbeans

- **1erPASO:InstalarelsoftwareJDKprerequisitoparainstalarelNetBeans**

  El JDK (Java Development Kit) viene hacer el kit de desarrollo de Java, que es un conjunto de herramientas , utilidades, documentación y ejemplos para desarrollar aplicaciones Java. El que vamos a utilizar es el **jdk-6u4-windows-i586-p.exe**

  A continuación se presenta el proceso de instalación de este software.

  1ero: Doble click sobre el software jdk-6u4-windows-i586-p.exe

Imagen 2: Software JDK

2do. Esperar mientras se carga el *Asistente de Instalación*.

Imagen 3: Asistente de instalación

3ero. Click en el boton aceptar el acuerdo de licencia (en ingles).

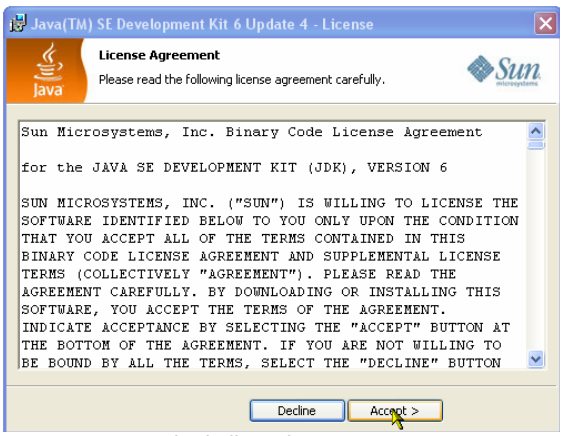


Imagen 4: Acuerdo de licencia

4to. Click en el boton next (siguiente). No modificar nada.

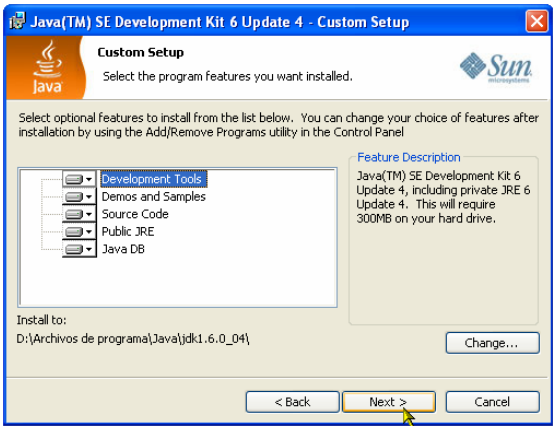


Imagen 5: Programas del kit de desarrollo de java que serán instalados

5to. Esprerar mientras se esta instalando el kit de desarrollo de java

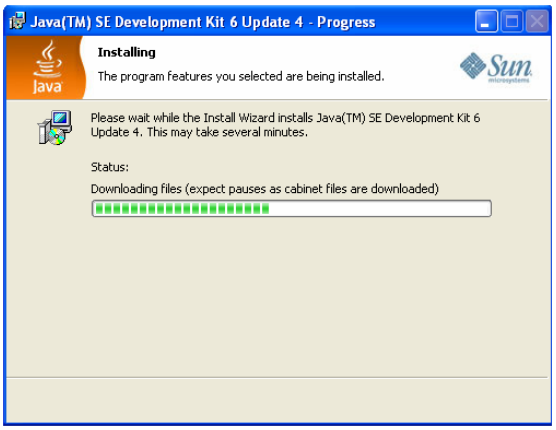


Imagen 6: Instalando el kit de desarrollo de java

6to. Click en el boton next (siguiente). No modificar nada.

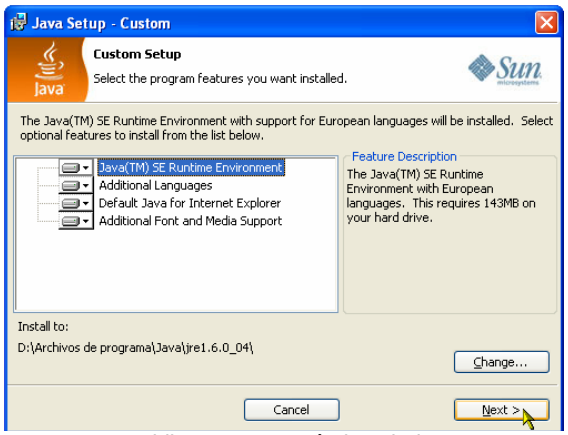


Imagen 7: Los idiomas que serán instalados

7mo. Esperar mientras se termina el proceso de instalación.

Imagen 8: Instalación en proceso

8vo. Finalización de la instalación, hacer click en el boton Finish

Imagen 9: Fin de la instalación

- **2doPASO:EjecutarelinstaladordelNetBeans**

La herramienta de programación NetBeans es software libre, es decir, no se necesita pagar licencia por tenerlo instalado en nuestra computadora.

A continuación se presenta el proceso de instalación del NetBeans

1ero: Doble click sobre el software netbeans-6.0-windows.exe

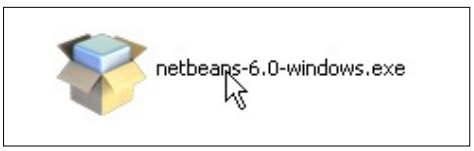


Imagen 10: Software NetBeans

2do. Esperar mientras se carga el *Asistente de Instalación*.

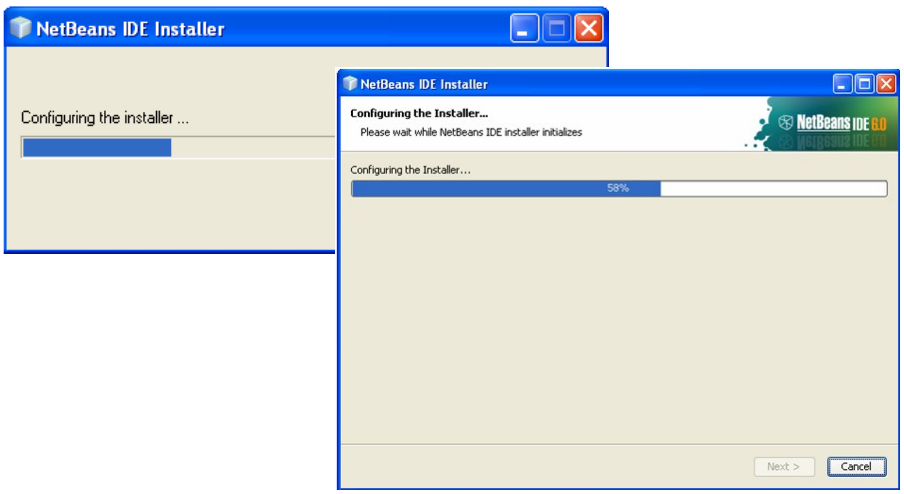


Imagen 11: Cargando el asistente de instalación

3ero. Click en el boton next (siguiente)

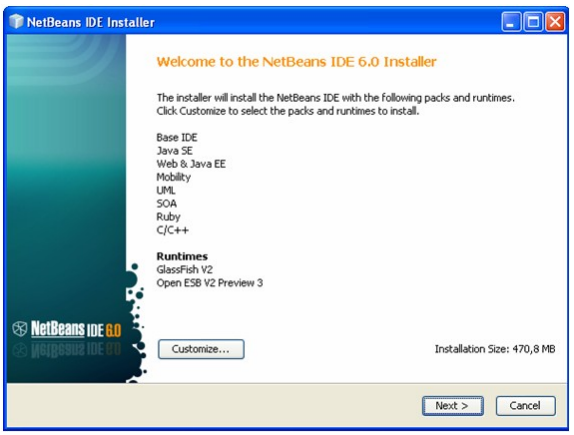


Imagen 12: Bienvenido al instalador del NetBeans

4to. Hacer click en el check para estar de acuerdo con la licencia y luego click en el boton next.

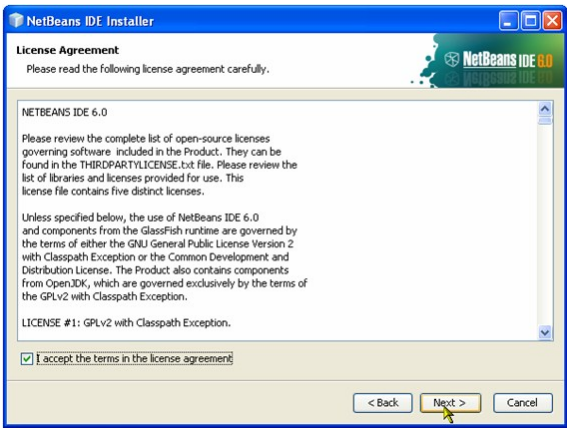


Imagen 13: Acuerdo de licencia

5to. Click en el boton next (siguiente). No modificar nada.

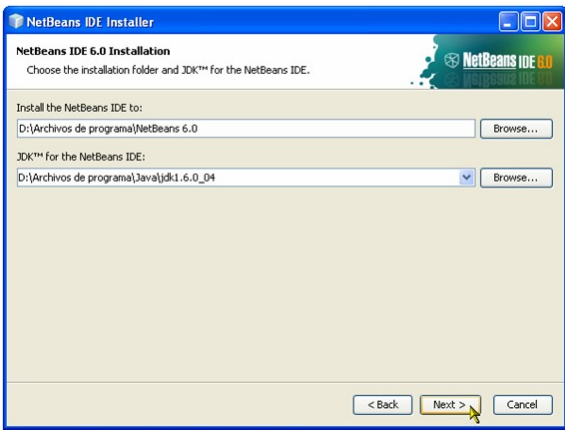


Imagen 14: Elección de las carpetas donde se instalará el NetBeans y JDK

6to. Click en el boton next (siguiente). No modificar nada.

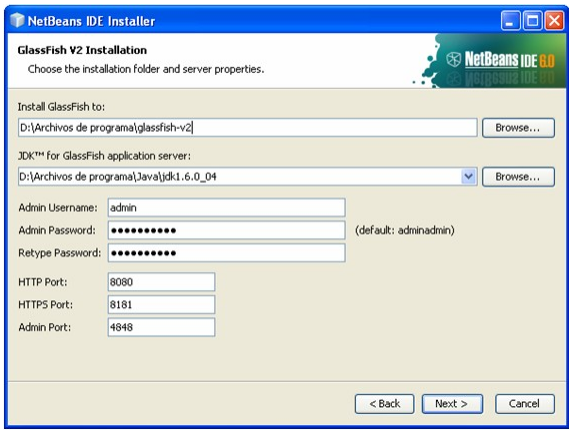


Imagen 15: Elección de las carpetas de instalación de servidores y propiedades

7mo. Click en el boton next (siguiente) para comenzar la instalación del NetBeans.

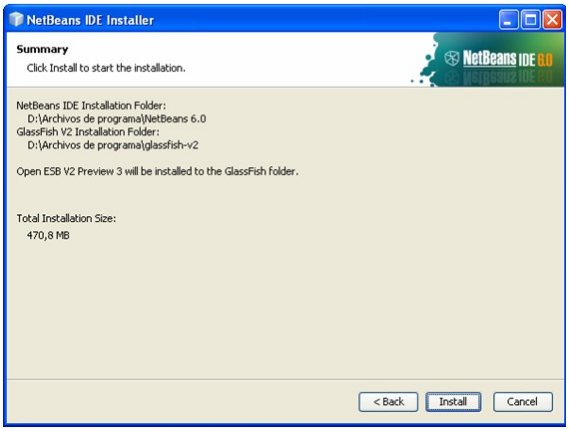


Imagen 16: Comienzo del proceso de instalación

8vo. Esperar mientras se termina el proceso de instalación.

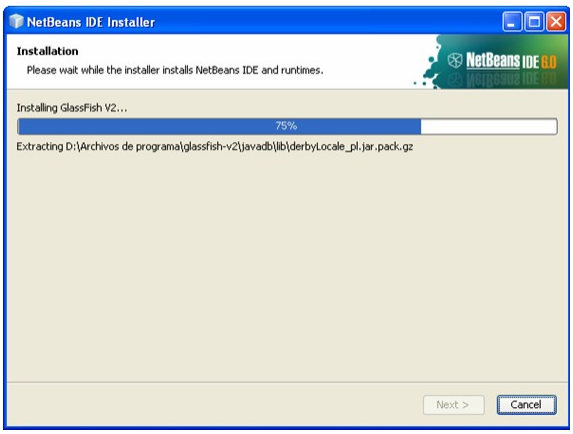


Imagen 17: Instalación en proceso

9no. Click en el boton finish (final) para terminar con la instalación.

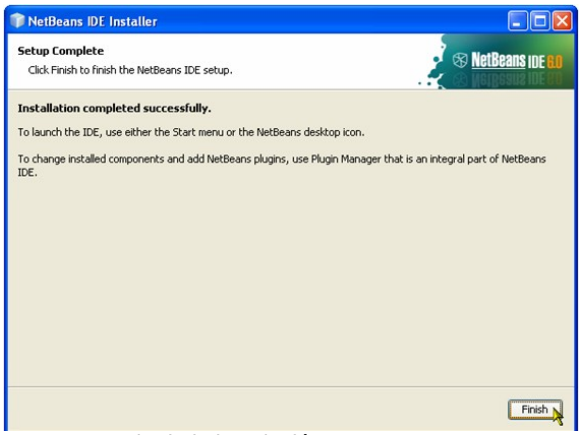


Imagen 18: Fin de la instalación

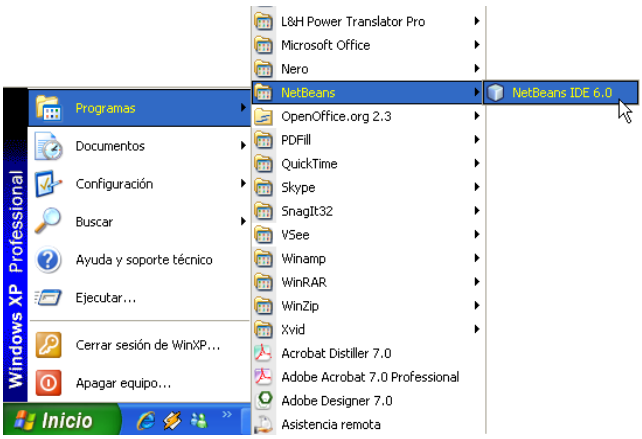
Terminado la instalación de los dos programas ya tenemos instalado el NetBeans en nuestra computadora y lo podemos apreciar por el siguiente icono.

Imagen 19: Icono del NetBeans

EJECUTANDONETBEANSPORPRIMERAVEZ

Existen dos caminos:

1. INICIO >> PROGRAMAS >> NETBEANS >> NETBEANS IDE 6.0



2. DOBLE CLICK SOBRE EL ICONO DEL NETBEANS UBICADO EN EL ESCRITORIO

INTERFAZGRAFICADEDESARROLLO(IDE) DELNETBEANS6.0

1. Barra de título: Icono del programa y el nombre de la herramienta de programación

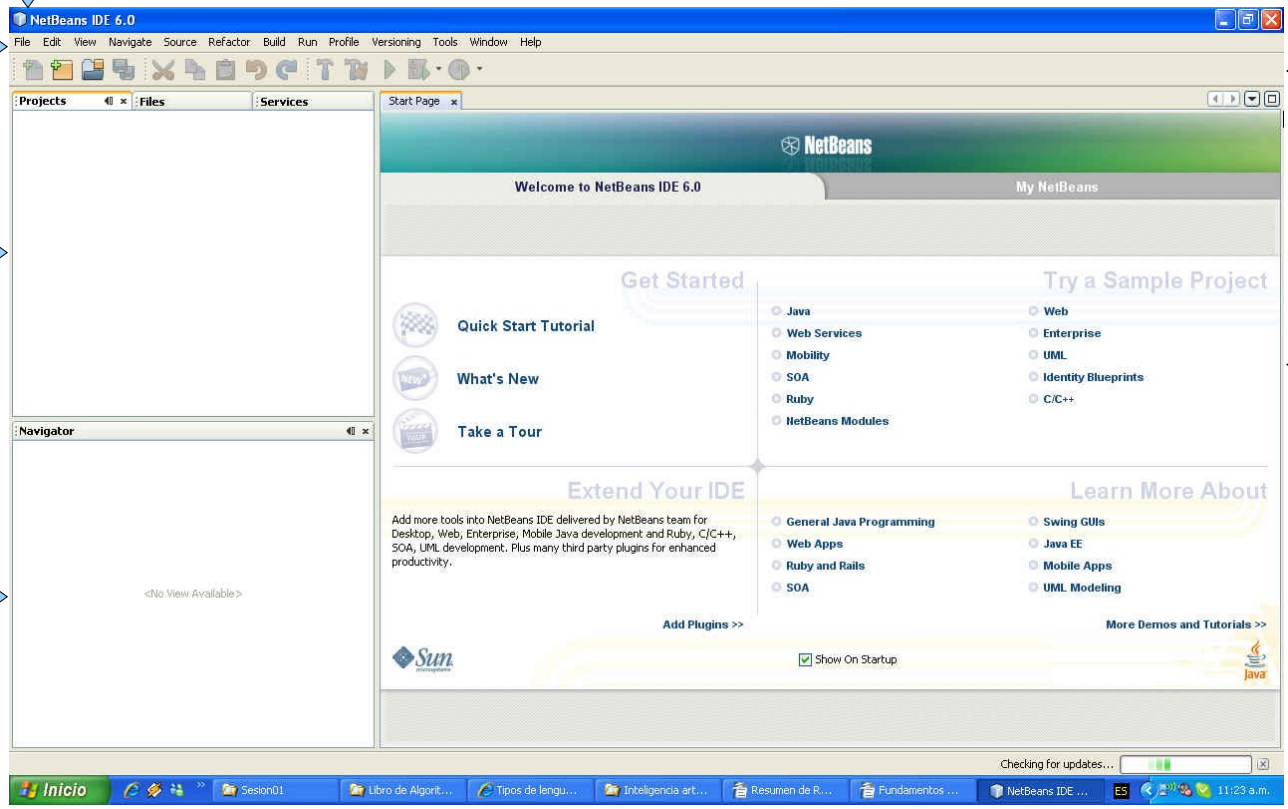
2. Barra de menús

3. Barra estandar

4. Ventana de proyectos, archivos y servicios

5. Ventana de navegación

6. Ventana de edición



Definición de NETBEANS

Netbeans es un entorno de desarrollo integrado (IDE) que permite editar programas en java, compilarlos, ejecutarlos, depurarlos, construir rápidamente el interfaz gráfico de una aplicación eligiendo los componentes de una paleta, etc.

1. Barra de título

Todas las ventanas de una herramienta de programación contienen una barra de título en el cual se ve el título de la aplicación (programa) y los botones de control de la ventana con los cuales es posible hacer que se reduzca a un botón en la barra de tareas (tamaño mínimo), ordenar que ocupe toda la pantalla (tamaño máximo), lograr que recupere el tamaño que tenía antes de ser la ventana máxima o mínima (restaurar) o, simplemente, cerrar la ventana.

2. Barra de menús

En está área aparecen los menús disponibles. Cada menú contiene acciones específicas relativas al nombre del menú.

3. Barra estandar

La barra de herramientas estándar contiene botones para las operaciones más habituales de los menús File(Archivo), Edit(Edición), etc. New Project(Nuevo proyecto), Open Project(abrir proyecto), Save all (guardar todo), etc. Estos botones se utilizan del mismo modo que los comandos de menú equivalentes.

4. Ventana de proyectos, archivos y servicios

Es la ventana que va a contener el objeto activo del proyecto, los paquetes y librerias. Ademas los archivos del proyecto y los servicios que se desean utilizar en la aplicación.

5. Ventana de navegación

Esta ventana nos permite visualizar los objetos que contiene el proyecto actual agrupados por categorías

6. Ventana de edición

Es la ventana donde se va a realizar el código de los programas, el diseño de los formularios, etc.

¿QUE ES UN PROGRAMA?

Conjunto de código, agrupados por instrucciones, donde cada instrucción le dice a la computadora que operaciones debe realizar para resolver el problema.

```
public static void main(String[] args) {
    String N1,N2;
    int n1,n2,suma;

    /*n1 = 5;n2=9;suma=n1+n2;
    System.out.println(suma);*/
    N1 = JOptionPane.showInputDialog("Ingrese N1 : ");
    N2 = JOptionPane.showInputDialog("Ingrese N2 : ");
    n1 = Integer.parseInt(N1);
    n2 = Integer.parseInt(N2);

    suma= n1+n2;

    JOptionPane.showMessageDialog(null, "La suma es : " + suma
    System.exit(0);
```

Conjunto de código

Imagen 1: Código de un programa.

¿QUE ES UN LENGUAJE DE PROGRAMACIÓN?

Un Lenguaje de Programación es un lenguaje que los programadores usan para comunicar instrucciones a una computadora y poder ejecutar un programa. Está constituido por un grupo de reglas gramaticales, un grupo de símbolos utilizables, un grupo de términos monosémicos (es decir, con sentido único) que definen su estructura y el significado de sus elementos y expresiones.

TIPOS DE LENGUAJE DE PROGRAMACIÓN

Existen tres tipos de lenguaje de programación:

1. Lenguajes de bajo nivel
2. Lenguajes de medio nivel
3. Lenguajes de alto nivel

Lenguajes de bajo nivel

Son lenguajes totalmente dependientes de la máquina, es decir que el programa que se realiza con este tipo de lenguajes no se pueden migrar o utilizar en otras maquinas. Este lenguaje es mucho más rápido que los lenguajes de alto nivel. La desventaja es que son bastantes difíciles de manejar y usar, además de tener códigos fuente enormes donde encontrar un fallo es casi imposible.

Ejemplo:

- Lenguaje ASSEMBLER

Imagen 2: Código Assembler

Lenguajes de alto nivel

Son aquellos que se encuentran más cercanos al lenguaje natural que al lenguaje máquina. Se

tratan de lenguajes independientes de la arquitectura del ordenador. Por lo que, en principio, un programa escrito en un lenguaje de alto nivel, lo puedes migrar de una máquina a otra sin ningún tipo de problema.

Ejemplos:
- JAVA
- PROLOG
- C++
- DELPHI
- Otros.

Imagen 3: Herramienta de Programación NetBeans 6.0

Lenguajes de Medio nivel
Estos lenguajes se encuentran en un punto medio entre los dos anteriores. Dentro de estos lenguajes podría situarse el lenguaje de programación C ya que puede acceder a los registros del sistema, trabajar con direcciones de memoria, todas ellas características de lenguajes de bajo nivel y a la vez realizar operaciones de alto nivel.
Ejemplos
- C
- BCPL

CARACTERÍSTICAS DE UN PROGRAMA
- Debe ser confiable y funcional.
- Advertir errores de entrada obvios comunes.
- Documentado adecuadamente.
- Ser comprensible.
- Códificado en el lenguaje apropiado.

DATOS
La materia prima de que se nutren los programas para producir resultados. Pueden ser de varios tipos: numéricos, alfabéticos, alfanuméricos (cualquier conjunto de símbolos) y lógicos (solo dos valores posibles, verdadero o falso).

INFORMACIÓN
Es lo que se obtiene del procesamiento de datos. Todo aquello que permite adquirir cualquier tipo de conocimientos

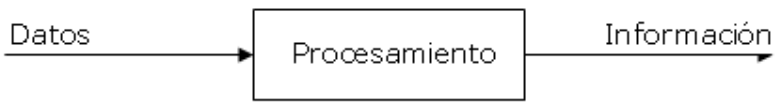


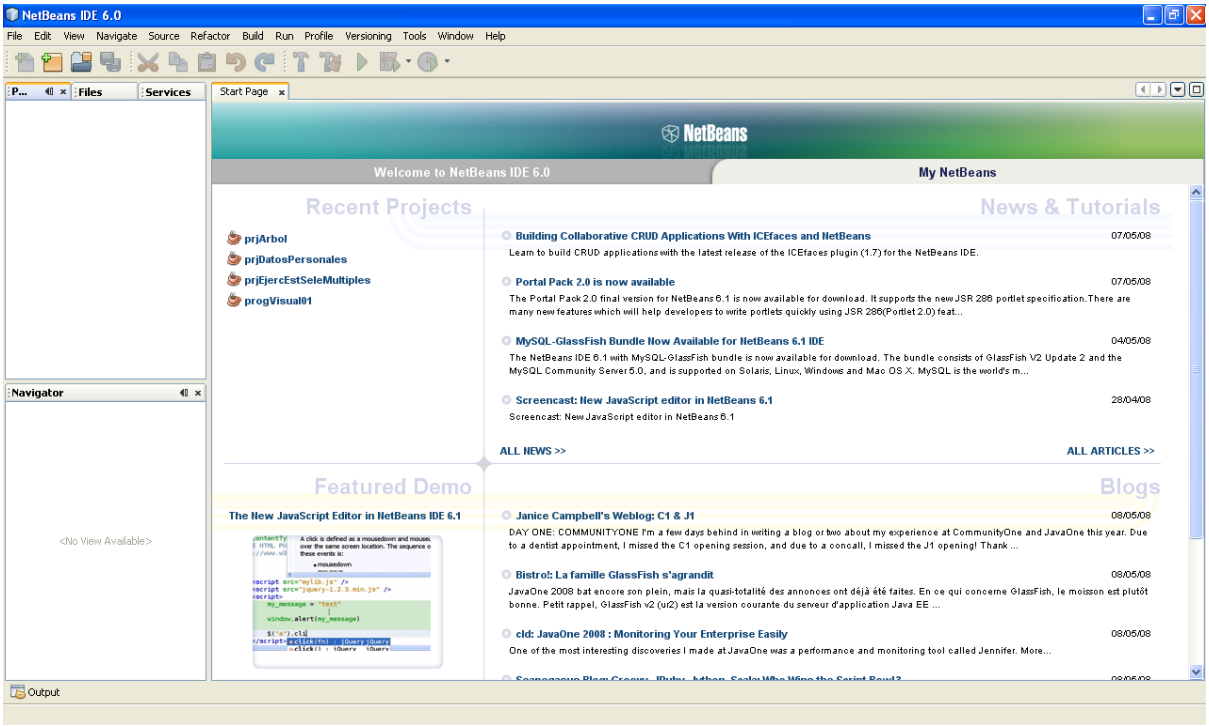
Imagen 4: Procesando los datos

MI PRIMER PROYECTO

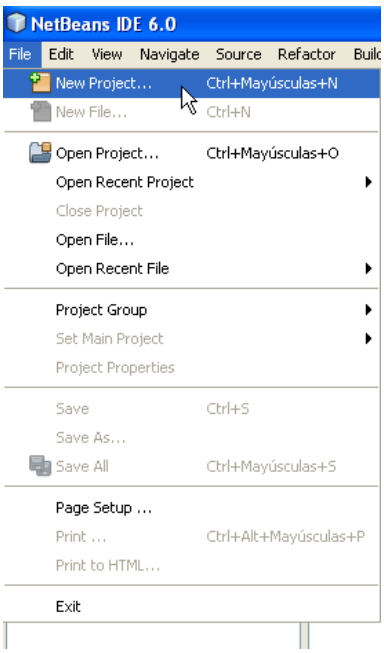
El NetBeans es un entorno de desarrollo integrado que permite crear aplicaciones de escritorio, aplicaciones web y aplicaciones móviles utilizando las últimas tecnologías para los desarrolladores de software de Java. El IDE de NetBeans es un producto gratuito y sin restricciones de uso pudiendo escribir, compilar, depurar e implementar programas en Java.

NetBeans es un proyecto open source de desarrollo escrito en Java. La plataforma NetBeans da soporte para escritura de servlets, ayuda on-line y ayudas con el código. Usaremos la versión 6.0 de NetBeans para la construcción y diseño de las aplicaciones.

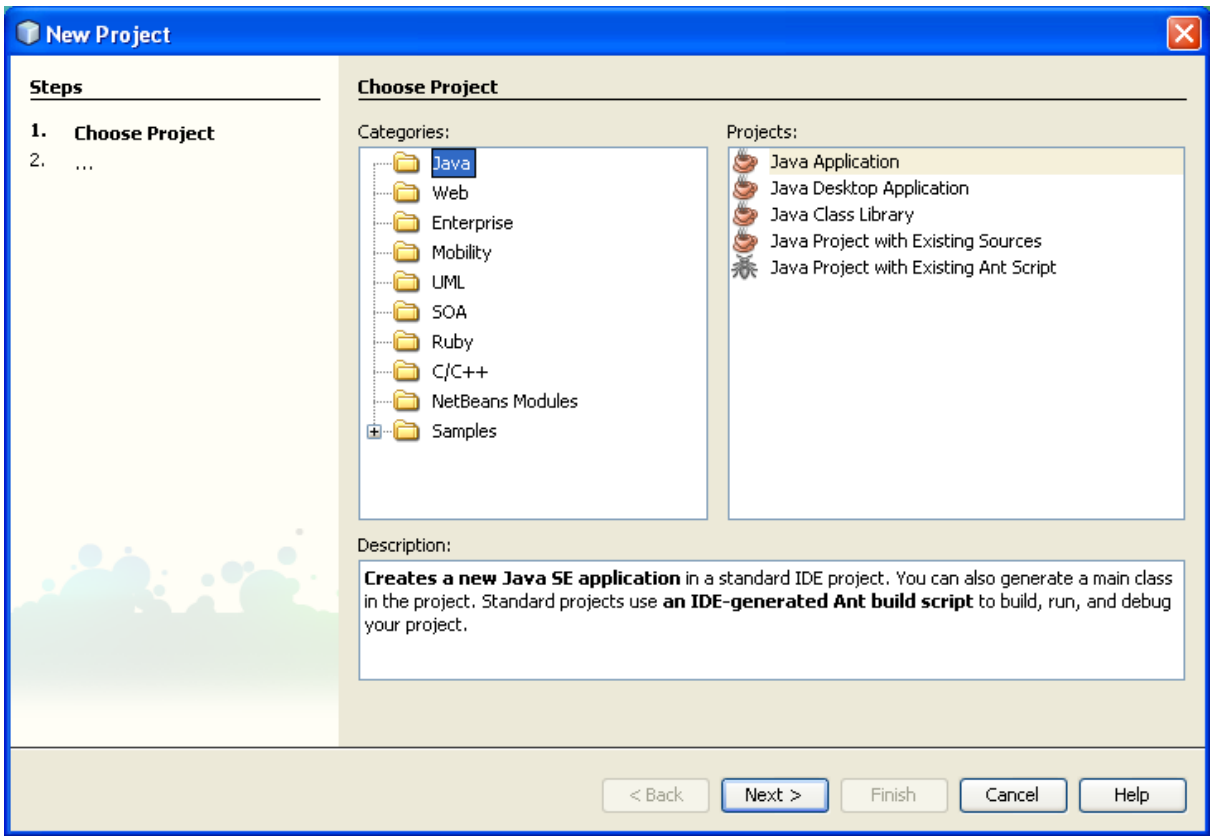
Una vez que ingresas al entorno de desarrollo de NetBeans se observa:



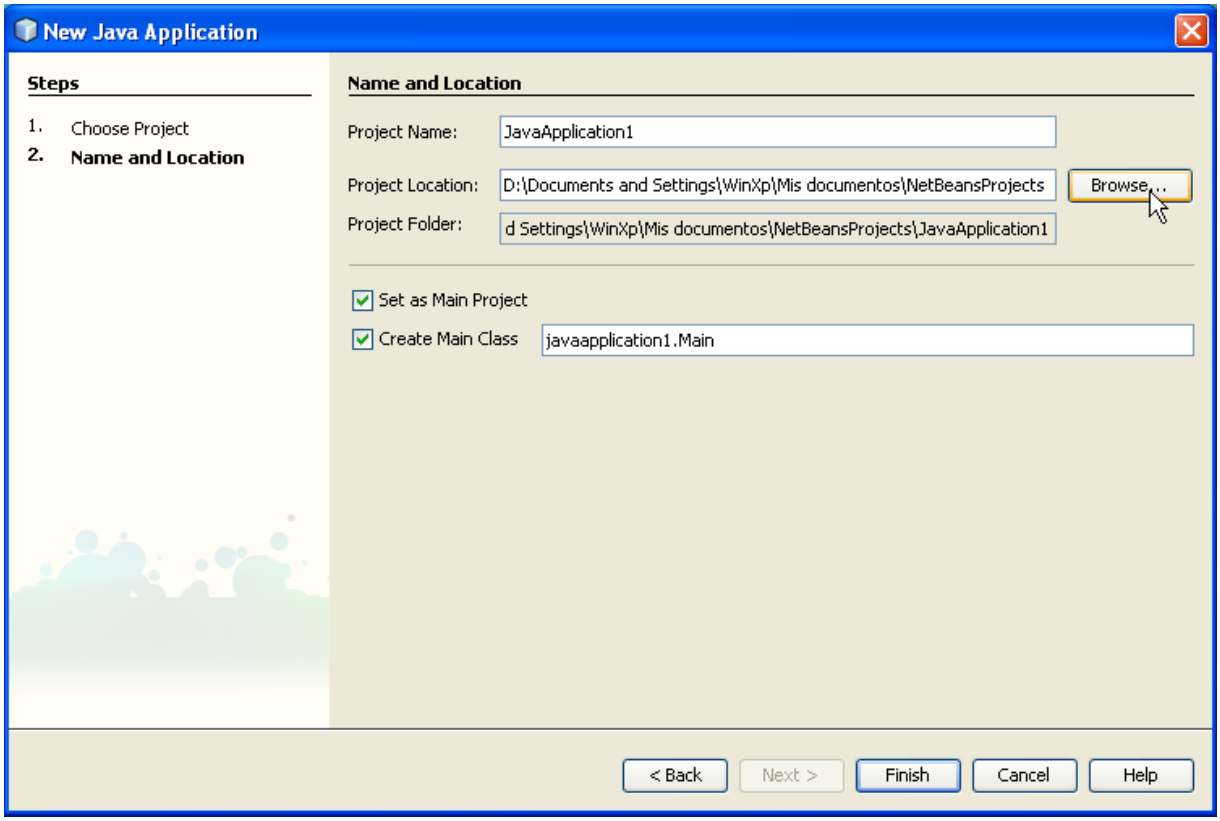
Para dar inicio a una aplicación de Java bajo el entorno de Netbeans se debe definir un proyecto, para ello, seleccionas la opción del menú denominada ***File.*** Se muestra inmediatamente un menú flotante cuya primera opción indica ***New Proyect***, está opción la seleccionas.



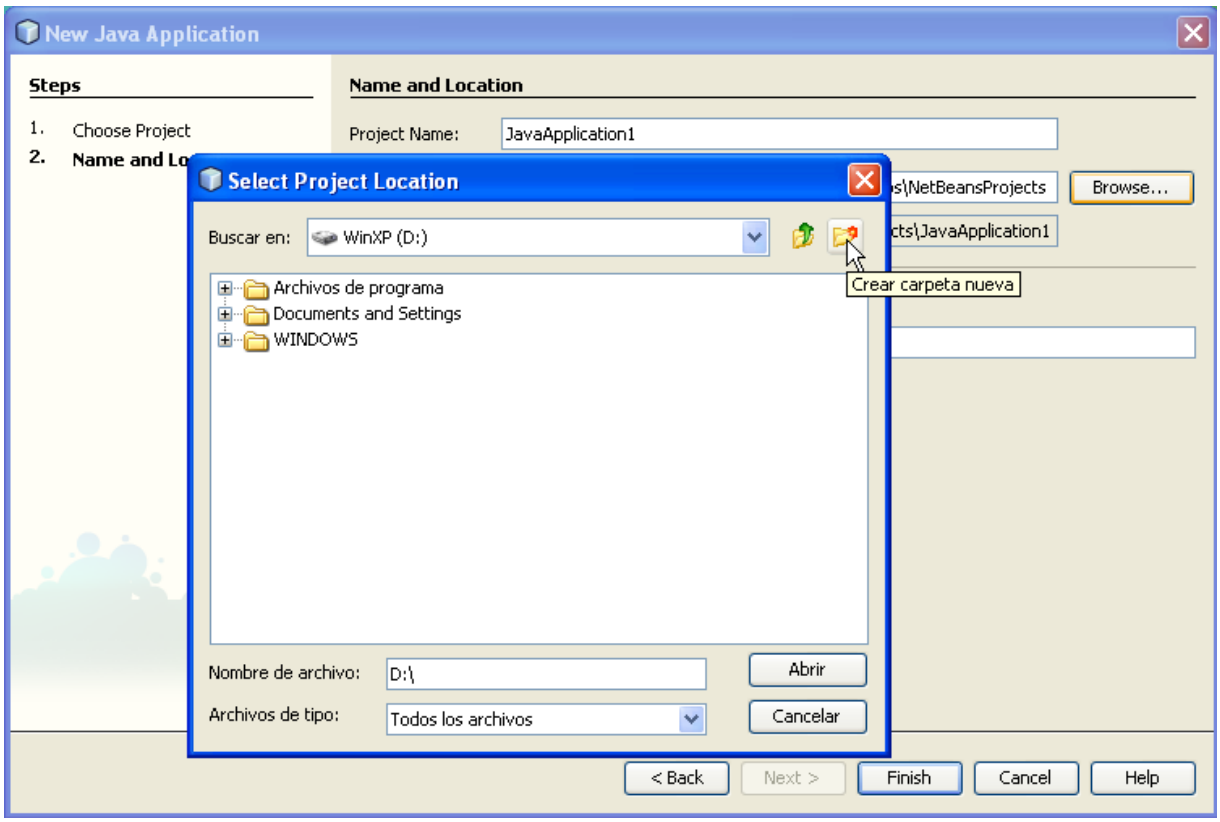
Al momento de seleccionar ***New Proyect*** se visualiza la ventana siguiente:

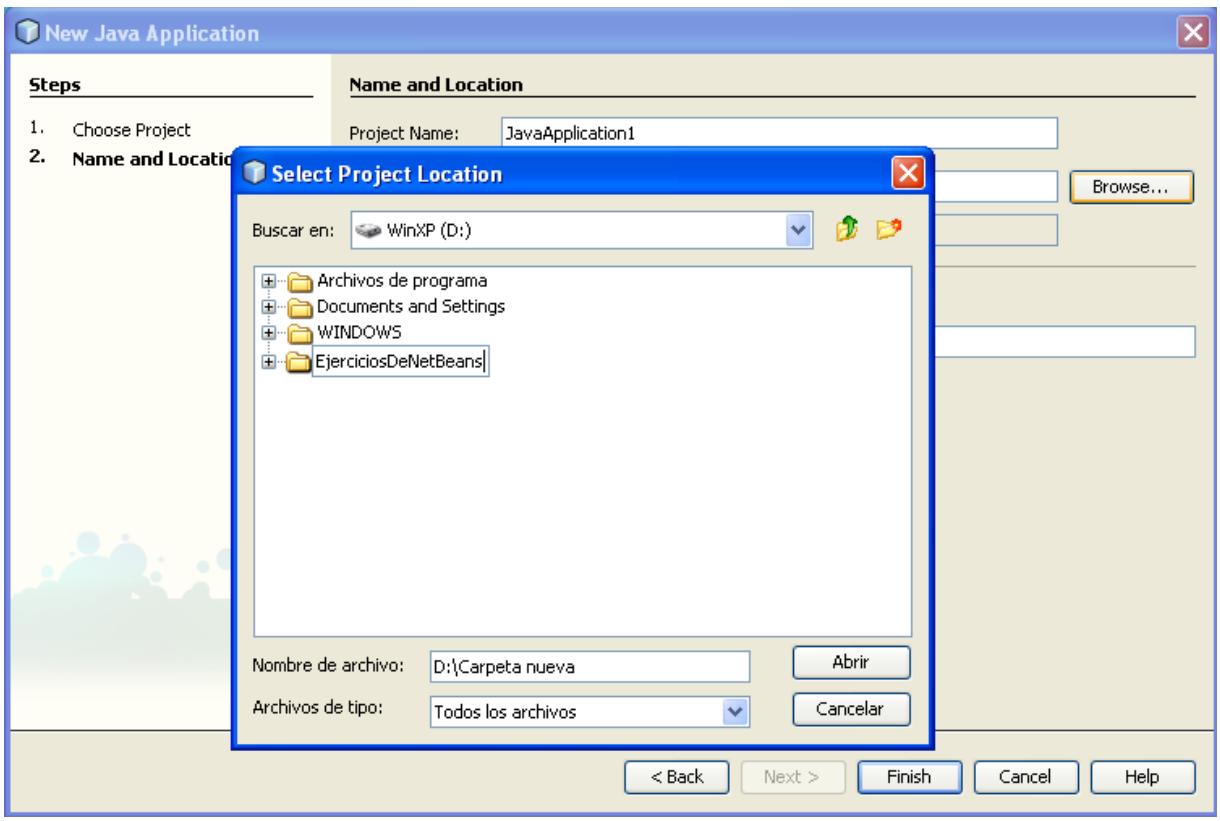


Dado que nuestras aplicaciones van ser desarrolladas en entorno no visual, es decir en modo consola, en **Categories** seleccionas la carpeta Java y en **Proyects** seleccionas Java Application. Luego hacer click en el botón de comando ***Next*** que mostrará la siguiente ventana:

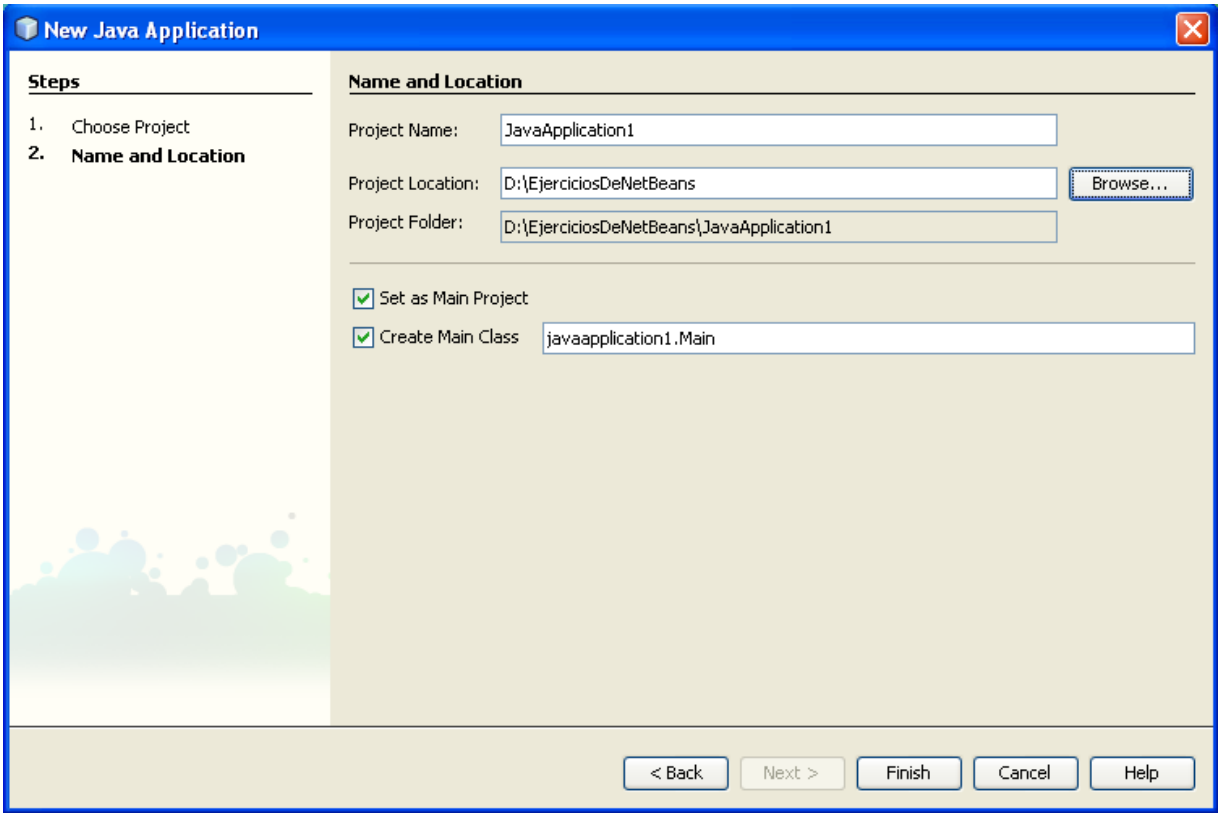


Es conveniente que uno mismo cree su carpeta de destino de los archivos que se generan para la construcción de una aplicación. Supongamos que la carpetas que necesitamos crear se llama EjerciciosDeNetBeans y la creamos en la unidad D, para ello es necesario dar click en el botón de comando ***Browse***.

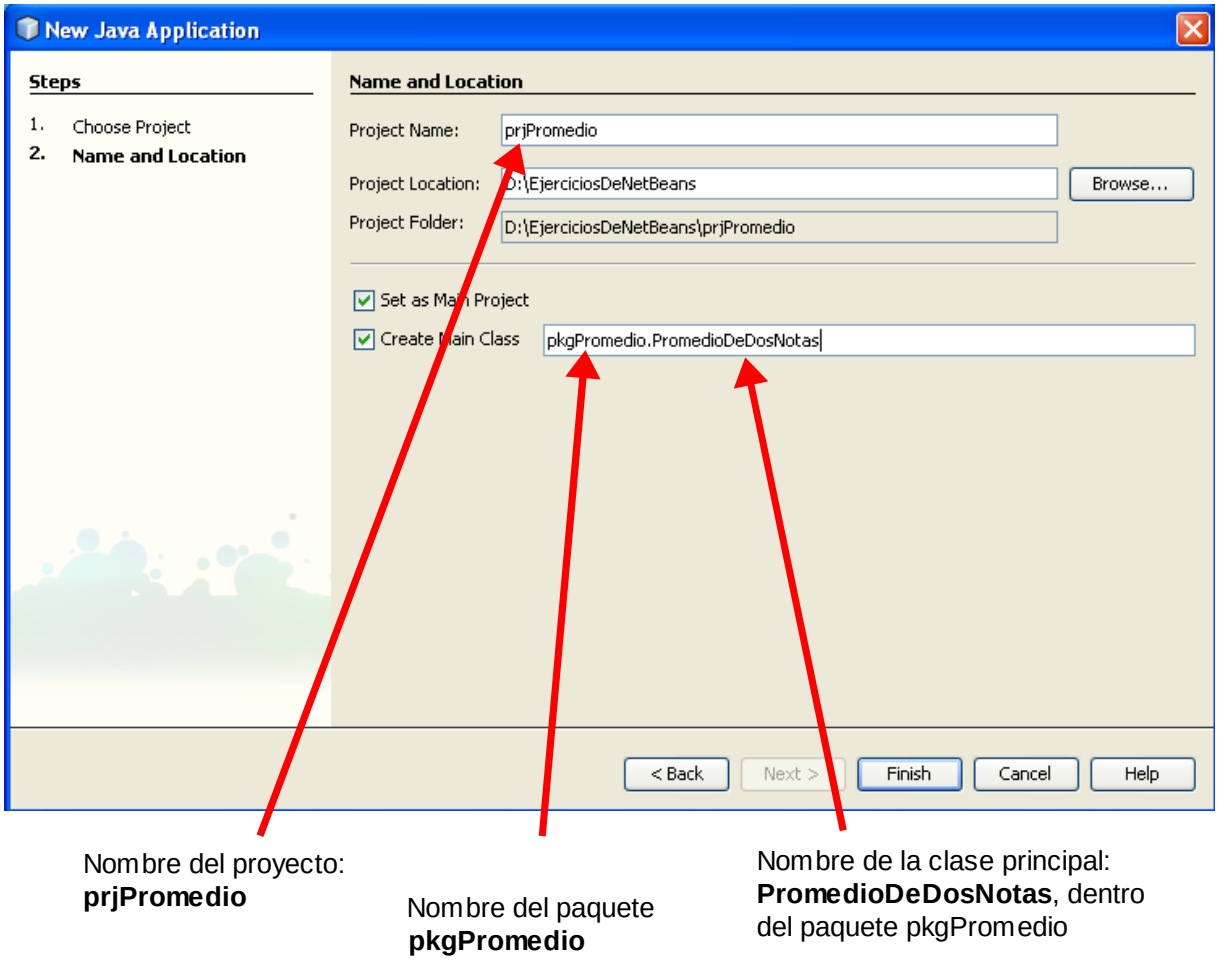

New Java Application
Steps
1. Choose Project
2. Name and Lo
Name and Location
Project Name:
JavaApplication1
Browse...
Select Project Location
Buscar en:
WinXP (D:)
Crear carpeta nueva
Archivos de programa
Documents and Settings
WINDOWS
Nombre de archivo:
D:\
Abrir
Archivos de tipo:
Todos los archivos
Cancelar
< Back
Next >
Finish
Cancel
Help


New Java Application
Steps
1. Choose Project
2. Name and Locati
Name and Location
Project Name:
JavaApplication1
Browse...
Select Project Location
Buscar en:
WinXP (D:)
Archivos de programa
Documents and Settings
WINDOWS
EjerciciosDeNetBeans
Nombre de archivo:
D:\Carpeta nueva
Abrir
Archivos de tipo:
Todos los archivos
Cancelar
< Back
Next >
Finish
Cancel
Help

Una vez indicada la nueva carpeta EjerciciosDeNetBeans, procede a dar click en el botón de comando ***Abrir*** quedando la ventana **New Java Application** de la siguiente forma:



Se observa que en **Proyect Location** se muestra la carpeta destino del proyecto. Vamos a suponer que se quiere construir un programa que calcule el promedio de dos notas, entonces la ventana debería quedar de la siguiente forma:

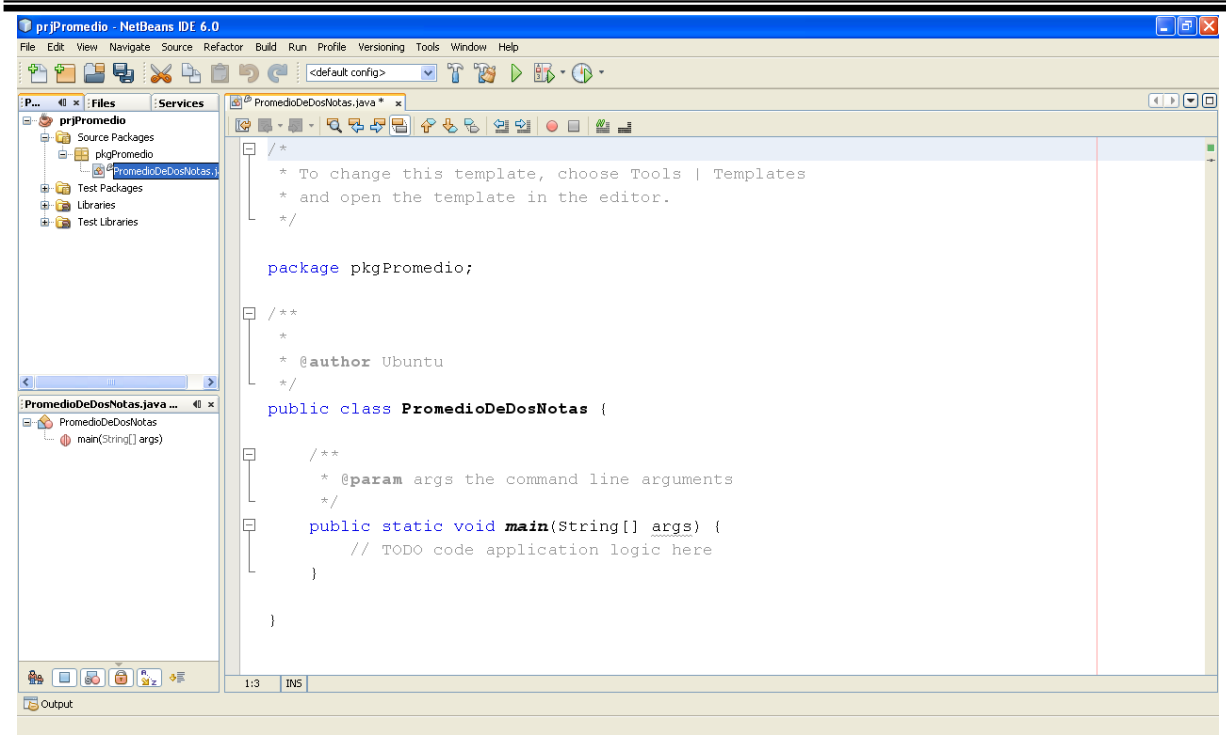


Nombre del proyecto: **prjPromedio**

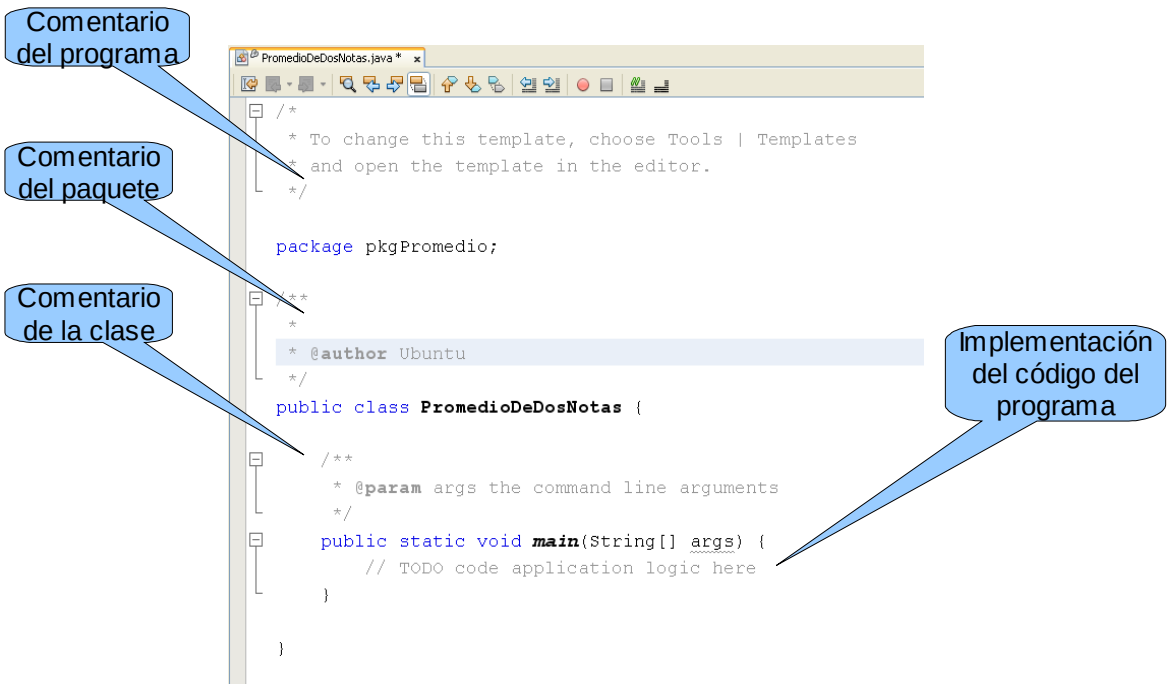
Nombre del paquete **pkgPromedio**

Nombre de la clase principal: **PromedioDeDosNotas**, dentro del paquete pkgPromedio

Al dar click en ***Finish*** se mostrará el entorno de desarrollo de NetBeans listo para dar inicio a la construcción de la aplicación.



Ahora estamos listo para comenzar a realizar nuestro primer ejercicio de programación en NetBeans. Pero antes conoceremos la estructura de la clase principal.



EJERCICIO 01

Realizar un programa para el curso de Fundamentos de Programación que permita calcular la nota promedio final de un estudiante en base a la nota de la 1ra unidad y de la 2da Unidad.

Solución

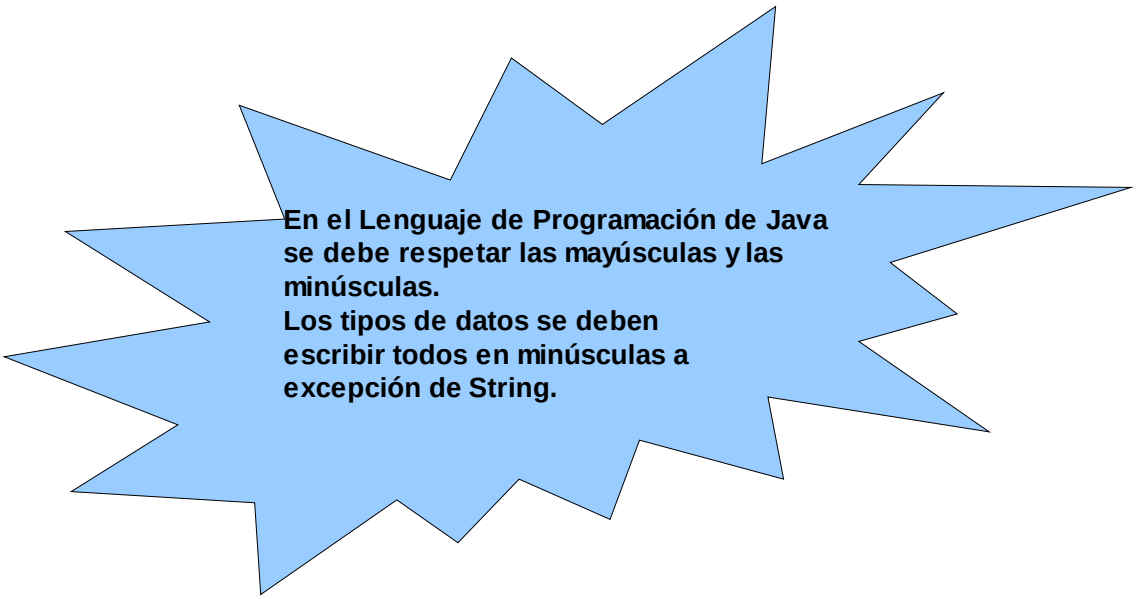
Este programa se va a desarrollar de dos fomas, para que ustedes estimados estudiantes puedan diferenciar la programación con datos fijos y variables de las notas de las dos unidades.

Lenguaje de Programación Java

Para resolver este ejercicio primero debemos conocer un poco el lenguaje de programación Java.

Tipos de Datos

Algorítmico	Java	Significado
entero	int	Entero corto
	long	Entero largo
real	float	Real corto
	double	Real Largo
carácter	char	Caracter
cadena	String	Cadena



Ejemplos:

- int nota1
- int hora_inicial
- int nota1, nota2, nota3
- float pi
- duoble promedio
- double precio_azucar, precio_producto
- char letra_abecedario
- char UnSimbolo
- String nombres
- String ApellidoPat, ApellidoMat
- String Nombre_Paises
- String Nombre_De_La_Ciudad

1eraforma:

Se desarrollará el problema conociendo que las dos notas de la 1era unidad y 2da unidad son: 13 y 17 respectivamente.

La solución en algoritmo por intermedio del pseudocódigo en esta 1era forma es el siguiente:

```
algoritmo  PromedioDeNotas
var
        entero : n1, n2
        real : prom
inicio
        n1 = 13
        n2 = 17
        prom = ( n1 + n2 )/2
        mostrar ( 'El promedio final es ', prom )
fin
```

La solución en código de este ejercicio en esta 1era forma es el siguiente:

```
public static void main(String[] args) {
    int n1,n2;
    double prom;
    n1=13;
    n2=17;
    prom=(n1+n2)/2;
    System.out.println("El promedio final es " + prom);
}
```

Nota: Toda linea de código debe acabar con un punto y coma (;)

Instrucciones de código	Significado
int n1,n2;	Declaración de dos variables **n1** y **n2** de tipo de dato entero
double prom;	Declaración de la variable **prom** de tipo real
n1=13;	A la variable **n1** se le asigna **13**
n2=17;	A la variable **n2** se le asigna **17**
prom=(n1+n2)/2;	Se realiza el proceso de obtener el promedio. El resultado se almacenará en la variable **prom**.
System.out.println("El promedio final es " + prom);	Se reporta el resultado del promedio, por intermedio de un código estandar. System = Sistema out = salida println = Imprimir o mostrar por pantalla los resultados y hacer un salto de línea.

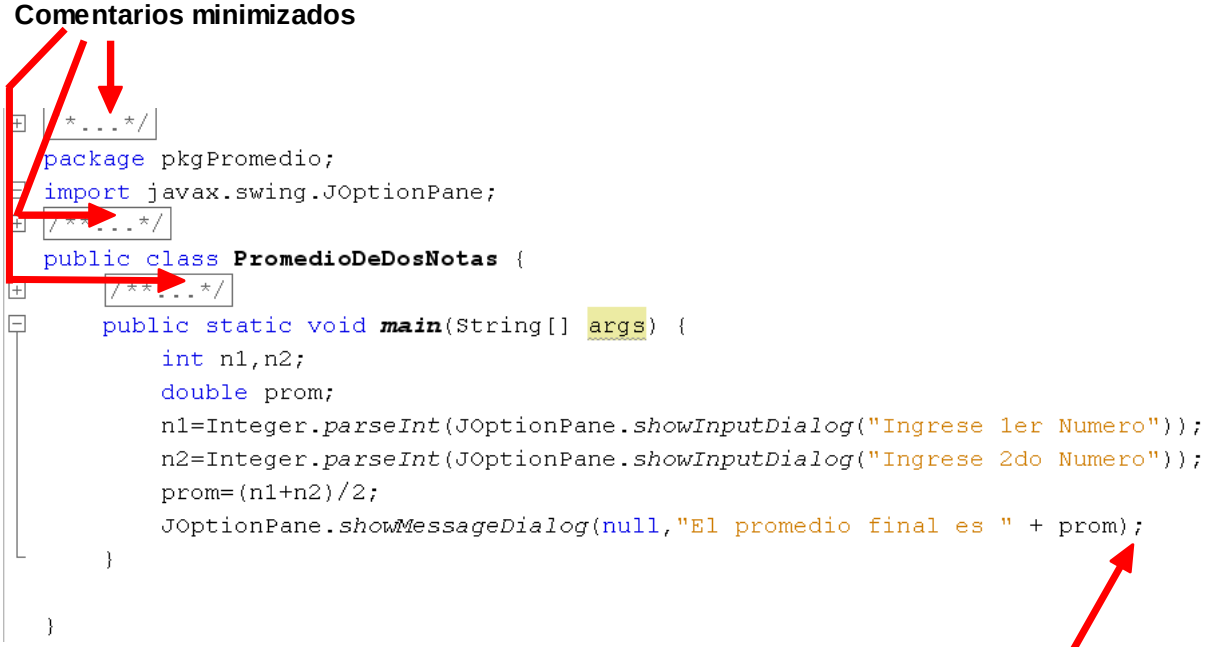
2daforma:

La solución en algoritmo por intermedio del pseudocódigo en esta 2da forma es el siguiente:

```
algoritmo  PromedioDeNotas
var
            entero : n1, n2
            real : prom
inicio
            Leer ( n1 )
            Leer ( n2 )
            prom = ( n1 + n2 )/2
            mostrar ( 'El promedio final es ', prom )
fin
```

La solución en código de este ejercicio en esta 2da forma es el siguiente:

Comentarios minimizados



```
/*...*/
package pkgPromedio;
import javax.swing.JOptionPane;
/**...*/
public class PromedioDeDosNotas {
    /**...*/
    public static void main(String[] args) {
        int n1,n2;
        double prom;
        n1=Integer.parseInt(JOptionPane.showInputDialog("Ingrese 1er Numero"));
        n2=Integer.parseInt(JOptionPane.showInputDialog("Ingrese 2do Numero"));
        prom=(n1+n2)/2;
        JOptionPane.showMessageDialog(null,"El promedio final es " + prom);
    }

}
```

Nota: Toda linea de código debe acabar con un punto y coma (;)

En la siguiente tabla se explica que hace cada instrucción, y los elementos que intervienen en ellas:

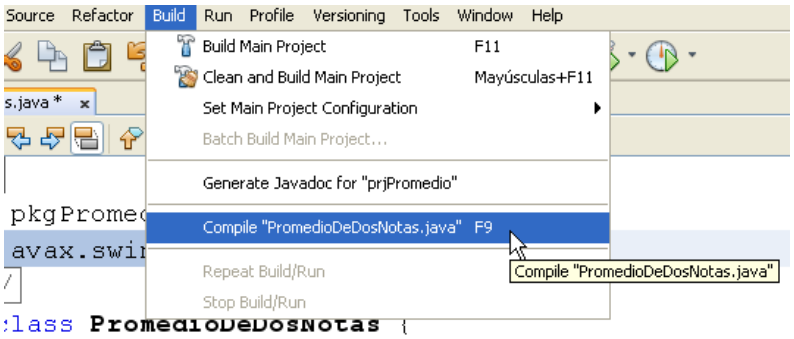
Instrucciones de código	Significado
import javax.swing.JOptionPane;	**import** es una instrucción de código que permite importar clases y pueda ser utilizado dentro del programa. **javax.swing.JOptionPane** es código que llama al objeto *JOptionPane* que sirve para representar un cuadro de dialgo (dialog box), para varios propósitos: 1. Mostrar Mensaje a través del uso de **showMessageDialog.** 2. Preguntar por la confirmación del usuario **showConfirmDialog.** 3. Obtener datos de entrada ingresados por el usuario **ShowInputDialog.** 4. La combinación de los tres de arriba **ShowOptionDialog.**
int n1,n2;	Declaración de dos variables **n1** y **n2** de tipo de dato entero
double prom;	Declaración de la variable **prom** de tipo real
n1=Integer.parseInt(JOptionPane.showInputDialog("Ingrese 1er Numero"));	Esta instrucción tiene dos partes: La 1era. **JOptionPane.showInputDialog("Ingrese 1er Número")** Solicita al usuario que ingrese un dato que viene hacer el 1er número. La 2da. **Integer.parseInt(...)** Esta instrucción premite convertir una cadena a valor entero. La instrucción JoptionPane devuelve una cadena. **Conclusión** El valor ingresado en JOptionPane, este lo devuelve como cadena y con la función “Integer.parseInt” lo convertirmos a entero y lo asignamos a la variable **n1**.
n2=Integer.parseInt(JOptionPane.showInputDialog("Ingrese 2do Numero"));	Lo mismo que la instrucción anterior. El segundo valor ingresado se asigna a la variable **n2**.
prom=(n1+n2)/2;	Se realiza el proceso de obtener el promedio. El resultado se almacenará en la variable **prom**.
JOptionPane.showMessageDialog(null,"El promedio final es " + prom);	Se reporta el resultado del promedio por intermedio de JoptionPane. Para ello se utiliza la propiedad **showMessageDialog**. 1ero, **null** significa que el cuadro de diálogo se mostrará en el centro de la pantalla, y 2do, el mensaje del resultado + el valor de la variable **prom**.

¿Que les parecio estas dos formas de programar?.....
Les comento:
Por la primera forma si ejecutamos este programa mil veces siempre dará el mismo resultado, porque los valores de entrada siempre son los mismos, por tanto reporta el mismo resultado.
En la segunda forma las cosas cambian, porque los datos de entrada pueden ser variables en cada vez que se ejecuta el programa, por tanto el resultado sera diferente.

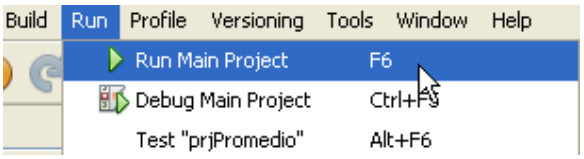
PASOSPARAEJECUTARUNPROGRAMA

Cuando este terminado todo el código del programa estamos listo de ver los resultados, para ello debemos hacer lo siguiente:

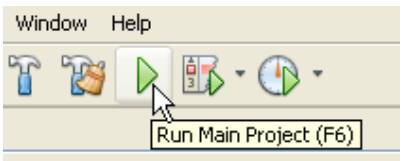
1er paso. Pulsamos la tecla F9 para compilar nuestro programa y ver si existen errores en la codificación de nuestro programa, si todo esta bien continuamos con el siguiente paso, pero si hubierán errores el programa jamas se ejecutará mientras no se corrigan los errores.
También se puede hacer este paso con el uso del Menu Build



2do paso. Despues de verificado que no hay errores, pulsamos la tecla F6 para ejecutar el programa.
También se puede hacer este paso con el uso del Menu Run



o por el comando encontrado en la barra de herramientas estandar



Nota: El ejecutar el programa también se llama *correr el programa*.

PROYECTO DE ESTRUCTURA SELECTIVA DOBLE

A continuación resolveremos el siguiente proyecto.

LABORATORIONº03-EJERCICIO01
Determinar si un alumno aprueba a desaprueba un curso, conociendo que aprobara si su promedio de tres calificaciones es mayor o igual a 10.5; desaprobado en caso contrario.

Para resolver este ejercicio abriremos un nuevo proyecto.

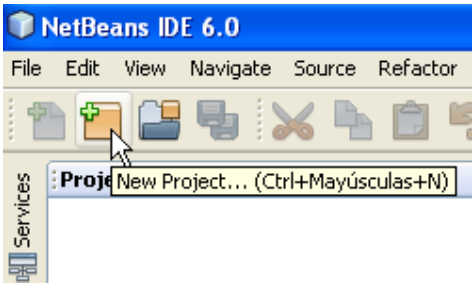


Figura 01: Eleccion de un nuevo proyecto

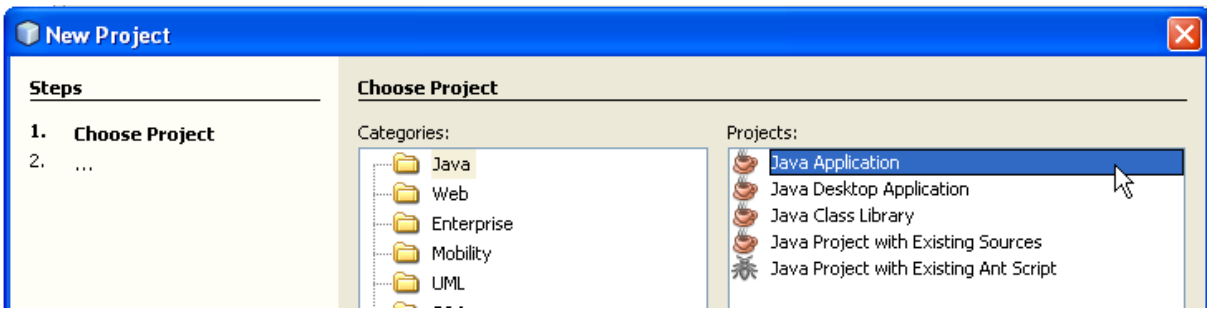


Figura 02: Eleccion de un proyecto Java Application

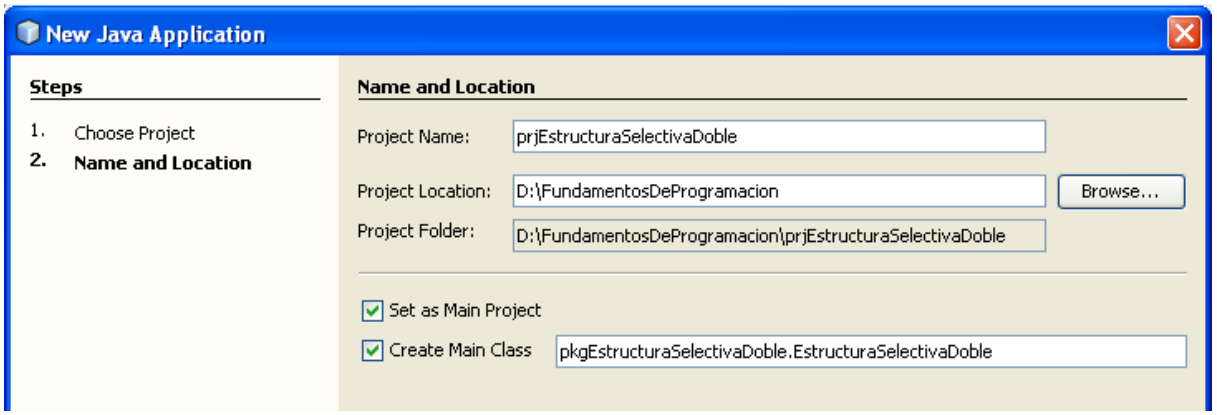


Figura 03: Ingresar nombre del proyecto, paquete y de la clase principal

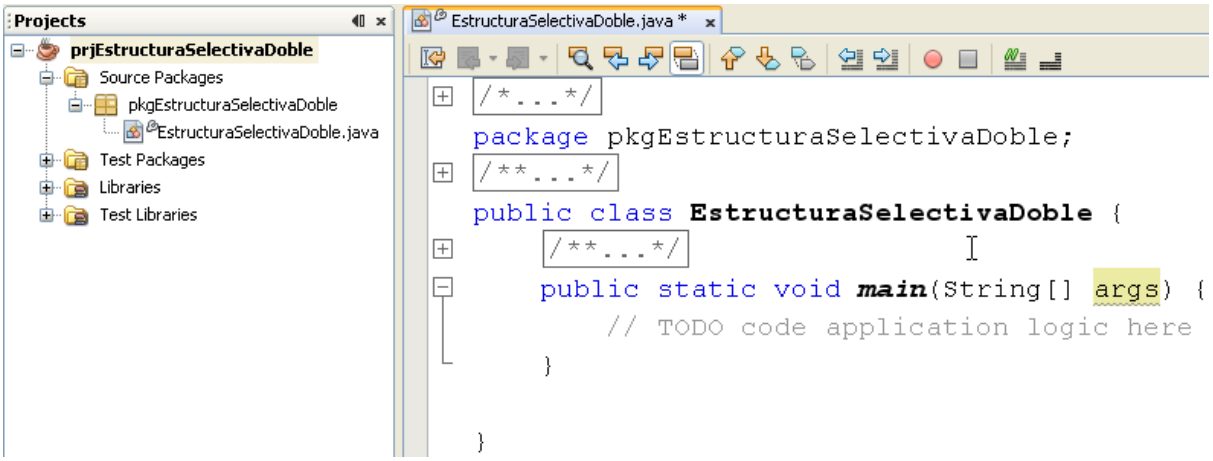


Figura 04: Proyecto listo para programar

Ahora estamos listo para comenzar con la programación de este ejercicio. Antes hay que recordar como esta resuelto en algoritmo.

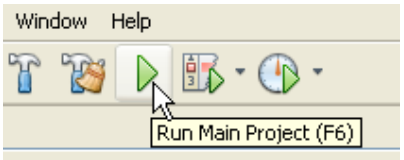
```
algoritmo Lab3Ejercicio01
var
        entero : calif1, calif2, calif3
        real : prom
inicio
        Leer calif1, calif2, calif3
        prom = (calif1 + calif2 + calif3)/3
        si prom >= 10.5 entonces
                Escribir 'ALUMNO APROBADO'
        si no
                Escribir 'ALUMNO DESAPROBADO'
        fin_si
fin
```

La solución en código de este ejercicio es el siguiente:

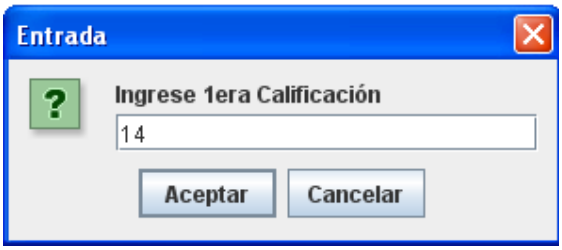
```
package pkgEstructuraSelectivaDoble;
import javax.swing.JOptionPane;

public class EstructuraSelectivaDoble {

   public static void main(String[ ] args) {
      int calif1, calif2, calif3;
      double prom;
      calif1=Integer.parseInt(JOptionPane.showInputDialog("Ingrese 1era Calificación"));
      calif2=Integer.parseInt(JOptionPane.showInputDialog("Ingrese 2da Calificación"));
      calif3=Integer.parseInt(JOptionPane.showInputDialog("Ingrese 3era Calificación"));
      prom=(calif1 + calif2 + calif3)/3.0;
      if(prom>=10.5)
         JOptionPane.showMessageDialog(null,"ALUMNO APROBADO");
      else
         JOptionPane.showMessageDialog(null,"ALUMNO DESAPROBADO");

   }
}
```

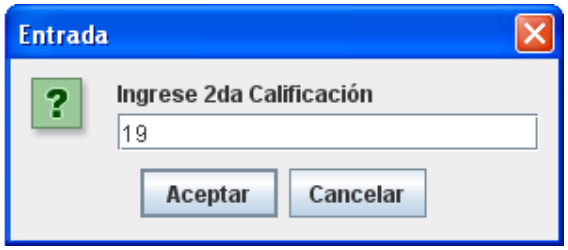
```
/*...*/
package pkgEstructuraSelectivaDoble;
import javax.swing.JOptionPane;
/**...*/
public class EstructuraSelectivaDoble {
    /**...*/
    public static void main(String[] args) {
        int calif1, calif2, calif3;
        double prom;
        calif1=Integer.parseInt(JOptionPane.showInputDialog("Ingrese 1era Calificación"));
        calif2=Integer.parseInt(JOptionPane.showInputDialog("Ingrese 2da Calificación"));
        calif3=Integer.parseInt(JOptionPane.showInputDialog("Ingrese 3era Calificación"));
        prom=(calif1 + calif2 + calif3)/3.0;
        if(prom>=10.5)
            JOptionPane.showMessageDialog(null,"ALUMNO APROBADO");
        else
            JOptionPane.showMessageDialog(null,"ALUMNO DESAPROBADO");
    }
}
```

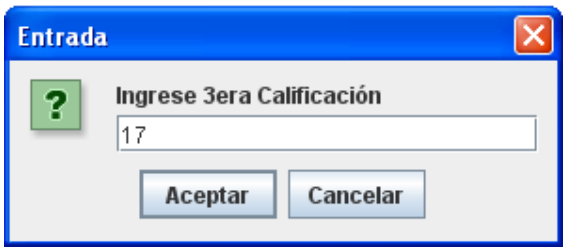
Comprendido el código de la estructura selectiva doble en Java, ahora pasamos a ejecutar el programa para ver los resultados obtenidos.



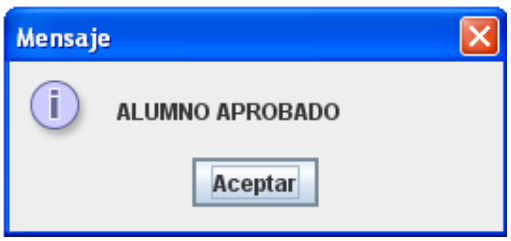
Si ingresamos los siguientes datos:





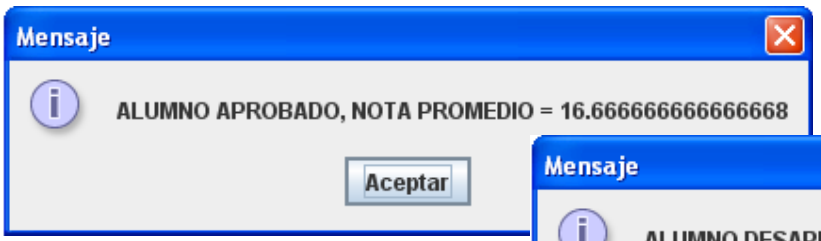


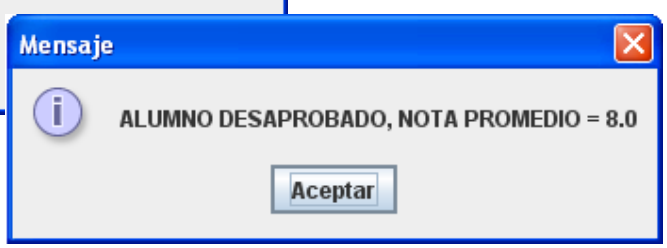
Debe de obtenerse el siguiente resultado:



Datoadicional:

Estimados estudiantes pueden ustedes mejorar el código a manera de práctica y mostrar en el mismo mensaje de aprobado o desaprobado la nota promedio obtenida.





ESTRUCTURA SELECTIVA MÚLTIPLE

A continuación resolveremos el siguiente proyecto de estructura selectiva múltiple.

EJERCICIO 03
Realizar un algoritmo que ingrese número del 1 al 10 y lo muestre en letras.

Para resolver este ejercicio abriremos un nuevo proyecto.

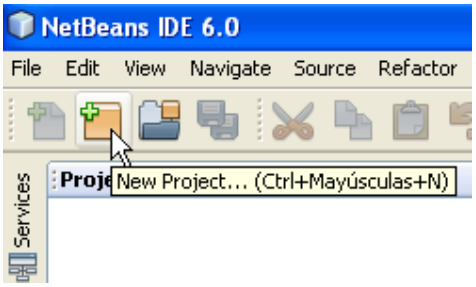


Figura 01: Eleccion de un nuevo proyecto

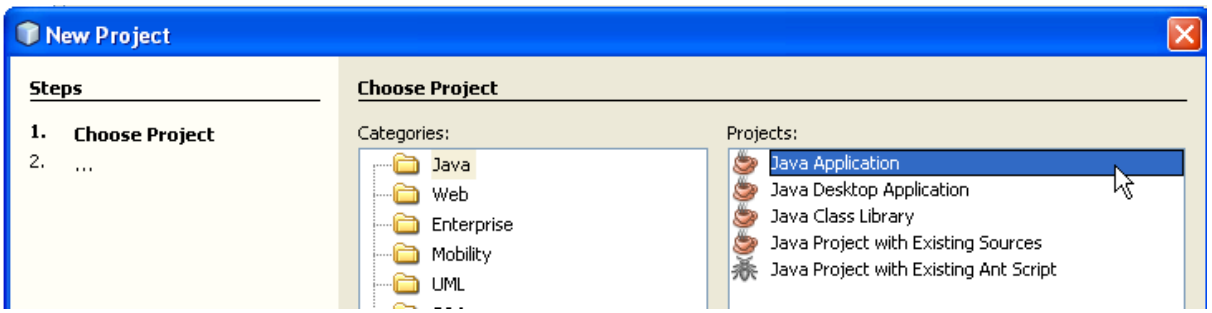


Figura 02: Eleccion de un proyecto Java Application

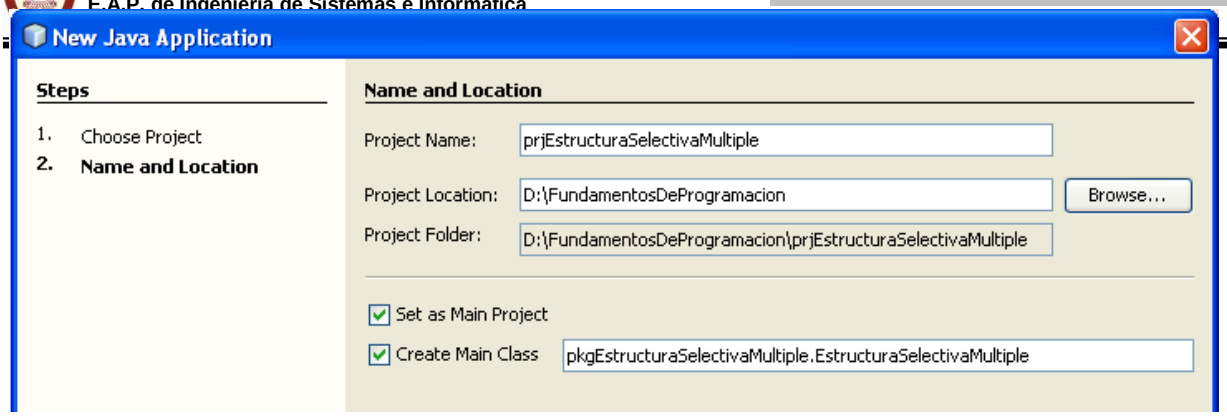


Figura 03: Ingresar nombre del proyecto, paquete y de la clase principal

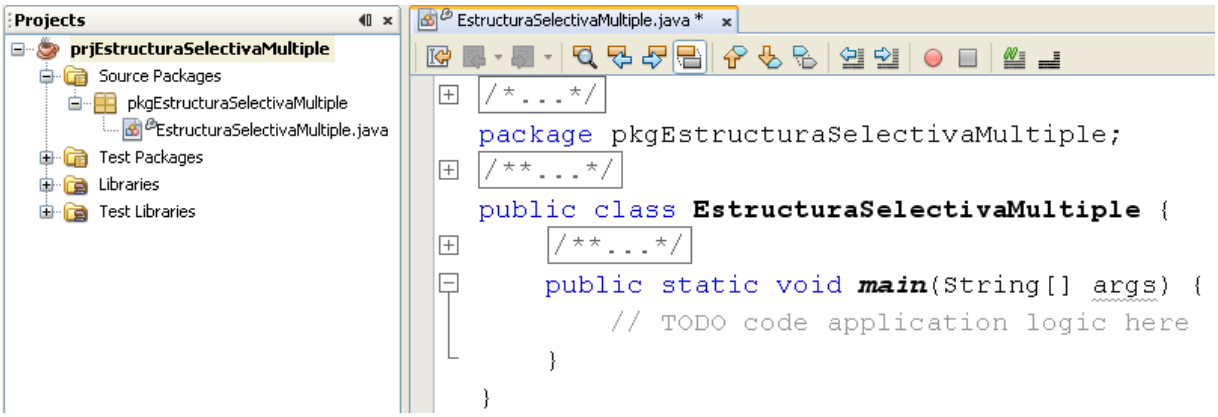


Figura 04: Proyecto listo para programar

Ahora estamos listo para comenzar con la programación de este ejercicio. Antes hay que recordar como esta resuelto en algoritmo.

```
Algoritmo Lab04Ejercicio03
var
        entero: Numero
inicio
        Leer Numero
        en_caso (Numero)
                1 : Mostrar 'UNO'
                2 : Mostrar 'DOS'
                3 : Mostrar 'TRES'
                4 : Mostrar 'CUATRO'
                5 : Mostrar 'CINCO'
                6 : Mostrar 'SEIS'
                7 : Mostrar 'SIETE'
                8 : Mostrar 'OCHO'
                9 : Mostrar 'NUEVE'
                10 : Mostrar 'DIEZ'
                sino : Mostrar 'Numero fuera del rango establecido'
        fin_caso
fin
```

La solución en código de este ejercicio es el siguiente:

```
package pkgEstructuraSelectivaMultiple;
import javax.swing.JOptionPane;

public class EstructuraSelectivaMultiple {
   public static void main(String[ ] args) {
      int Numero;
      Numero=Integer.parseInt(JOptionPane.showInputDialog("Ingrese numero (1-10)"));
      switch(Numero)
      {
         case 1 : JOptionPane.showMessageDialog(null,  "UNO");break;
         case 2 : JOptionPane.showMessageDialog(null,  "DOS");break;
         case 3 : JOptionPane.showMessageDialog(null, "TRES");break;
         case 4 : JOptionPane.showMessageDialog(null, "CUATRO");break;
         case 5 : JOptionPane.showMessageDialog(null, "CINCO");break;
         case 6 : JOptionPane.showMessageDialog(null, "SEIS");break;
         case 7 : JOptionPane.showMessageDialog(null, "SIETE");break;
         case 8 : JOptionPane.showMessageDialog(null, "OCHO");break;
         case 9 : JOptionPane.showMessageDialog(null, "NUEVE");break;
         case 10: JOptionPane.showMessageDialog(null, "DIEZ");break;
         default: JOptionPane.showMessageDialog(null, "Numero fuera del rango establecido");
                  break;
      }
   }
}
```

```
package pkgEstructuraSelectivaMultiple;
import javax.swing.JOptionPane;
public class EstructuraSelectivaMultiple {
    public static void main(String[] args) {
        int Numero;
        Numero=Integer.parseInt(JOptionPane.showInputDialog("Ingrese numero (1-10)"));
        switch(Numero)
        {
            case 1 : JOptionPane.showMessageDialog(null, "UNO");break;
            case 2 : JOptionPane.showMessageDialog(null, "DOS");break;
            case 3 : JOptionPane.showMessageDialog(null, "TRES");break;
            case 4 : JOptionPane.showMessageDialog(null, "CUATRO");break;
            case 5 : JOptionPane.showMessageDialog(null, "CINCO");break;
            case 6 : JOptionPane.showMessageDialog(null, "SEIS");break;
            case 7 : JOptionPane.showMessageDialog(null, "SIETE");break;
            case 8 : JOptionPane.showMessageDialog(null, "OCHO");break;
            case 9 : JOptionPane.showMessageDialog(null, "NUEVE");break;
            case 10: JOptionPane.showMessageDialog(null, "DIEZ");break;
            default: JOptionPane.showMessageDialog(null, "Numero fuera del rango establecido");
                     break;
        }
    }
}
```

ESTRUCTURA SELECTIVA MÚLTIPLE

En las estructuras selectivas múltiples solo se pueden evaluar variables de tipo entero:

```
int Numero;
Numero=Integer.parseInt(JOptionPane
switch(Numero)
{
   case 1 : JOptionPane.showMessag
```

No se puede evaluar variables reales (double), de cadena (String) y ni de carácter (char).

La orden **break** hace que la ejecución salga del switch; de no ponerlo en cada case se ejecutarán todas las órdenes hacia abajo sin importar el valor de los case hasta encontrar un break o la llave de cierre del switch.

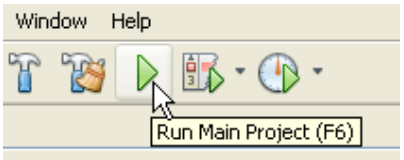
Nota

Los ejercicios 1 y 2 del Laboratorio 4 no se podrán llevar a NetBeans igual como está en el pseudocódigo, porque estariamos evaluando una **variable de tipo carácter.**

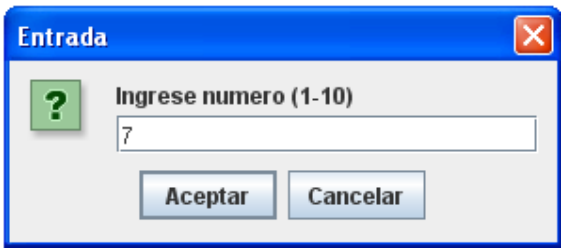
La realización de un algoritmo viene hacer la solución de un problema independiente del lenguaje de programación. En este caso el NetBeans solo evalua variables de valor tipo entero.

EJECUCIÓNDELPROGRAMA

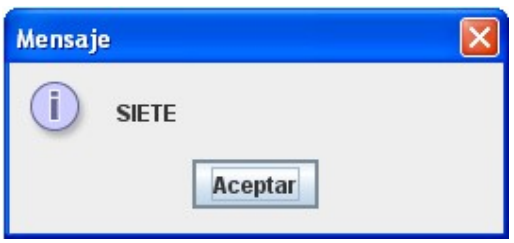
Ahora pasamos a ejecutar el programa para ver los resultados obtenidos.



Si ingresamos el siguiente dato:



Debemos de obtener el siguiente resultado:



PROYECTO DE ESTRUCTURA SELECTIVA ANIDADA

A continuación resolveremos el siguiente proyecto.

EJERCICIO 01
Leer un número entero y decir si es positivo, negativo o neutro.

Para resolver este ejercicio abriremos un nuevo proyecto.

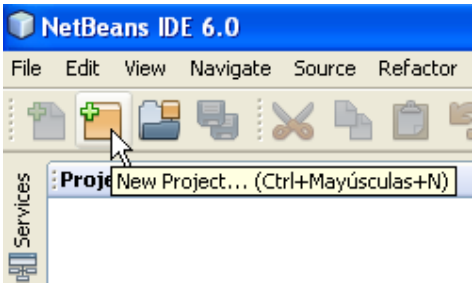


Figura 01: Eleccion de un nuevo proyecto

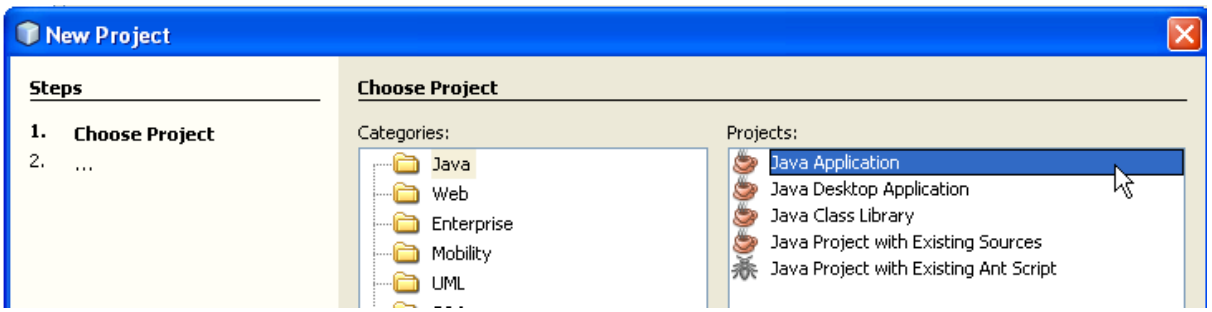


Figura 02: Eleccion de un proyecto Java Application

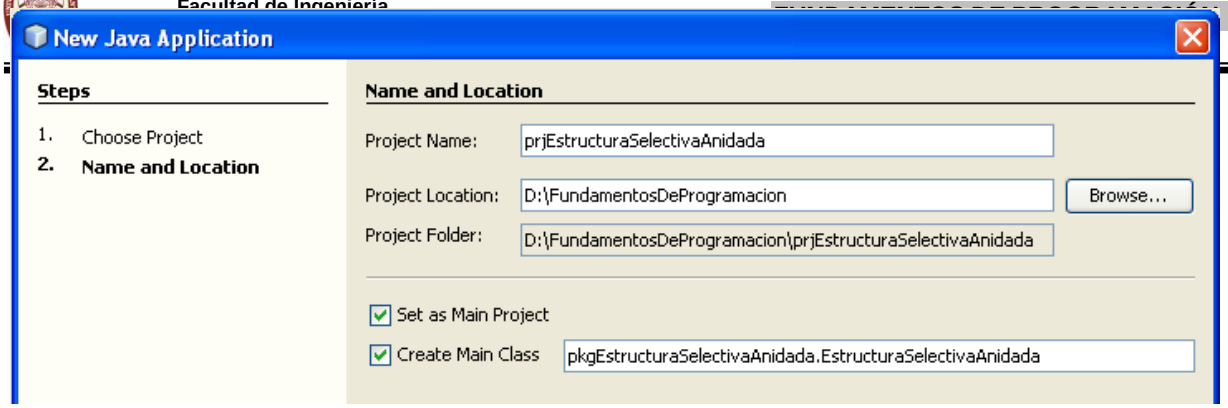


Figura 03: Ingresar nombre del proyecto, paquete y de la clase principal

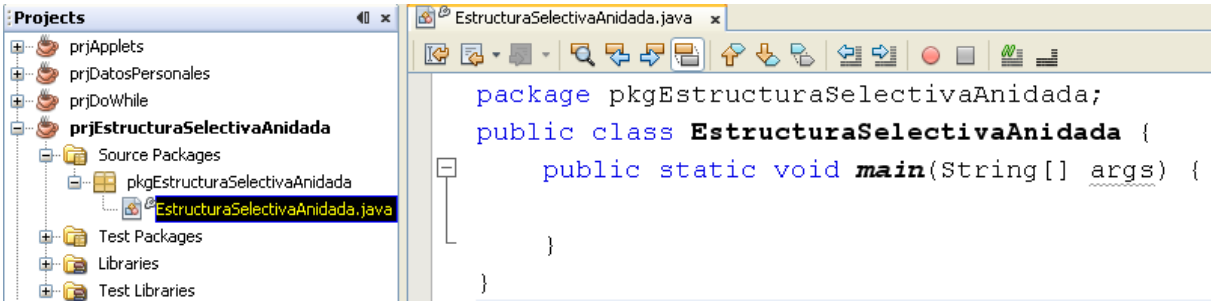


Figura 04: Proyecto listo para programar

Ahora estamos listo para comenzar con la programación de este ejercicio. Antes hay que recordar como esta resuelto en algoritmo.

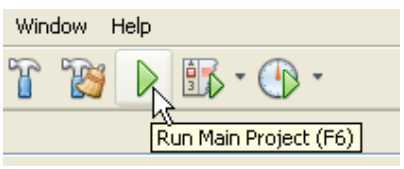
```
algoritmo Lab5Ejercicio01
var
        entero : num

inicio
        Leer num
        si (num = 0) entonces
                Escribir 'NÚMERO NEUTRO'
        si no
                si (num > 0) entonces
                        Escribir 'NÚMERO POSITIVO'
                sino
                        Escribir 'NÚMERO NEGATIVO'
                fin_si
        fin_si
fin
```

La solución en código de este ejercicio es el siguiente:

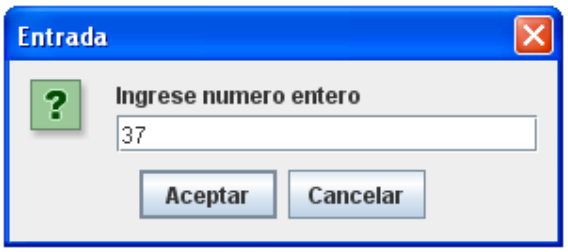
```
package pkgEstructuraSelectivaAnidada;
import javax.swing.JOptionPane;

public class EstructuraSelectivaAnidada {
   public static void main(String[] args) {
      int num;
      num=Integer.parseInt(JOptionPane.showInputDialog("Ingrese numero entero"));
      if(num==0)
         JOptionPane.showMessageDialog(null, "NUMERO NEUTRO");
      else
      {
         if(num>0)
            JOptionPane.showMessageDialog(null, "NUMERO POSITIVO");
         else
            JOptionPane.showMessageDialog(null, "NUMERO NEGATIVO");
      }
   }
}
```

```
package pkgEstructuraSelectivaAnidada;
import javax.swing.JOptionPane;

public class EstructuraSelectivaAnidada {
    public static void main(String[] args) {
        int num;
        num=Integer.parseInt(JOptionPane.showInputDialog("Ingrese numero entero"));
        if(num==0)
            JOptionPane.showMessageDialog(null, "NUMERO NEUTRO");
        else
        {
            if(num>0)
                JOptionPane.showMessageDialog(null, "NUMERO POSITIVO");
            else
                JOptionPane.showMessageDialog(null, "NUMERO NEGATIVO");
        }
    }
}
```

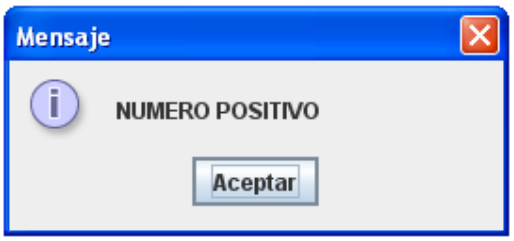
Comprendido el código de la estructura selectiva anidada en Java, ahora pasamos a ejecutar el programa para ver los resultados obtenidos.



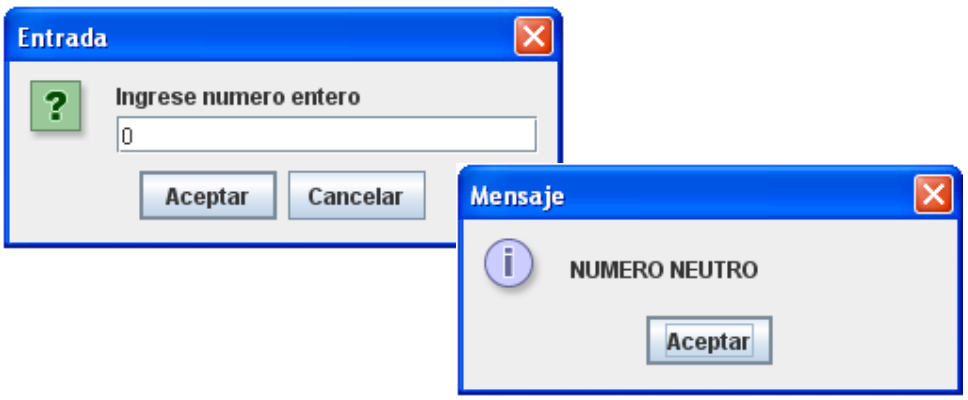
Si ingresamos el siguiente dato:

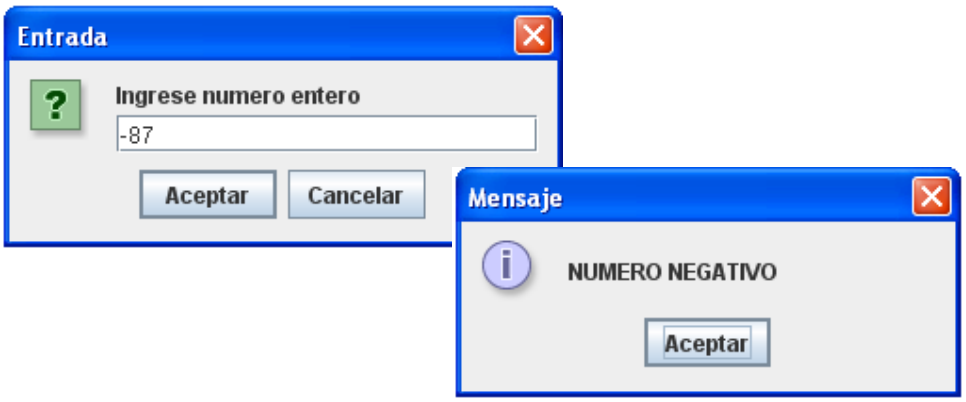


Debe de obtenerse el siguiente resultado:



También pueden obtenerse los siguientes resultados:





PROYECTO DE ESTRUCTURA REPETITIVA FOR

A continuación resolveremos el siguiente proyecto.

EJERCICIO 01

Diseñar un algoritmo que permita visualizar la tabla de multiplicar de un número entero ingresado por teclado. Por ejemplo:

1 x 5 = 5
2 x 5 = 10
3 x 5 = 15
……
12 x 5 = 60

Para resolver este ejercicio abriremos un nuevo proyecto.

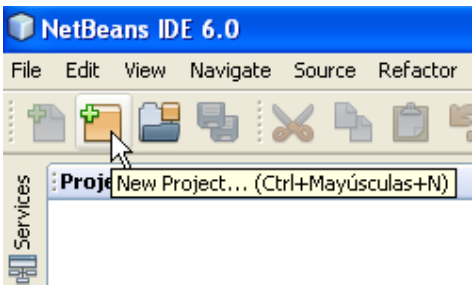


Figura 01: Eleccion de un nuevo proyecto

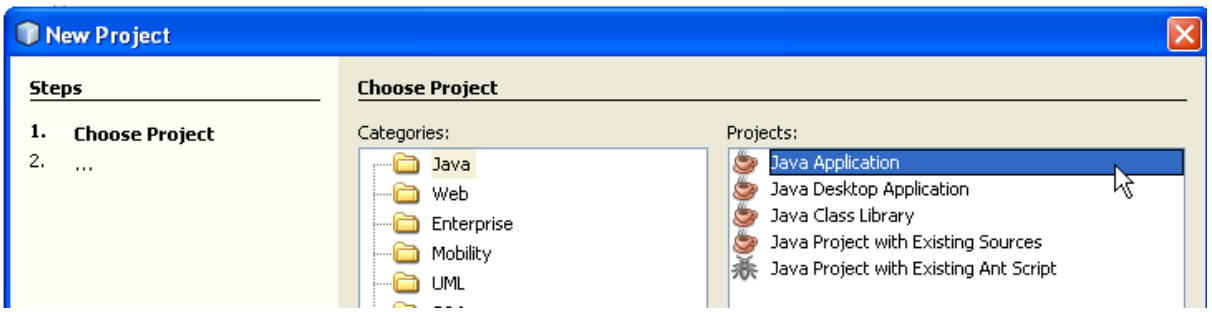


Figura 02: Eleccion de un proyecto Java Application

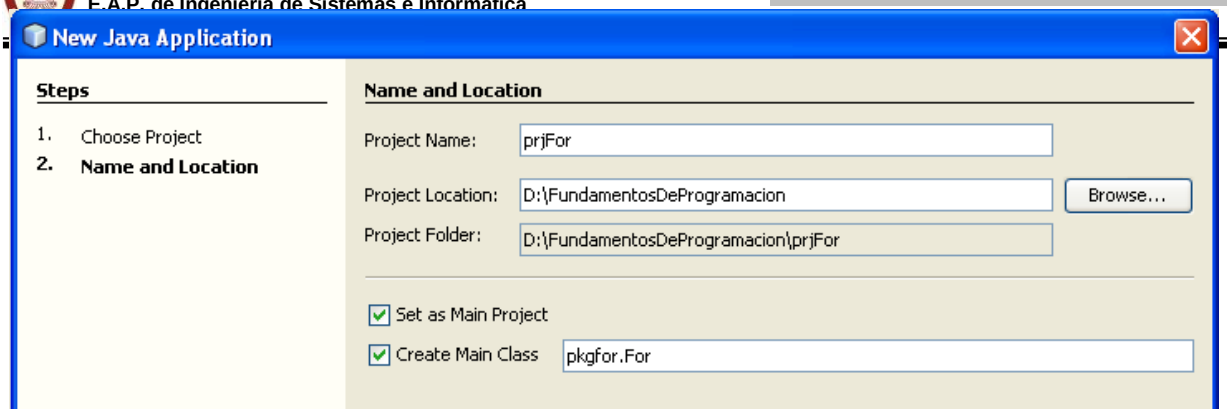


Figura 03: Ingresar nombre del proyecto, paquete y de la clase principal

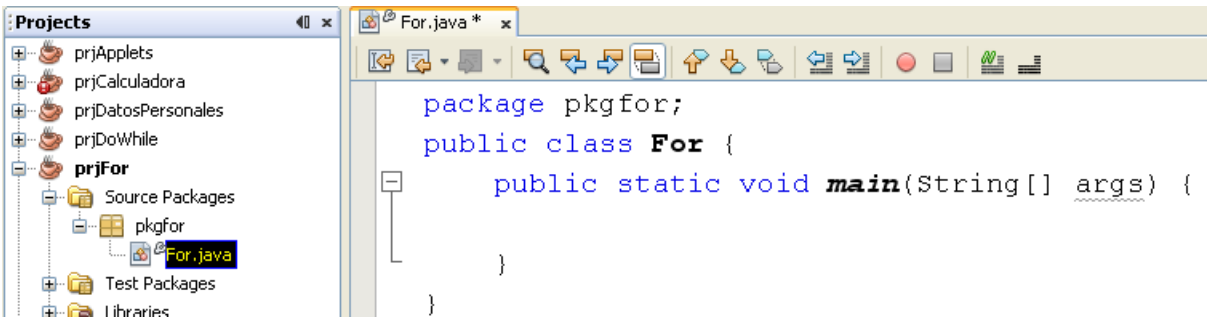


Figura 04: Proyecto listo para programar

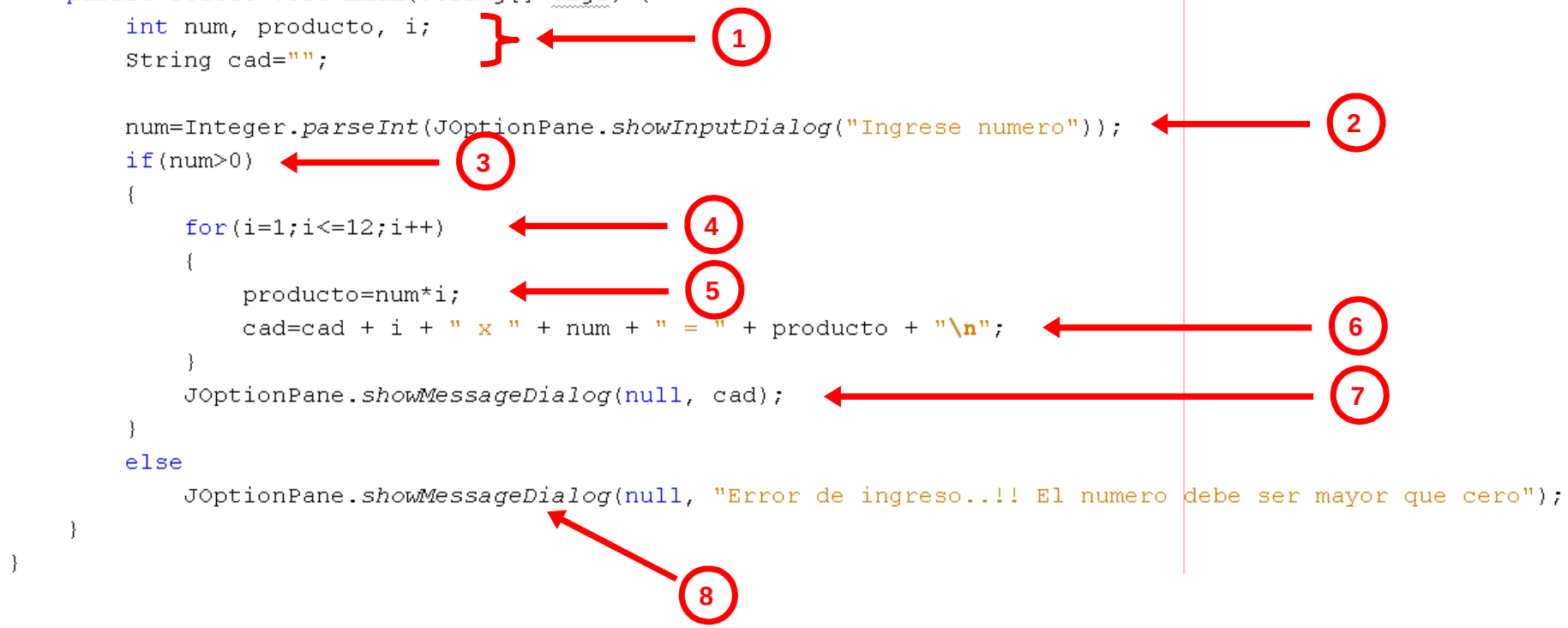
Ahora estamos listo para comenzar con la programación de este ejercicio. Antes hay que recordar como esta resuelto en algoritmo.

```
algoritmo Lab06Ejercicio01
var
    entero : num, producto, i
    cadena: cad
inicio
    Leer num
    cad = ' '
    si (num >0) entonces
        desde i =1 hasta 12 inc 1 hacer
            producto = i * num
            cad = cad, i , ' * ' , num, ' = ', producto
        fin_desde
        Mostrar (cad)
    sino
        Mostrar ('Error de ingreso...!! El Numero debe ser mayor que cero')
    fin_si
fin
```

La solución en código de este ejercicio es el siguiente:

```
package pkgfor;
import javax.swing.JOptionPane;

public class For {
   public static void main(String[ ] args) {
      int num, producto, i;
      String cad="";

      num=Integer.parseInt(JOptionPane.showInputDialog("Ingrese numero"));
      if(num>0)
      {
         for(i=1;i<=12;i++)
         {
            producto=num*i;
            cad=cad + i + " x " + num + " = " + producto + "\n";
         }
         JOptionPane.showMessageDialog(null, cad);
      }
      else
         JOptionPane.showMessageDialog(null, "Error de ingreso..!! El numero debe ser mayor que cero");
   }
}
```

```
package pkgfor;
import javax.swing.JOptionPane;
public class For {
    public static void main(String[] args) {
        int num, producto, i;
        String cad="";

        num=Integer.parseInt(JOptionPane.showInputDialog("Ingrese numero"));
        if(num>0)
        {
            for(i=1;i<=12;i++)
            {
                producto=num*i;
                cad=cad + i + " x " + num + " = " + producto + "\n";
            }
            JOptionPane.showMessageDialog(null, cad);
        }
        else
            JOptionPane.showMessageDialog(null, "Error de ingreso..!! El numero debe ser mayor que cero");
    }
}
```

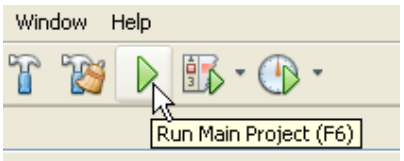


En este ejercicio de estructura repetitiva For nos mostrará la tabla de multiplicar de un número entero positivo.

La explicación del programa paso a paso es el siguiente

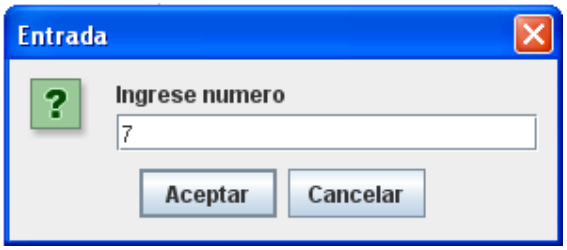
1. Declaración de variables; en estas instrucciónes también se pueden inicializar dichas variables. Inicializamos la variable **cad = " "**.
2. Ingreso de los datos de entrada.
3. Esta instrucción es una condición para determinar:
   3.1. Si la condición es verdadera, quiere decir que el número es positivo y se ingresa al cuerpo del **if**. Luego se ejecuta el paso 4.
   3.2. Si la condición es falsa se salta al paso 8.
4. Para ingresar al bucle del For, se llevarán acabo las siguientes acciones:
   4.1. Si es la primera vez que se ejecuta la instrucción del For, a la variable inicial de tipo entero que se le asigna un valor; para este ejemplo: **i = 1**.
   4.2. Para ingresar al bucle se evaluará la condición; para este ejemplo: **i <= 12**, si es verdad se ingresa y se pasa al paso 5, de lo contrario se pasa al paso 7.
5. En la variable **producto** se almacenarán los valores obtenidos del producto de num * i.
6. Esta instrucción es la mas importante porque en la variable **cad** almacenamos toda la tabla de multiplicar de un número **N**.
   6.1. Analizaremos primero esta parte del código: cad + i + " * " + num + " = " + producto, la variable **cad** al comenzar el programa se le inicializo como una cadena vacia, a esta se le concatena el valor de la variable **i** (De 1 a 12 dependiendo de la interación en que se encuentra), y tambien se le concatena el símbolo *****, el valor de la variable **num**, el símbolo **=** y el valor de la variable **producto** obtenido en el paso 5. Formando así una cadena de la siguiente forma, para este ejemplo: **1 x 5 = 5**
   6.2. Esta parte de la instrucción: + "\n" significa que a la cadena obtenida se le va a concatenar un salto de línea. Esto hace que el resultado que se va a mostrar salga línea por línea.
   6.3. Toda esta cadena concatenada se almacena en la variable **cad**, que nos seguirá sirviendo en las iteraciones del bucle como repositorio de toda la tabla de multiplicar.
   6.4. Terminado de realizar todas las instrucciones del cuerpo del For, se pasa al paso 4 (acción 4.2) pero antes se incrementa o decrementa la variable **i** un valor constante, esto depende de la instrucción del For, para este ejemplo: **i++**, significa que se incrementará el valor de uno en uno.
7. Se muestra el valor almacenado en la variable **cad**, que viene hacer la tabla de multiplicar de un número **N**.
8. Muestra el mensaje de "Error de ingreso..!!" por ser un número negativo y se acaba el programa.

Comprendido el código de la estructura repetitiva For en Java, ahora pasamos a ejecutar el programa para ver los resultados obtenidos.

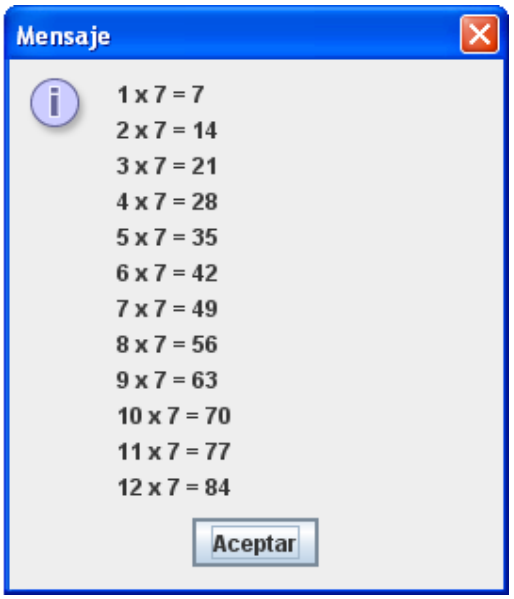


1ercaso

Ingresaremos un numero entero positivo:

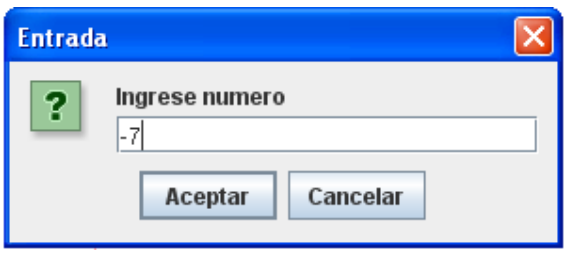


Debe de obtenerse el siguiente resultado:

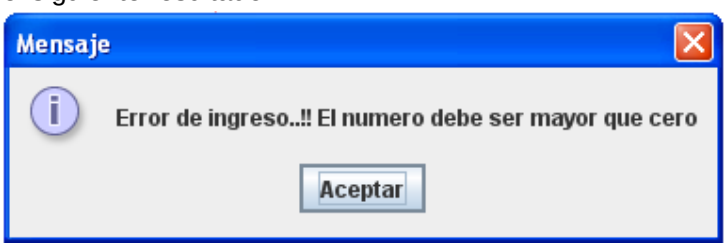


2docaso

Ingresaremos un numero entero negativo:



Debe de obtenerse el siguiente resultado:



PROYECTO DE ESTRUCTURA REPETITIVA DO WHILE

A continuación resolveremos el siguiente proyecto.

EJERCICIO 01

Realizar un programa para un Supermercado que calcule el total a pagar de un cliente, por la compra de varios productos, el ingreso de las compras debe continuar si digitamos el valor 1, y 2 se termina el ingreso y muestra el resultado.

Para resolver este ejercicio abriremos un nuevo proyecto.

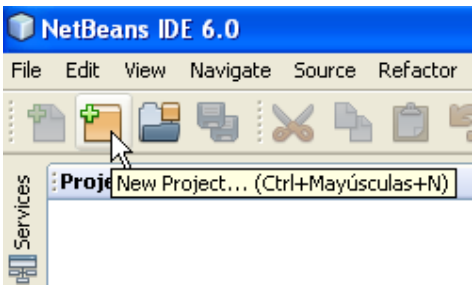


Figura 01: Eleccion de un nuevo proyecto

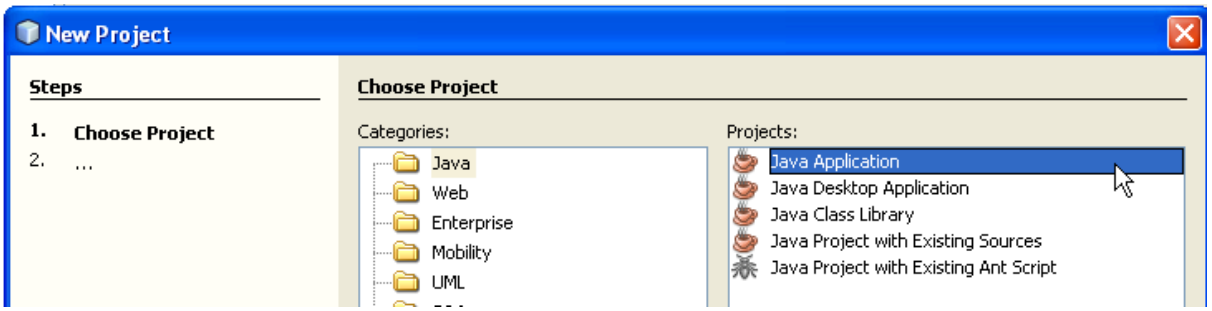


Figura 02: Eleccion de un proyecto Java Application

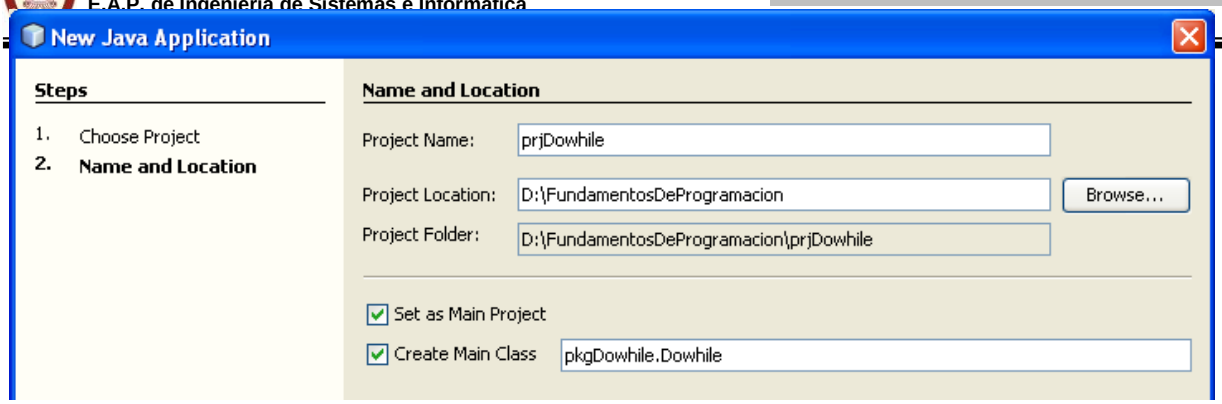


Figura 03: Ingresar nombre del proyecto, paquete y de la clase principal

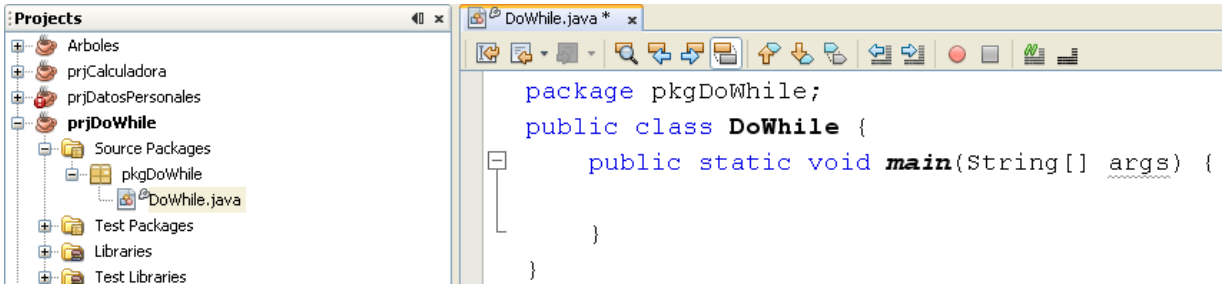


Figura 04: Proyecto listo para programar

Ahora estamos listo para comenzar con la programación de este ejercicio. Antes hay que recordar como esta resuelto en algoritmo.

```
algoritmo Ejercicio01
var
        entero : CantProducto, opcion, cont
        real : PrecioProducto, compra
inicio
        compra=0
        cont = 0
        Hacer
                cont = cont + 1
                Leer PrecioProducto, CantProducto
                compra = compra + PrecioProducto * CantProducto

                Escribir 'Desea continuar [si=1 , no=2] : '
                Leer opcion
        Mientras (opcion = 1)
        Mostrar ('La compra total por ', cont, ' productos es : ', compra)
fin
```

La solución en código de este ejercicio es el siguiente:

```
package pkgDoWhile;
import javax. swing.JOptionPane;
public class DoWhile {
   public static void main(String[ ] args) {
      int cantProducto,opcion,cont;
      double precioProducto,compra;

      compra=0;cont=0;
      do
      {
         //contador de productos ingresados
         cont=cont+1;
         //Ingreso de cantidad y precio del producto a comprar
         cantProducto=Integer.parseInt(JOptionPane.showInputDialog("Ingrese Cantidad del Producto " + cont));
         precioProducto=Double.parseDouble(JOptionPane.showInputDialog("Ingrese Precio Producto " + cont));
         //Obtencion del subtotal de la compra de uno o mas productos
         compra=compra + cantProducto * precioProducto;


         //Pregunta de si DESEA SEGUIR ingresando al bucle para seguir
         //acumulando el subtotal de los productos comprados.
         opcion=Integer.parseInt(JOptionPane.showInputDialog("Desea continuar[Si=1 , No=2]"));
      }while(opcion==1);
      JOptionPane.showMessageDialog(null, "El total a pagar por la compra de " + cont + " productos" +
                                          "\nes : " + compra);
   }
}
```

```
package pkgDoWhile;
import javax. swing.JOptionPane;
public class DoWhile {
  public static void main(String[] args) {
    int cantProducto,opcion,cont;
    double precioProducto,compra;

    compra=0;cont=0;
    do
    {
        //contador de productos ingresados
        cont=cont+1;
        //Ingreso de cantidad y precio del producto a comprar
        cantProducto=Integer.parseInt(JOptionPane.showInputDialog("Ingrese Cantidad del Producto " + cont));
        precioProducto=Double.parseDouble(JOptionPane.showInputDialog("Ingrese Precio Producto " + cont));
        //Obtencion del subtotal de la compra de uno o mas productos
        compra=compra + cantProducto * precioProducto;

        //Pregunta de si DESEA SEGUIR ingresando al bucle para seguir
        //acumulando el subtotal de los productos comprados.
        opcion=Integer.parseInt(JOptionPane.showInputDialog("Desea continuar[Si=1 , No=2]"));
    }while(opcion==1);
    JOptionPane.showMessageDialog(null, "El total a pagar por la compra de " + cont + " productos" +
                                  "\nes : " + compra);
    }
}
```

En este ejercicio de estructura repetitiva Do While podemos ingresar un sin número de compras siempre y cuando digitemos el valor **"1"** a la consulta "Desea continuar [si=1 , no=2] : ", si la respuesta es positiva el programa va acumulando las compras subtotales en la variable **compra**. Recordemos que una compra viene hacer el producto de la cantidad y el precio de un producto, es decir:

compra = PrecioProducto * CantProducto

Ahora la instrucción que me permite acumular los subtotales de las compras es:

compra = compra + PrecioProducto * CantProducto

Laexplicacióndelprogramapasoapasoeselsiguiente

1. Declaración de variables.
2. Se inicializa las variables **compra** y **cont** con un valor igual a cero.
3. **Do**, instrucción que determina el comienzo del bucle Do While.
4. Esta instrucción nos sirve de contador de cuantas compras vamos realizando.
5. Ingreso de los datos de entrada de la 1era compra.
6. Esta instrucción se ejecuta de la siguiente manera:
   8.1. Se multiplica el **PrecioProducto * CantProducto**
   8.2. El resultado se suma con el valor de la variable compra, esta variable fue inicializado con cero al comenzar el programa, es decir:

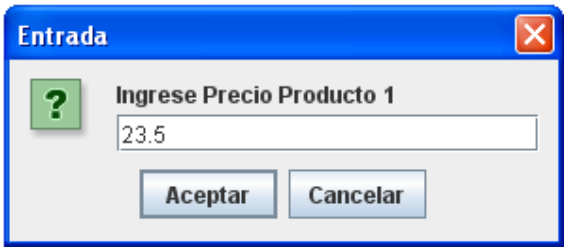
   0 + PrecioProducto * CantProducto

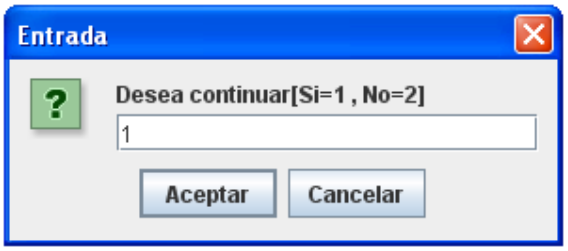
   8.3. El resultado de toda esta instrucción se almacena en la misma variable **compra**, que nos servirá como variable que va acumulando los subtotales de las compras.
7. En esta parte el programa nos pregunta si deseamos seguir ingresando compras.
8. Se evalua la condición del While, aquí pueden pasar dos cosas:
   8.1. Si la condición es **verdadera** vuelve a ingresar al bucle y se continua con:
      a. El paso 4, se incrementa el contador.
      b. El paso 5; ingresamos los valores de una segunda compra.
      c. El paso 6; se suma el valor de la primera compra que se tenia almacenado en la variable **compra** con el producto de los datos de entrada de la segunda compra, almacenando nuevamente el resultado en la variable *compra*.
      d. Se realiza el paso 7.
      e. Se realiza el paso 8, si vuelve a ingresar al bucle se incrementa el contador, se ingresan los datos de la 3era compra y asi sucesivamente hasta llegar al paso 9.
   8.2. Si la condición es **falso** se continua con el paso 9.
9. Se muestra el resultado total de los **N** productos comprados.

Comprendido el código de la estructura repetitiva Do While en Java, ahora pasamos a ejecutar el programa para ver los resultados obtenidos.

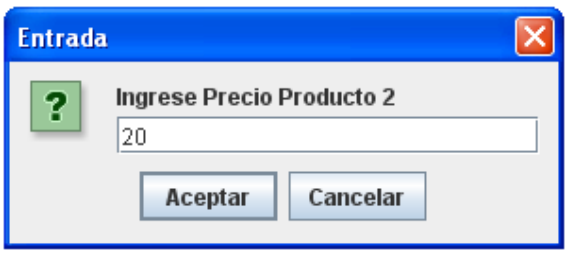
Ingresaremos los datos de tres compras:

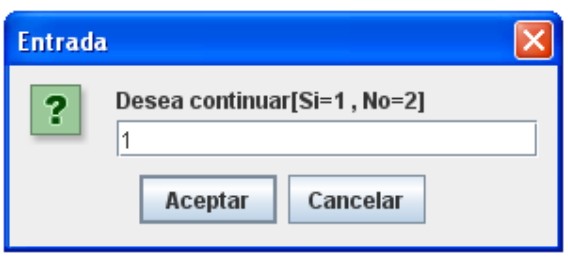
1eracompra



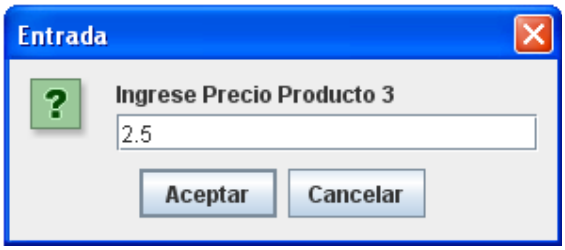


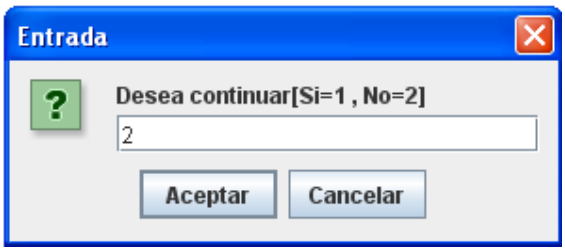
2dacompra



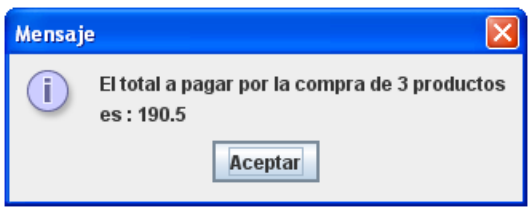


3eracompra





Debe de obtenerse el siguiente resultado:



PROYECTO DE ESTRUCTURA REPETITIVA WHILE

A continuación resolveremos el siguiente proyecto.

EJERCICIO 01

Diseñar un algoritmo que permita visualizar la tabla de multiplicar de un número entero ingresado por teclado. Por ejemplo:

1 x 5 = 5
2 x 5 = 10
3 x 5 = 15
……
12 x 5 = 60

Para resolver este ejercicio abriremos un nuevo proyecto.

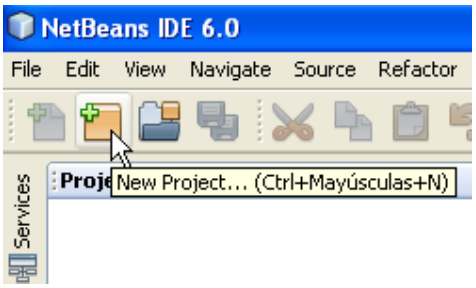


Figura 01: Eleccion de un nuevo proyecto

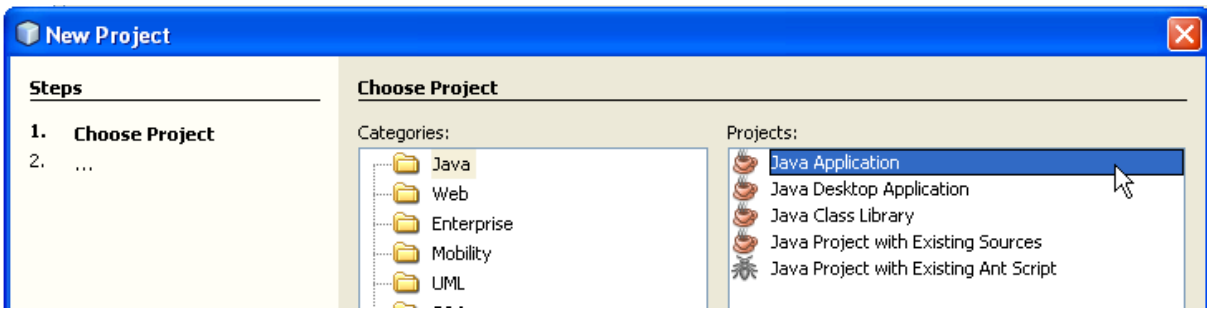


Figura 02: Eleccion de un proyecto Java Application

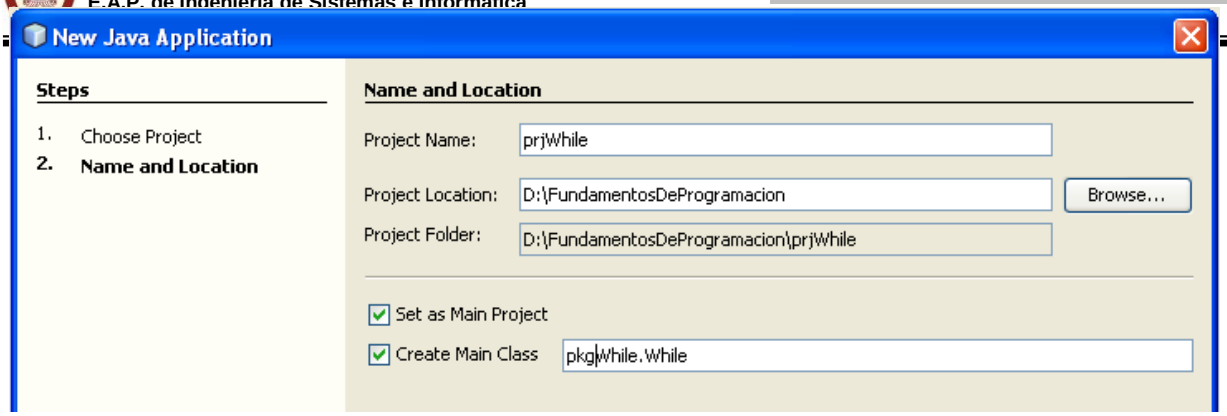


Figura 03: Ingresar nombre del proyecto, paquete y de la clase principal

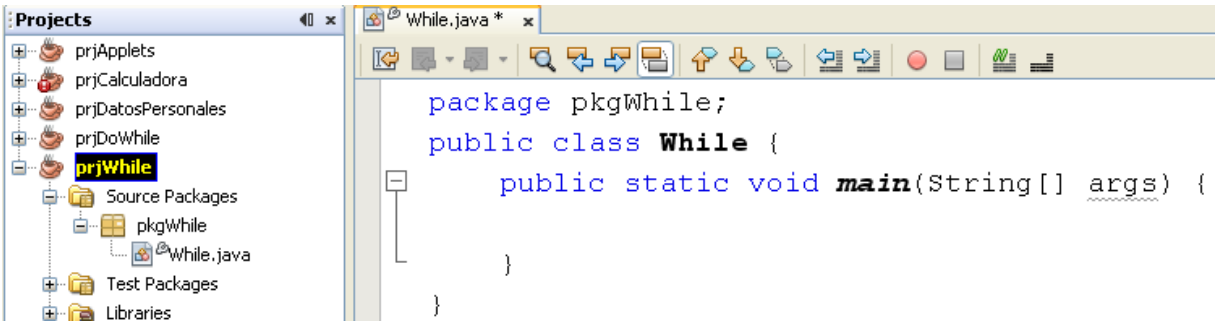


Figura 04: Proyecto listo para programar

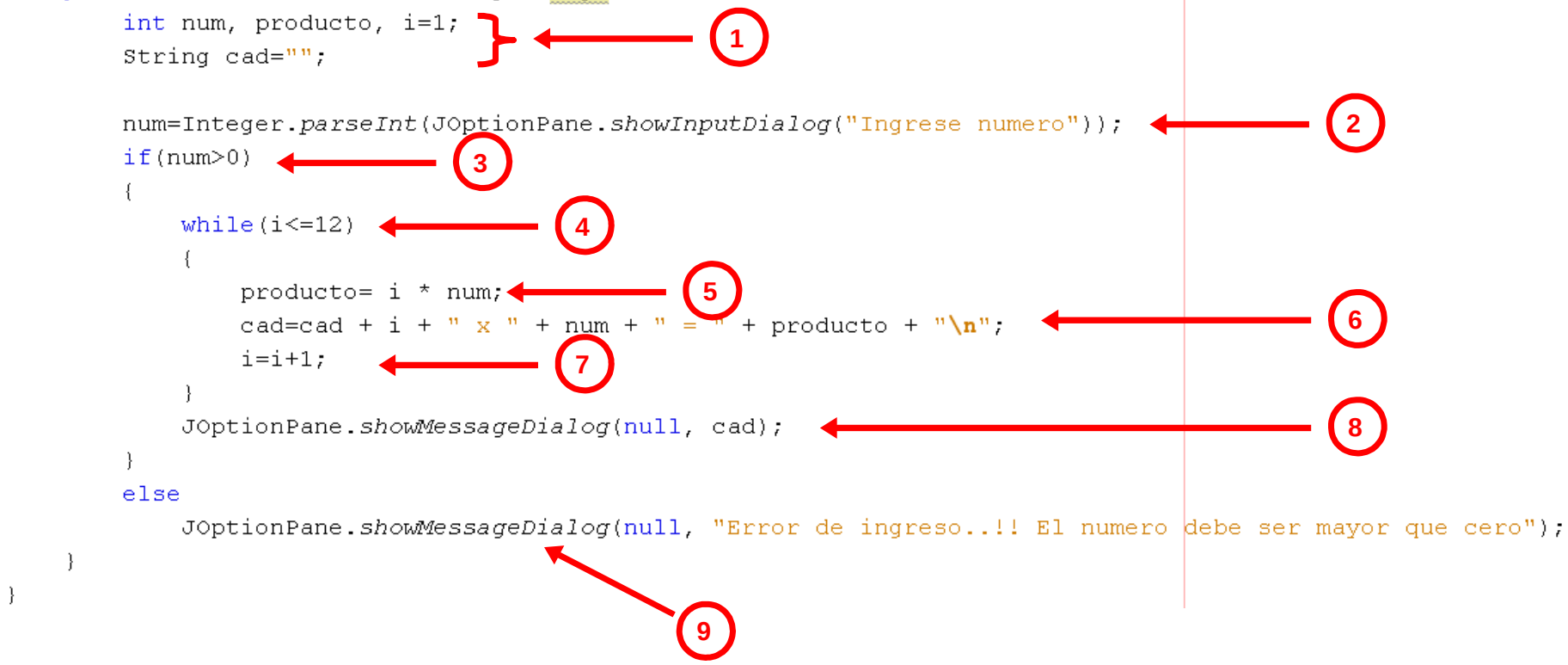
Ahora estamos listo para comenzar con la programación de este ejercicio. Antes hay que recordar como esta resuelto en algoritmo.

```
algoritmo Lab06Ejercicio01
var
    entero : num, producto, i
    cadena: cad
inicio
    Leer num
    i = 1
    cad = ' '
    si (num >0) entonces
        mientras(i <=12)
            producto = i * num
            cad = cad, i , ' * ' , num, ' = ', producto
            i = i + 1
        fin_mientras
        Mostrar (cad)
    sino
        Mostrar ('Error de ingreso...!! El Numero debe ser mayor que cero')
    fin_si
fin
```

La solución en código de este ejercicio es el siguiente:

```
package pkgWhile;
import javax.swing.JOptionPane;
public class While {
    public static void main(String[ ] args) {
        int num, producto, i=1;
        String cad="";

        num=Integer.parseInt(JOptionPane.showInputDialog("Ingrese numero"));
        if(num>0)
        {
            while(i<=12)
            {
                producto=num*i;
                cad=cad + i + " x " + num + " = " + producto + "\n";
                i=i+1;
            }
            JOptionPane.showMessageDialog(null, cad);
        }
        else
            JOptionPane.showMessageDialog(null, "Error de ingreso..!! El numero debe ser mayor que cero");
    }
}
```

```
package pkgWhile;
import javax.swing.JOptionPane;
public class While {
    public static void main(String[] args) {
        int num, producto, i=1;
        String cad="";

        num=Integer.parseInt(JOptionPane.showInputDialog("Ingrese numero"));
        if(num>0)
        {
            while(i<=12)
            {
                producto= i * num;
                cad=cad + i + " x " + num + " = " + producto + "\n";
                i=i+1;
            }
            JOptionPane.showMessageDialog(null, cad);
        }
        else
            JOptionPane.showMessageDialog(null, "Error de ingreso..!! El numero debe ser mayor que cero");
    }
}
```

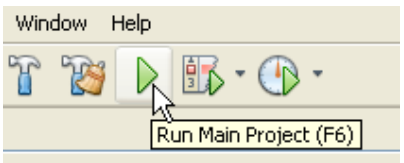


En este ejercicio de estructura repetitiva While nos mostrará la tabla de multiplicar de un número entero positivo.

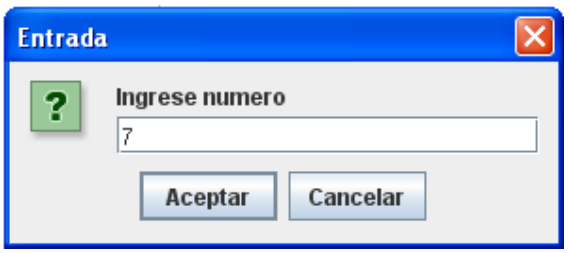
Laexplicacióndelprogramapasoapasoeselsiguiente

1. Declaración de variables; en estas instrucciónes también se pueden inicializar dichas variables. Inicializamos las variables **i = 1** y **cad = " "**.
2. Ingreso de los datos de entrada.
3. Esta instrucción es una condición para determinar:
   3.1. Si la condición es verdadera, quiere decir que el número es positivo y se ingresa al cuerpo del **if**. Luego se ejecuta el paso 4.
   3.2. Si la condición es falsa se salta al paso 9.
4. Se ingresa al bucle While siempre y cuando la condición sea verdadera. Si la condición es verdadera se sigue al paso 5, de lo contrario se pasa al paso 8.
5. En la variable **producto** se almacenarán los valores obtenidos del producto de num * i.
6. Esta instrucción es la mas importante porque en la variable cad almacenamos toda la tabla de multiplicar de un número **N**.
   6.1. Analizaremos primero esta parte del código: cad + i + " * " + num + " = " + producto, la variable **cad** al comenzar el programa se le inicializo como una cadena vacia, a esta se le concatena el valor de la variable **i** (De 1 a 12 dependiendo de la interación en que se encuentra), y tambien se le concatena el símbolo *****, el valor de la variable **num**, el símbolo **=** y el valor de la variable **producto** obtenido en el paso 5. Formando así una cadena de la siguiente forma, por ejemplo: **1 x 5 = 5**
   6.2. Esta parte de la instrucción: + "\n" significa que a la cadena obtenida se le va a concatenar un salto de línea. Esto hace que el resultado que se va a mostrar salga línea por línea.
   6.3. Toda esta cadena concatenada se almacena en la variable **cad**, que nos seguirá sirviendo en las iteraciones del bucle como repositorio de toda la tabla de multiplicar.
7. Se va incrementando el valor de la variable **i**, que es la que en algún momentos nos permitirá salir del bucle While. Al terminar esta instrucción se regresa al paso 4.
8. Se muestra el valor almacenado en la variable **cad**, que viene hacer la tabla de multiplicar de un número **N**.
9. Muestra el mensaje de "Error de ingreso..!!" por ser un número negativo y se acaba el programa.

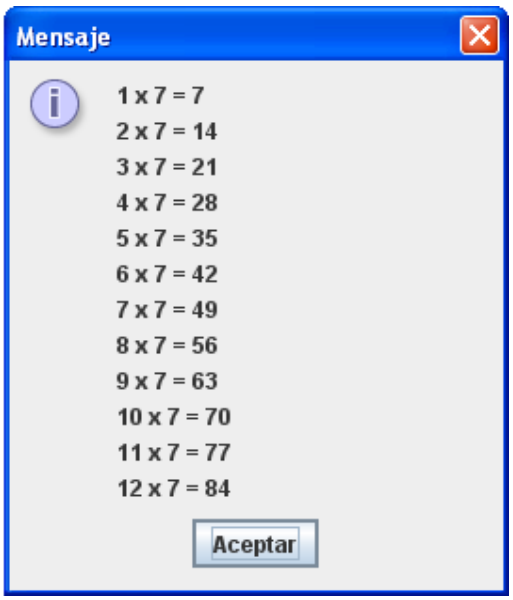
Comprendido el código de la estructura repetitiva While en Java, ahora pasamos a ejecutar el programa para ver los resultados obtenidos.



1ercaso
Ingresaremos un numero entero positivo:

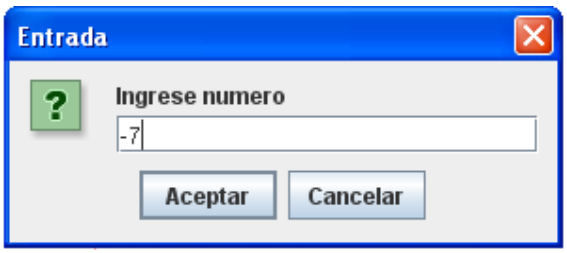


Debe de obtenerse el siguiente resultado:



2docaso

Ingresaremos un numero entero negativo:



Debe de obtenerse el siguiente resultado: