

Capítulo 1

Historia de la Programación

Lorem

Capítulo 2

.. Introducción a Lazarus Project Free Pascal

2.1 Instalación de Lazarus Project Free Pascal

Para descargar
el IDE puede acceder a través del siguiente link: <https://sourceforge.net/projects/lazarus/files/Lazarus%20Windows%2032%20bits/Lazarus%202.2.0/lazarus-2.2.0-fpc-3.2.2-win32.exe/download>

, este le descargará una aplicación que pesa aproximadamente 170 mb. Una vez descargada procedemos a su instalación haciendo doble clic sobre el ejecutable sobre el cual se mostrará la siguiente ventana.

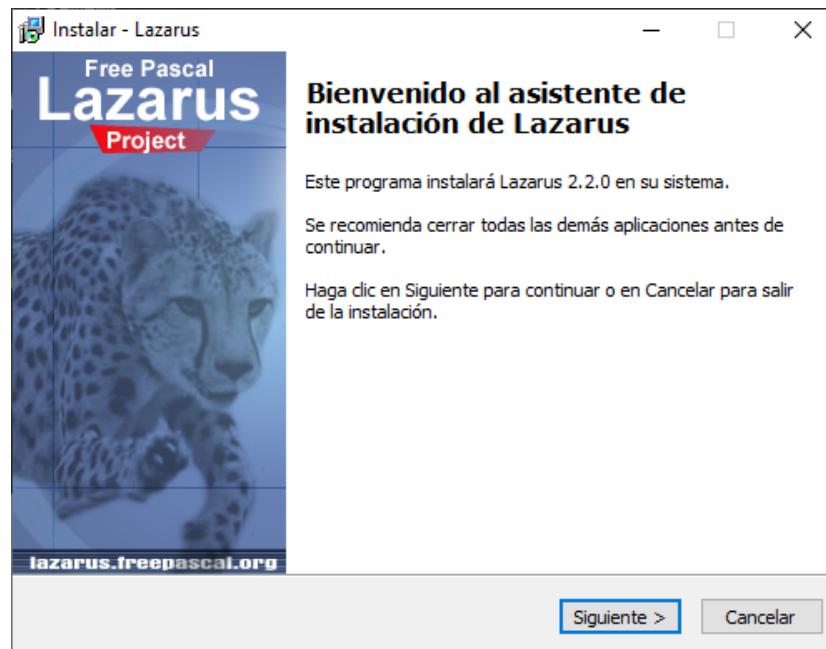


Figura 2.1: Ventana de instalación del IDE de desarrollo

Hacemos clic en [Siguiete], y se mostrará la ventana donde nos pide la ruta de instalación. Por defecto esta en la raíz del disco duro, la cual podemos dejar por defecto.

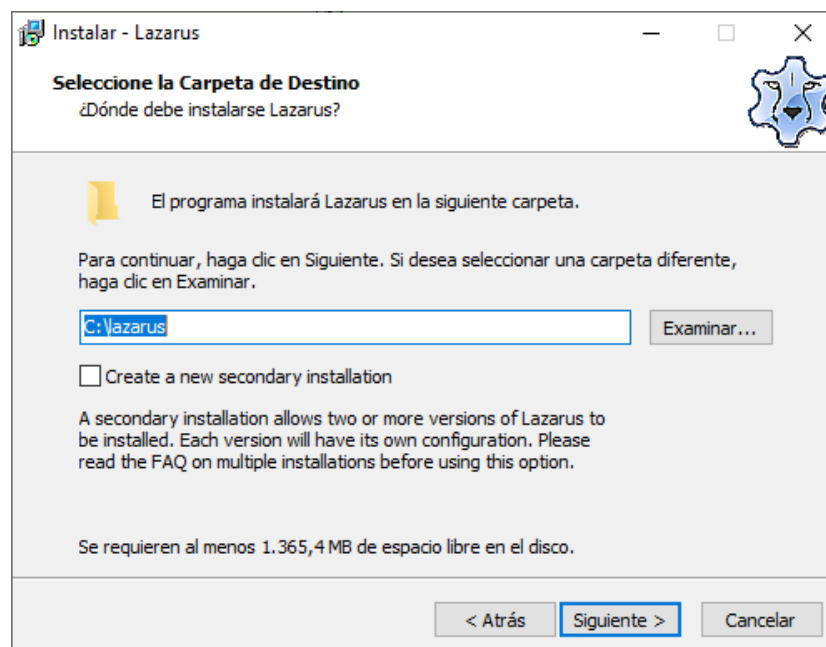


Figura 2.2: Ruta de instalación

Hacemos clic en [Siguiente], nos preguntará por el tipo de instalación, el cual dejaremos en la opción de [Instalación Completa]

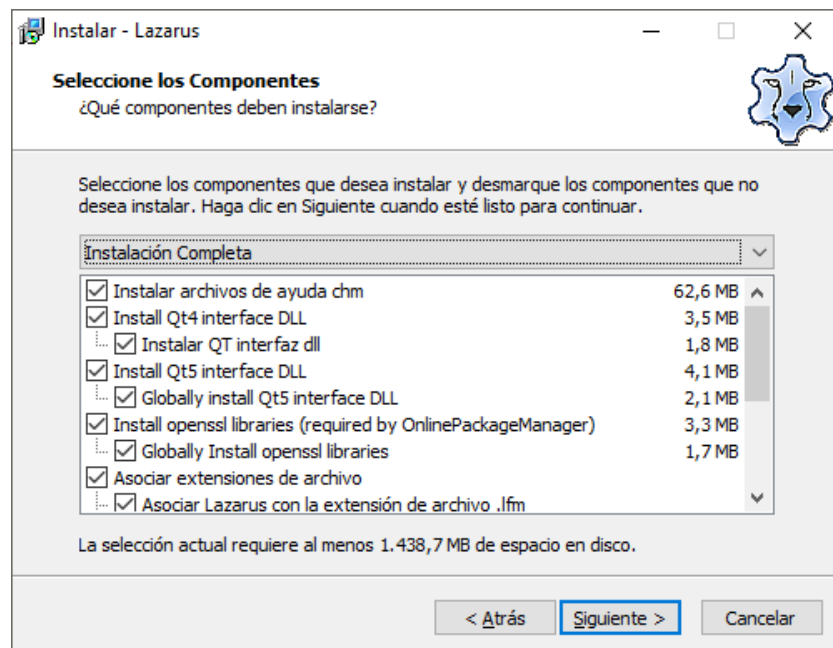


Figura 2.3: Tipo de Instalación

Hacemos clic en [Siguiente], y nos preguntará por el nombre del acceso directo, el cual dejaremos por defecto como lo indica el instalador. Sin embargo, usted puede seleccionar una ruta diferente en donde quiere que el instalador cree los accesos directos

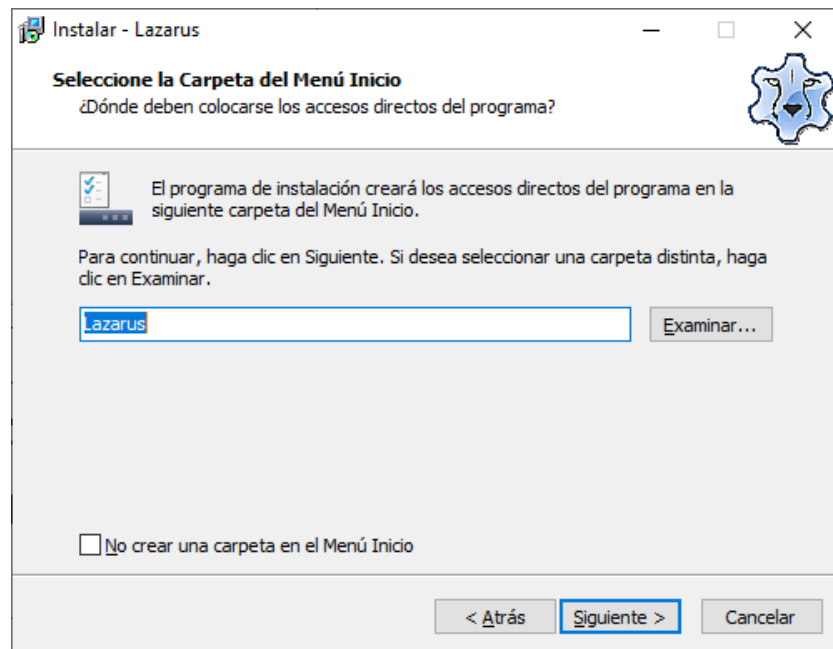


Figura 2.4: Accesos directos

Hacemos clic en [Siguiente], y el asistente nos preguntará si queremos crear un acceso directo en el escritorio, opción que puede ser marcada si así lo queremos, pero, este acceso lo podemos encontrar también en el menú inicio en el sistema operativo Windows

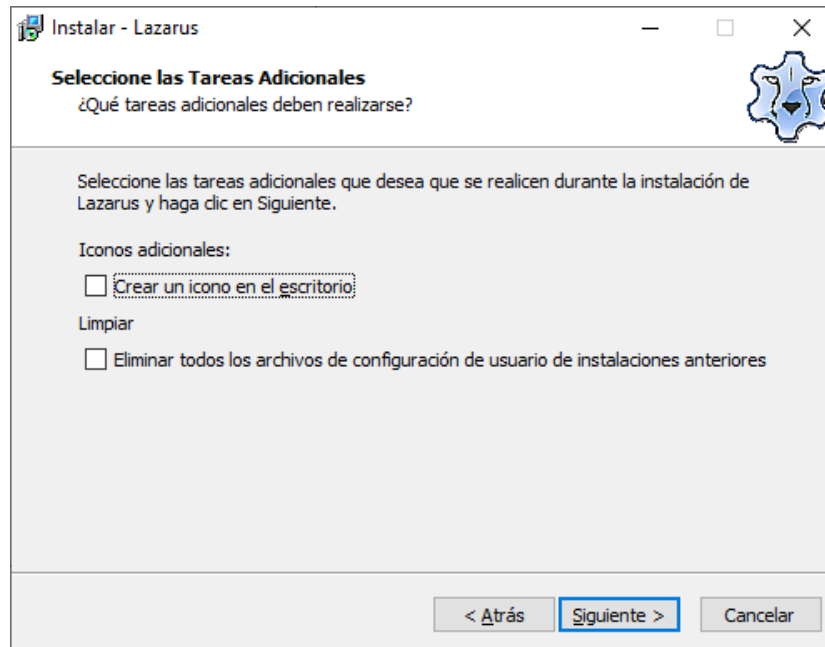


Figura 2.5: ¿Desea crear el acceso directo en el escritorio?

Hacemos clic en [Siguiente] y ya estamos listo para instalar

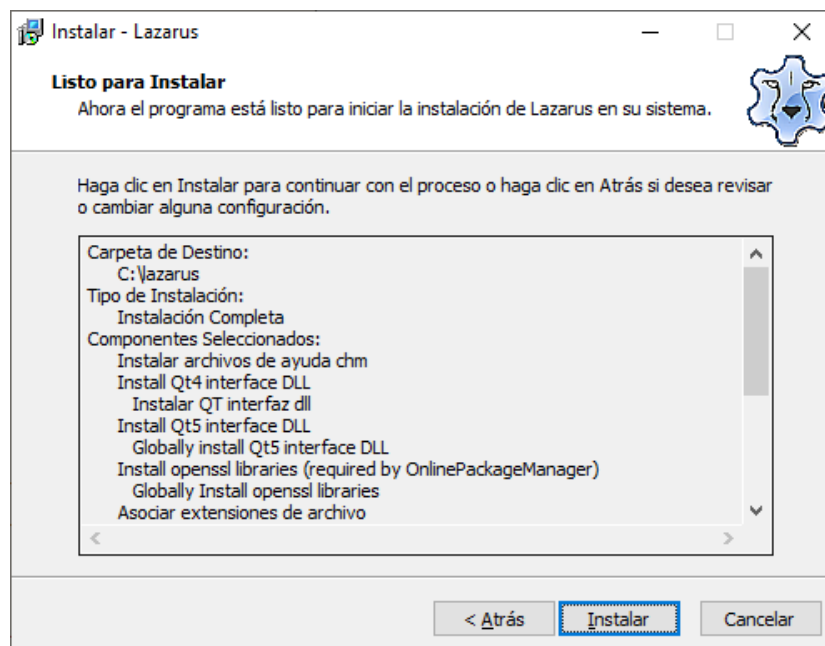


Figura 2.6: Programa listo y configurado para instalar

Hacemos clic en [Instalar] y esperamos al proceso de instalación

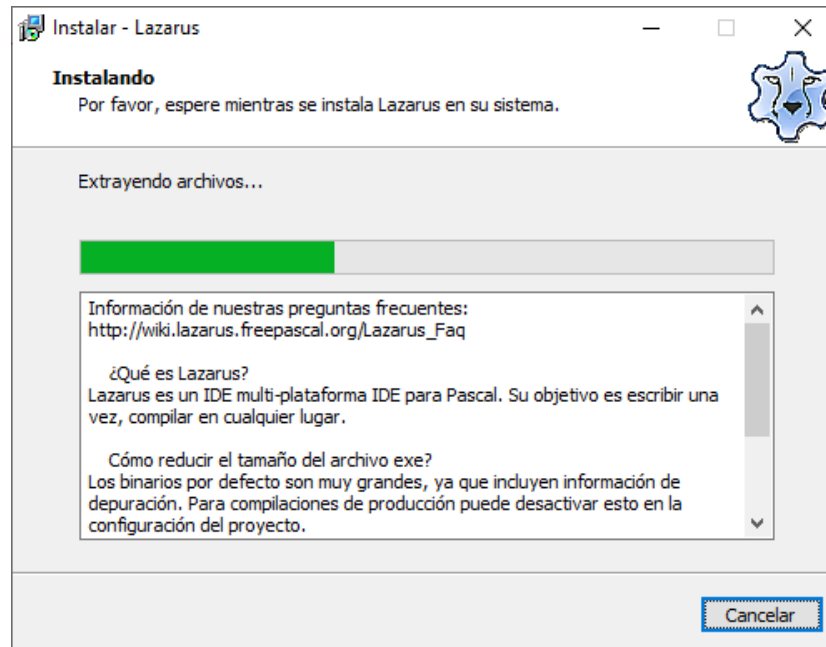


Figura 2.7: Proceso de instalación

Hacemos clic en [Finalizar]

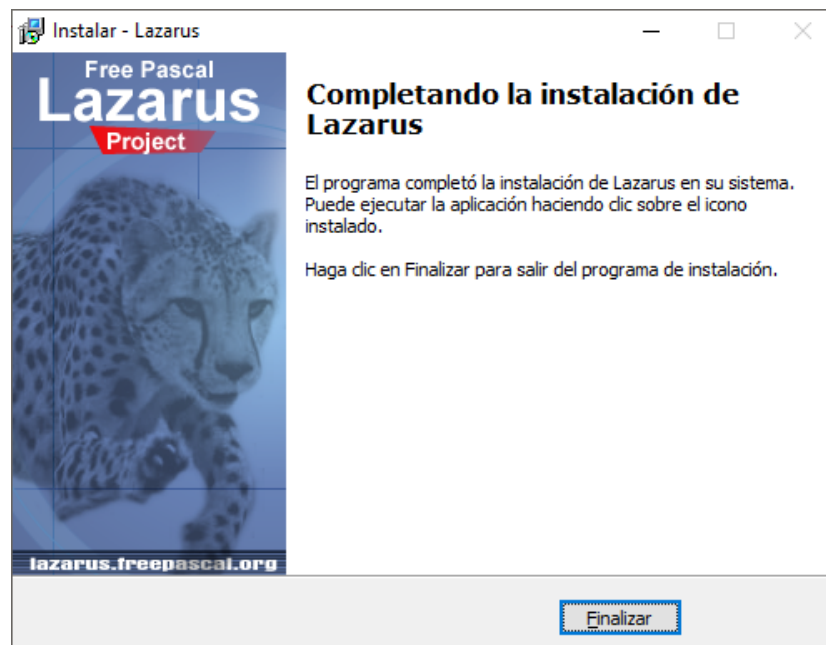


Figura 2.8: Finalización de la instalación

2.2 El Entorno de Lazarus

Para realizar nuestro primer programa, vamos a buscar la instalación en el menú inicio

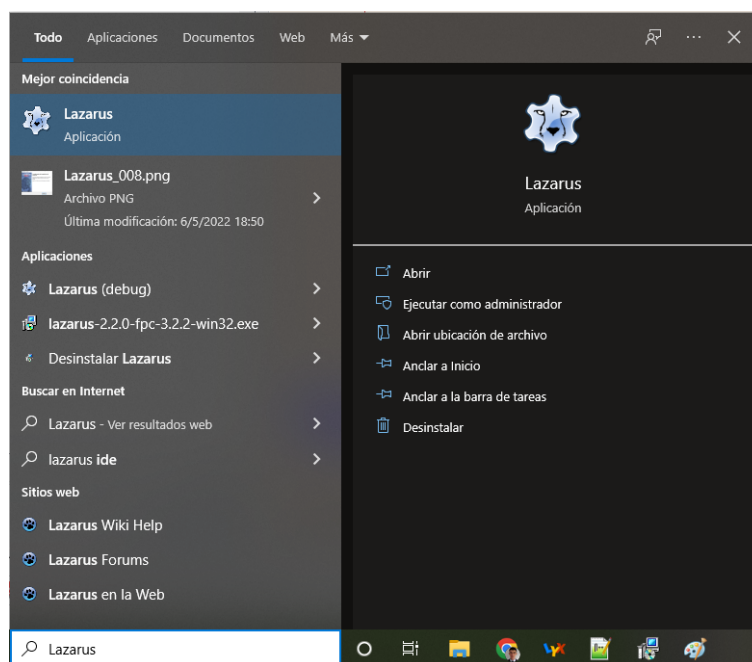


Figura 2.9: Buscando el IDE de Lazarus en el Inicio

Hacemos clic sobre la aplicación [Lazarus], que nos abrirá la siguiente ventana

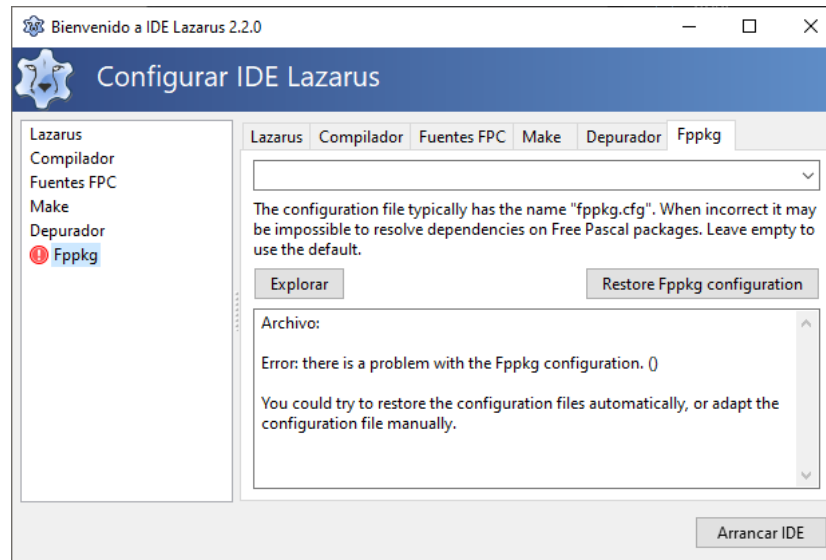


Figura 2.10: Ventana de inicio del IDE de Lazarus

Hacemos clic en [Arrancar IDE], y encontraremos varias ventanas, que vienen por defecto.

2.2.1 Menú principal

La ventana de menú principal contiene los menús para crear los proyectos, los botones rápidos para crear, abrir, guardar y ejecutar los proyectos, además contiene la pentaña de componentes por medio de la cual podemos diseñar nuestros aplicaciones.

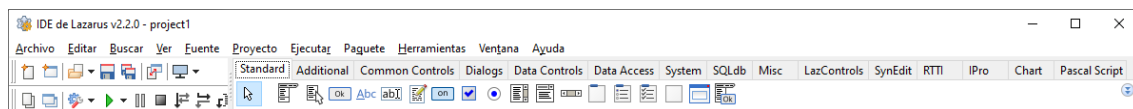


Figura 2.11: Ventana de menú principal

2.2.2 Inspector de Objetos

A través del inspector de objetos, podemos cambiar las propiedades de los diferentes elementos del formulario. Es una ventana que muestra las propiedades más importantes de cada elemento del formulario que estamos diseñando y que nos permite cambiarlo rápidamente.

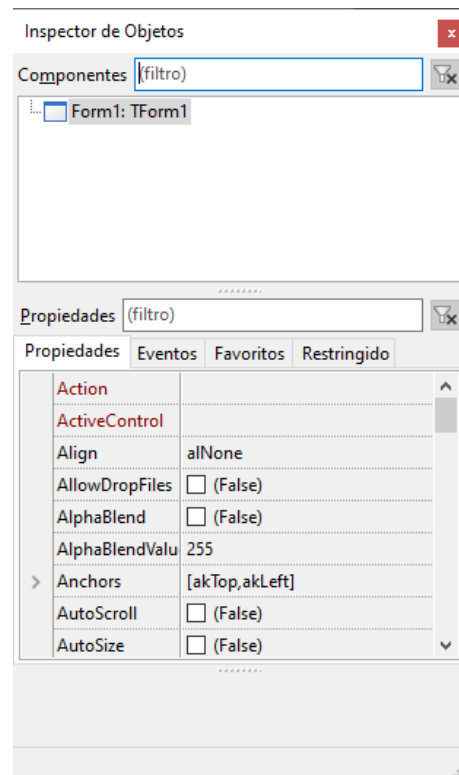


Figura 2.12: Inspector de Objetos

2.2.3 Editor de Código Fuente

El editor de código fuente, es la ventana en donde vamos a programar todo nuestro código.



Figura 2.13: Editor de Código Fuente

2.2.4 Ventana de Mensajes

La ventana de mensajes, se activa cuando ejecutamos el código, y en ella podemos ver los errores de código que tenemos en nuestro programa.

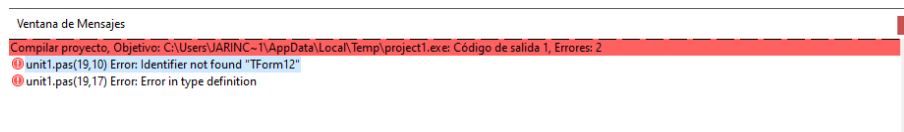


Figura 2.14: Ventana de mensajes

Estas ventanas que hemos visto hasta ahora siempre están desacopladas o sueltas. A algunos usuarios no les gusta este diseño, y por fortuna Lazarus trae una herramienta que nos permite tener una sola ventana.

2.3 Unificando las ventanas de Lazarus

Si así lo desea, puede unificar las ventanas del IDE de Lazarus siguiendo estos pasos.

1. Búsque en el menú principal la opción Paquete ▷Abrir archivo de paquete (.lpk) como se muestra en la siguiente imagen

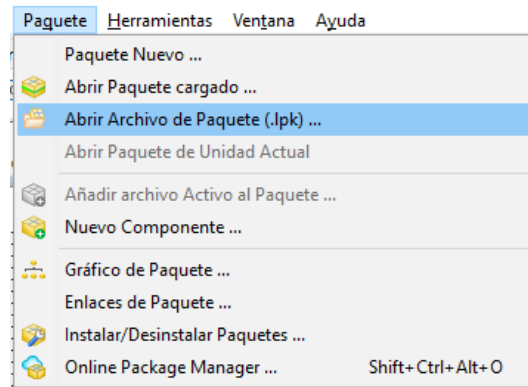


Figura 2.15: Accediendo al menú de paquetes

2. Al hacer clic sobre la opción [Abrir Archivo de Paquete (.lpk) ...] se abrirá una ventana de selección de archivos y carpetas en la cual debemos buscar la siguiente ruta: C :¹, allí debemos seleccionar el archivo [anchordocking.lpk] tal como se muestra en la imagen y hacer clic en el botón [Abrir]

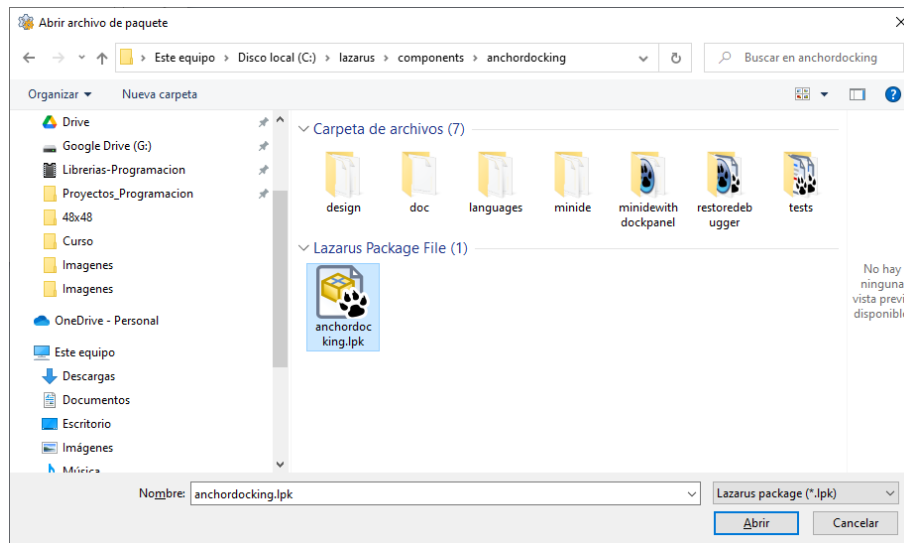


Figura 2.16: Seleccionando el archivo anchordocking.lpk

¹Nota: Esto, si no cambiamos la ruta por defecto de instalación de lazarus. Pero si la cambiamos al momento de la instalación, debemos buscar la ruta de instalación que elegimos y seguir las subcarpetas que se muestran . . .

3. Al abrilo veremos la siguiente ventana

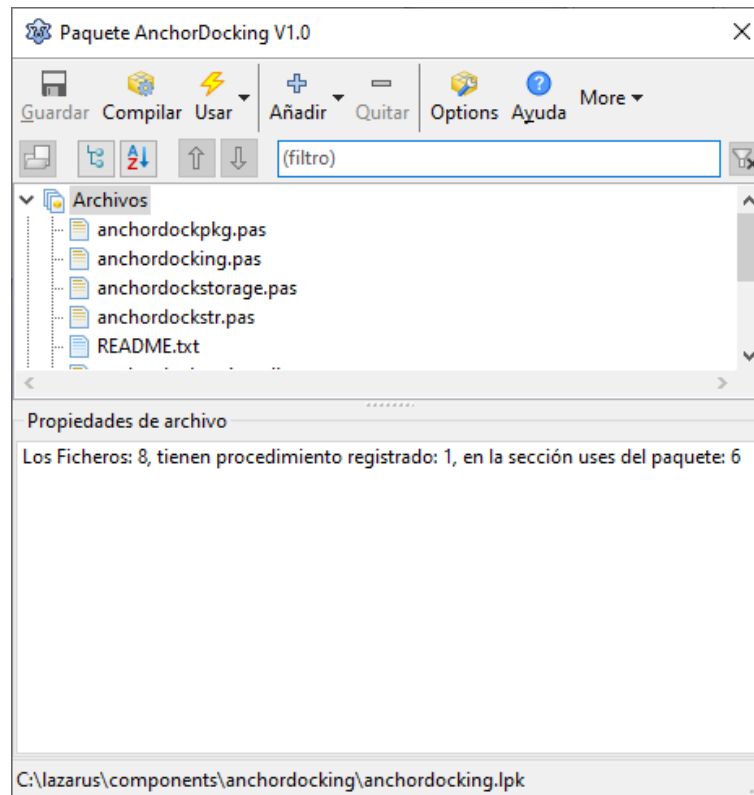
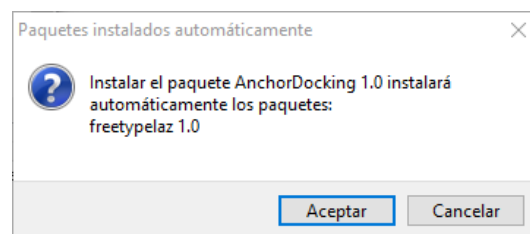
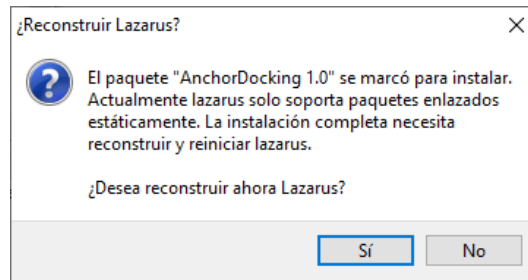


Figura 2.17: Ventana de instalación del componente

4. Procedemos a instalar haciendo clic sobre el botón [Usar] y luego en la opción de [Instalar] y [Aceptamos] el mensaje que nos presenta Lazarus



luego presianamos [Si] en la advertencia de Instalación



5. El componente se empezará a instalar, mostrando unos procesos en la ventana de mensajes. En caso de que hallamos hecho algun cambio en algun proyecto que estemos ejecutando, nos preguntará si queremos guardar los cambios, a lo cual responderemos que [Si]. El IDE se cerrará solo y se abrirá de nuevo, mostrando como lo es habitual en cada inicio la siguiente ventana.

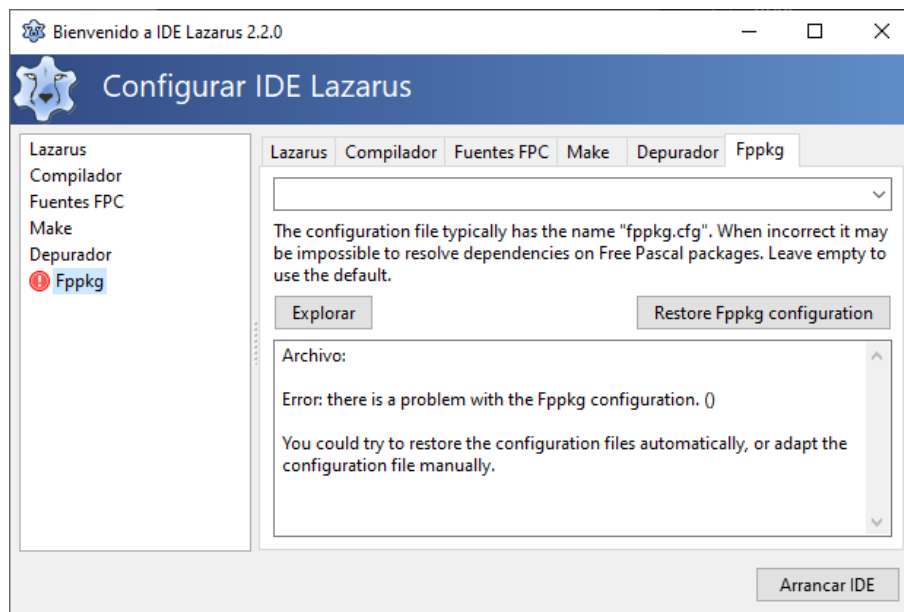


Figura 2.18: Ventana de inicio del IDE de Lazarus

hacemos clic en [Arrancar IDE].

6. Repetimos los mismos pasos 2 y 3 pero en esta ocasión debemos buscar el componente [anchordockingdsgn.lpk] que se encuentra dentro de la carpeta [design] de la ruta que abrimos inicialmente. Es decir que ahora el siguiente archivo que queremos abrir esta en la ruta por defecto C:.. Seleccionamos el archivo y veremos la siguiente ventana

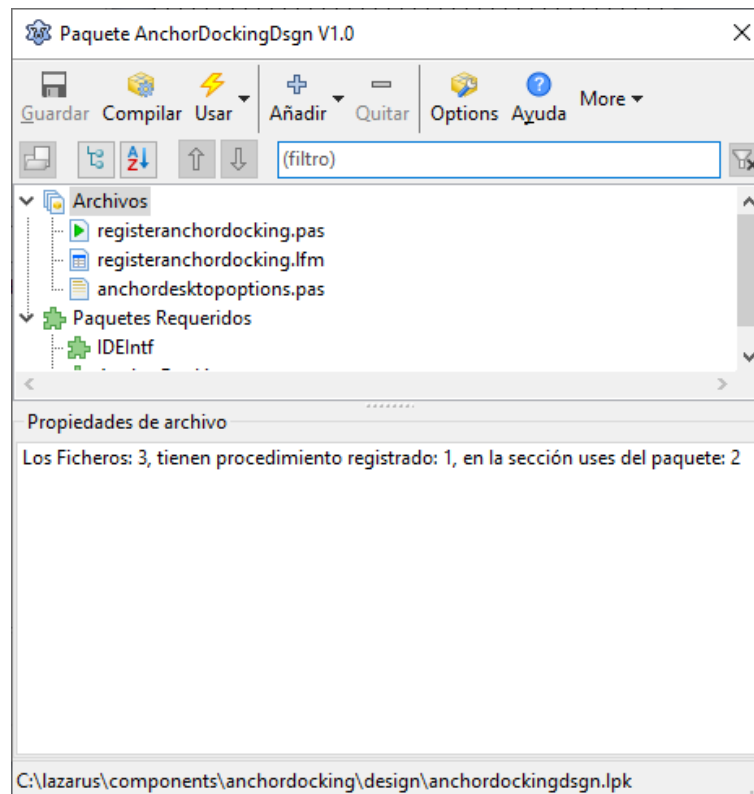
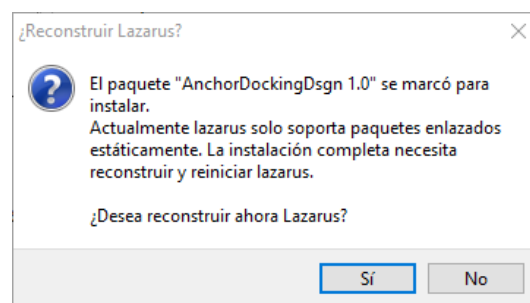


Figura 2.19: Instalación del componente para diseño

7. Buscamos el botón [Usar], luego [Instalar], aceptamos la [Advertencia]



esperamos a que Lazarus realice la instalación, se cierre y se vuelva a abrir.

8. Hacemos clic en la opción de [Arrancas IDE] y ahora vemos una pantalla totalmente unificada, tal como se muestra en la siguiente imagen.

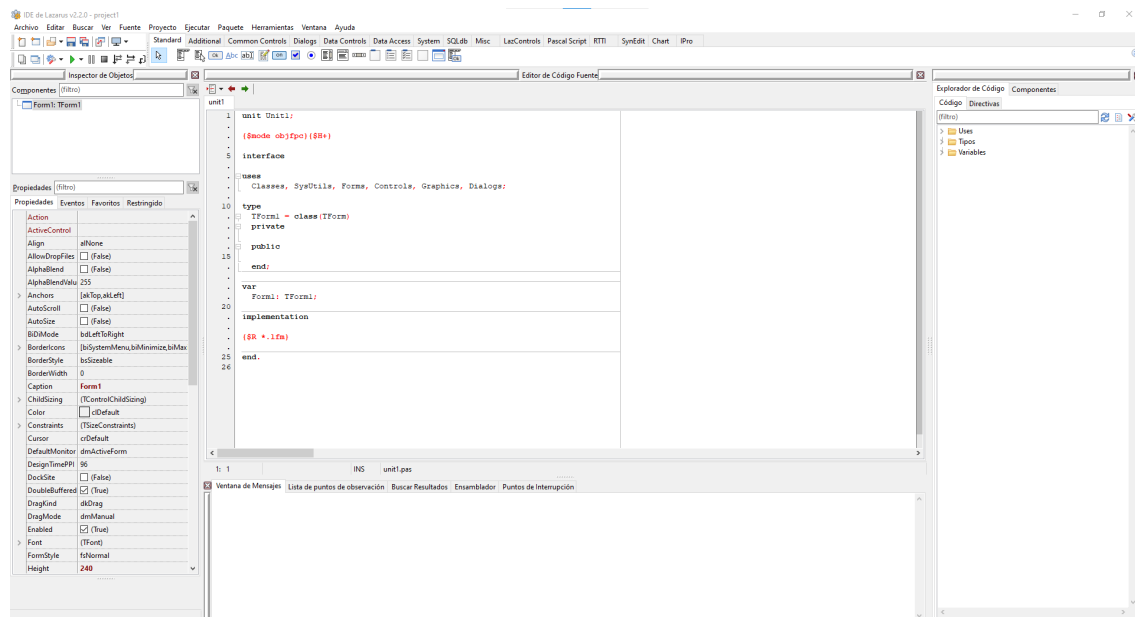


Figura 2.20: Ventana unificada del IDE de Lazarus

2.3.1 Conclusión

Al poder realizar este cambio sobre la aplicación, podemos notar que no solamente realizamos este cambio, si no que también podemos hacer muchos otros cambios, puesto que Lazarus es de código abierto y usa el mismo lenguaje pascal para compilarse así mismo, de tal forma que podemos realizar todos los cambios que queramos a nuestro IDE para mejorar su apariencia o funcionamiento.

2.4 Tipos de Programas

Lazarus contiene varios tipos de programas, aplicaciones y tipos de proyectos que se pueden desarrollar, estos los podemos visualizar al hacer clic en el menú Archivo > Nuevo ..., tal como se muestra en la siguiente imagen

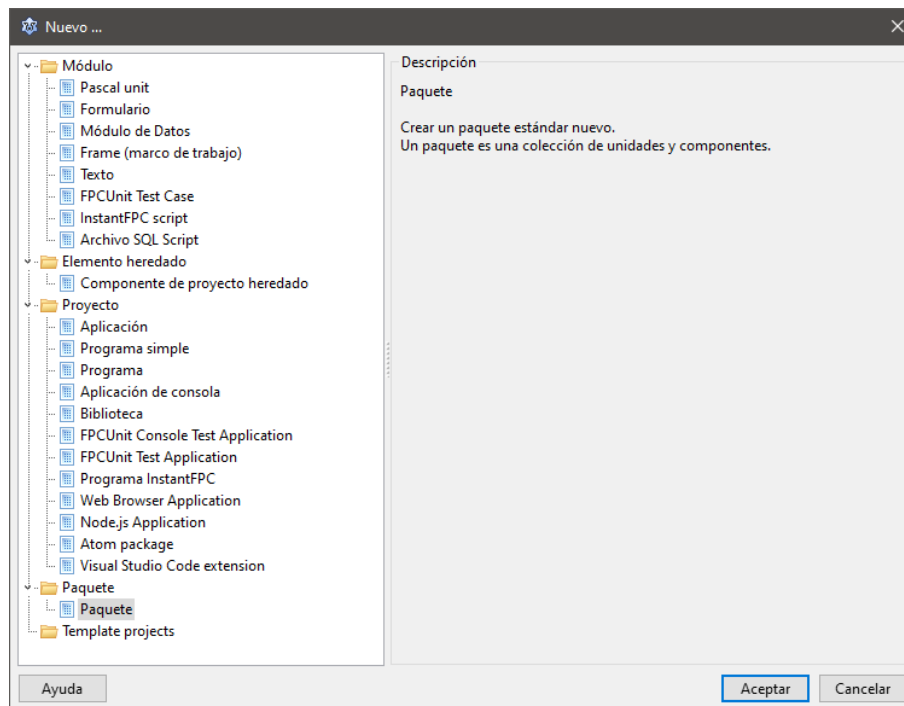


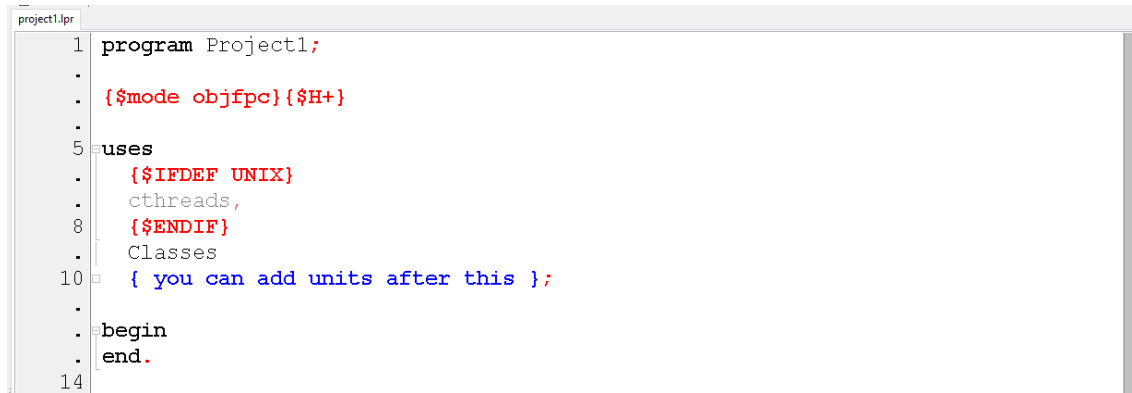
Figura 2.21: Tipos de proyectos en Lazarus

podemos observar cinco tipos de opciones, entre las cuales tenemos:

- **Módulo:** Son tipos de archivos que se pueden agregar a un proyecto determinado.
- **Elemento heredado:** Permite crear un componente heredado de algún proyecto o formulario.
- **Proyecto:** En este elemento encontramos los tipos de proyectos más destacados, los más comunes son, Aplicación y Programa, sin embargo hay otros muy útiles como las Bibliotecas que permiten crear archivos DLL con nuestras propias funciones, además de proyectos de otros lenguajes de programación como Node.js
- **Paquete:** Permite crear paquetes con estructuras especiales que podemos usar en otros proyectos. Los podemos ver como componentes que contienen código encapsulado que podemos estar reutilizando en muchos otros proyectos.
- **Template projects:** Permite diseñar ejemplos y usarlos como plantillas para nuevos proyectos.

2.5 Primer programa

Vamos a diseñar nuestro primer programa, para ello vamos a Archivo ▸ Nuevo ... y seleccionamos la opción [Programa], esto nos creará el siguiente código



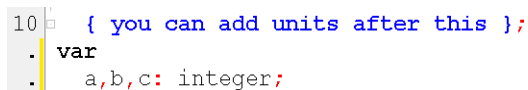
```
project1.ppr
1  program Project1;
.
.  {$mode objfpc}{$H+}
.
5  uses
.    {$IFDEF UNIX}
.    cthreads,
8    {$ENDIF}
.    Classes
10   { you can add units after this };
.
.  begin
.  end.
14
```

Figura 2.22: Código de un programa

La estructura que observamos es muy sencilla:

- la palabra reservada **program** y el nombre del programa *Project1*
- la definición de que el programa correrá o se compilará en el modo **objfpc** es decir object free pascal
- la declaración **uses**, para definir las librerías que usará y una **directiva condicional** para definir el tipo de sistema operativo en el que se ejecutará
- las palabras reservadas **begin** y **end**, en medio de las cuales se deben escribir todas las instrucciones de nuestro programa.

Vamos a definir tres variables de tipo entero para nuestro primer programa, esto se hace mediante la palabra reservada **var**, que se escriba antes de la declaración **begin**. Por otra parte para definir variables se usa **dos puntos** y el nombre del tipo de variable. En la siguiente imagen podemos apreciar el resultado de la definición de las variables



```
10 { you can add units after this };
. var
.   a,b,c: integer;
```

Figura 2.23: Definiendo variables de tipo entero (integer)

luego vienen las instrucciones para generar el primer programa

```
. var
.   a,b,c: integer;
. begin
.   a := 5;
15  b := 6;
.   c := a + b;
.
.   writeln('La suma de los números es = ', c);
19  readln;|
20 end.
21
```

Figura 2.24: Código completo del programa

2.6 ¿Cómo guardar un proyecto correctamente?

Para asegurarnos de que el programa funcionará correctamente tenemos que seguir algunas reglas a la hora de guardar nuestro programa.

1. No podemos usar espacios ni caracteres especiales, o letras con tildes y acentos.
2. Usar la nomenclatura **Camel Case** para el nombramiento de programas y proyectos. Esta nomenclatura que asemeja las jorobas del camello tiene esta forma por ejemplo: EsteEsMiPrimerPrograma. Note como cada palabra empieza con una letra mayúscula y el conjunto completo pareciera que tuviera varias jorobas de camello o **Camel Case**.
3. También debemos asegurarnos que la ruta donde voy a guardar mis proyectos y programas este lo mas cerca de la raíz y que esta ruta no contenga espacios.

Para guardar el primer programa que hemos creado vamos a pulsar las teclas Ctrl+S, así se abrirá una ventana preguntando donde queremos guardar nuestro proyecto o programa y que nombre le daremos. En la siguiente imagen se muestra como realizar lo forma correcta

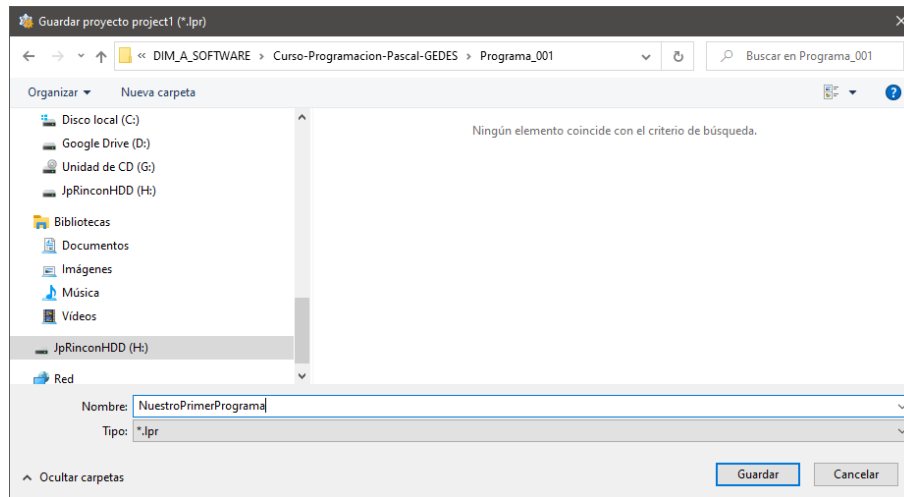


Figura 2.25: Forma correcta de guardar nuestros programas y/o proyectos

presionamos el botón [Guardar] y este creará varios archivos en el directorio donde lo hemos guardado

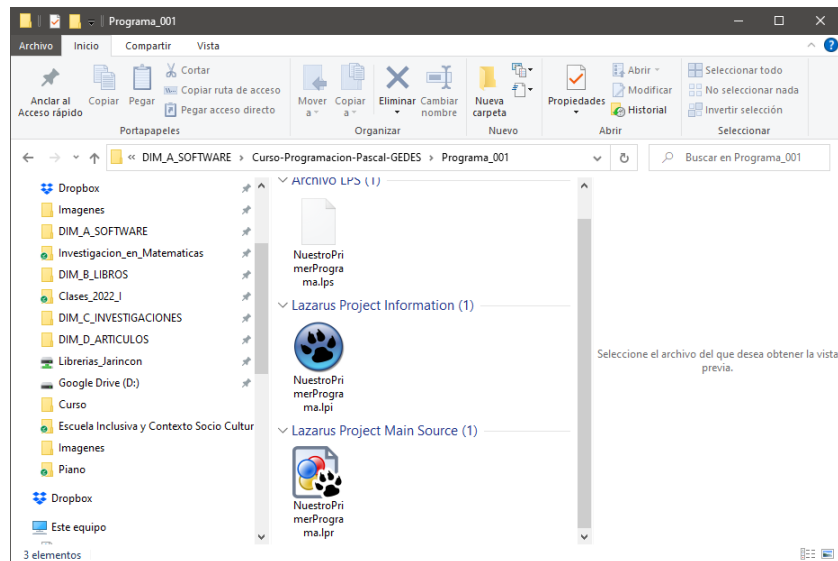


Figura 2.26: Diferentes archivos generados por el IDE al guardar

2.7 Ejecutando el programa

Para ejecutar el programa presionamos la tecla F9, que nos mostrará el siguiente resultado

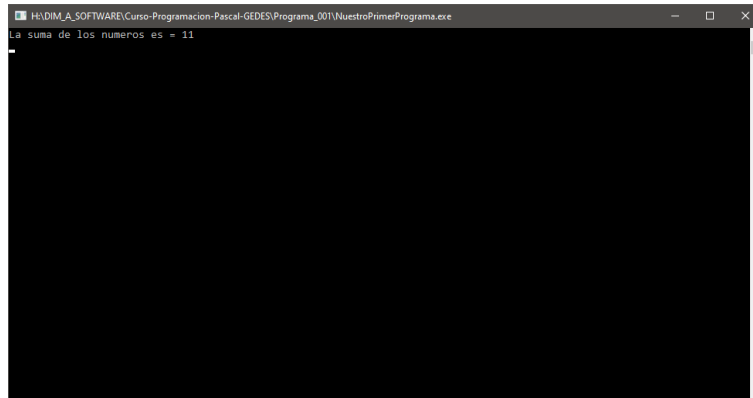


Figura 2.27: Programa en Ejecución

2.7.1 ¿En que consiste nuestro programa?

En la siguiente imagen se muestra de nuevo el código

```
. var
.   a,b,c: integer;
. begin
.   a := 5;
15  b := 6;
.   c := a + b;
.
.   writeln('La suma de los números es = ', c);
19  readln;
20 end.
21
```

Figura 2.28: Código completo del programa

en primer lugar se deben definir las variables a usar, **a**, **b** y **c** de tipo entero (**integer**).

Seguidamente se asignan las variables con sus respectivos valores usando el símbolo **:=**

Luego se usa la función **WriteLn** que recibe dos parámetros un texto que se escribe en comillas y la variable **c** que contiene el resultado de la suma.

Finalmente se usa el comando **ReadLn** para indicarle al programa que se quede en espera hasta que el usuario presione una tecla.

2.8

Capítulo 3

Variables

Capítulo 4

Estructuras de Programación

Capítulo 5

Procedimientos y Funciones

Capítulo 6

.. Programación Orientada a Objetos

Capítulo 7

Resolución de Problemas de Matemáticas