

Sistemas Concurrentes y distribuidos



Universidad
Zaragoza

Practica 1

Pablo Villa
Alvaro Perez

Índice

Portada	-----	1
Índice	-----	2
Ejercicio 1	-----	3
Ejercicio 2	-----	4
Ejercicio 3	-----	6
Ejercicio 4	-----	8
Ejercicio 5	-----	10

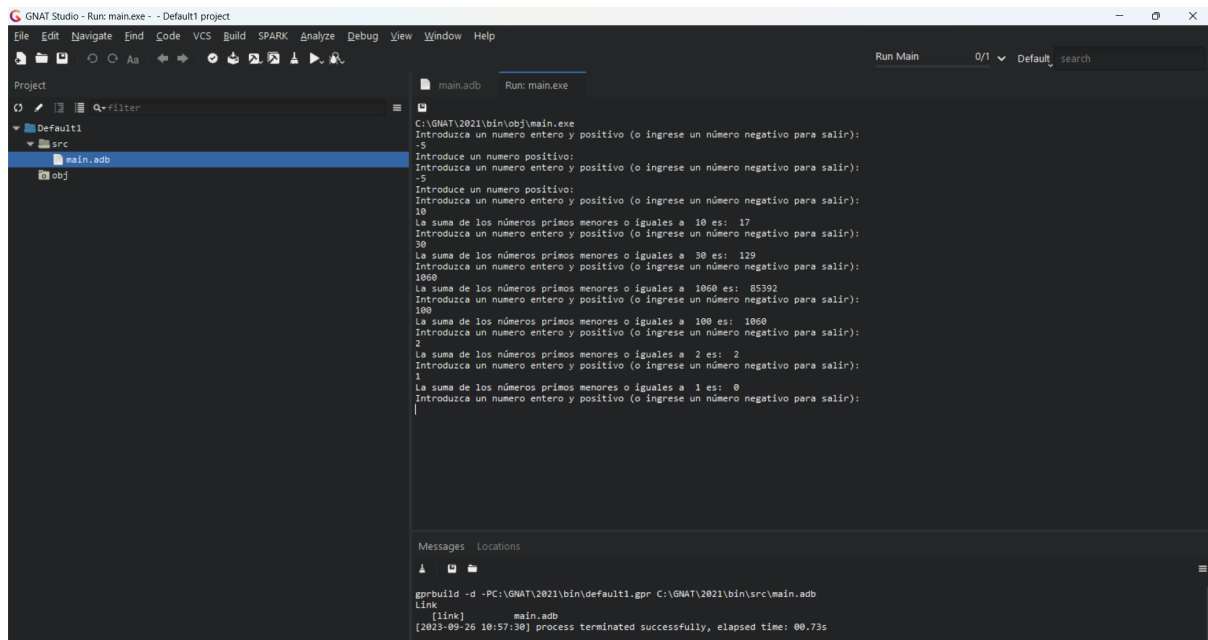
Ejercicio 1

Implementar en Ada un programa que solicite al usuario un número entero positivo y luego calcule e imprima la suma de todos los números primos menores o iguales a ese número. Deberás declarar una función auxiliar que determine si un número es primo o no

Para este ejercicio lo primero fue pedir al usuario que introduzca un número positivo si el número ingresado no lo cumple se devuelve el número y se pide que se introduzca otro.

Además creamos una función auxiliar la que devuelve un booleano si el número es primo significa que este es solo divisible por el 1 y sí mismo es decir que si hacemos el modulo del numero con todos sus anteriores hasta el dos y no es divisible por ninguno de esos devolverá que el número es primo.

en el main recorremos con un for desde 2 a el número dentro de este hacemos llamadas a la función si se prueba que el número es primo se suma , cuando finalice el for mostrará en pantalla la suma total.



```
GNAT Studio - Run: main.exe - - Default1 project
File Edit Navigate Find Code VCS Build SPARK Analyze Debug View Window Help

Project
  Default1
    src
      main.adb
    obj

Run: main.exe
C:\GNAT\2021\bin\obj\main.exe
Introduce un numero entero y positivo (o ingrese un número negativo para salir):
-5
Introduce un numero positivo:
Introduce un numero entero y positivo (o ingrese un número negativo para salir):
-5
Introduce un numero positivo:
Introduce un numero entero y positivo (o ingrese un número negativo para salir):
10
La suma de los números primos menores o iguales a 10 es: 17
Introduce un numero entero y positivo (o ingrese un número negativo para salir):
30
La suma de los números primos menores o iguales a 30 es: 129
Introduce un numero entero y positivo (o ingrese un número negativo para salir):
1000
La suma de los números primos menores o iguales a 1000 es: 85392
Introduce un numero entero y positivo (o ingrese un número negativo para salir):
100
La suma de los números primos menores o iguales a 100 es: 1060
Introduce un numero entero y positivo (o ingrese un número negativo para salir):
2
La suma de los números primos menores o iguales a 2 es: 2
Introduce un numero entero y positivo (o ingrese un número negativo para salir):
1
La suma de los números primos menores o iguales a 1 es: 0
Introduce un numero entero y positivo (o ingrese un número negativo para salir):

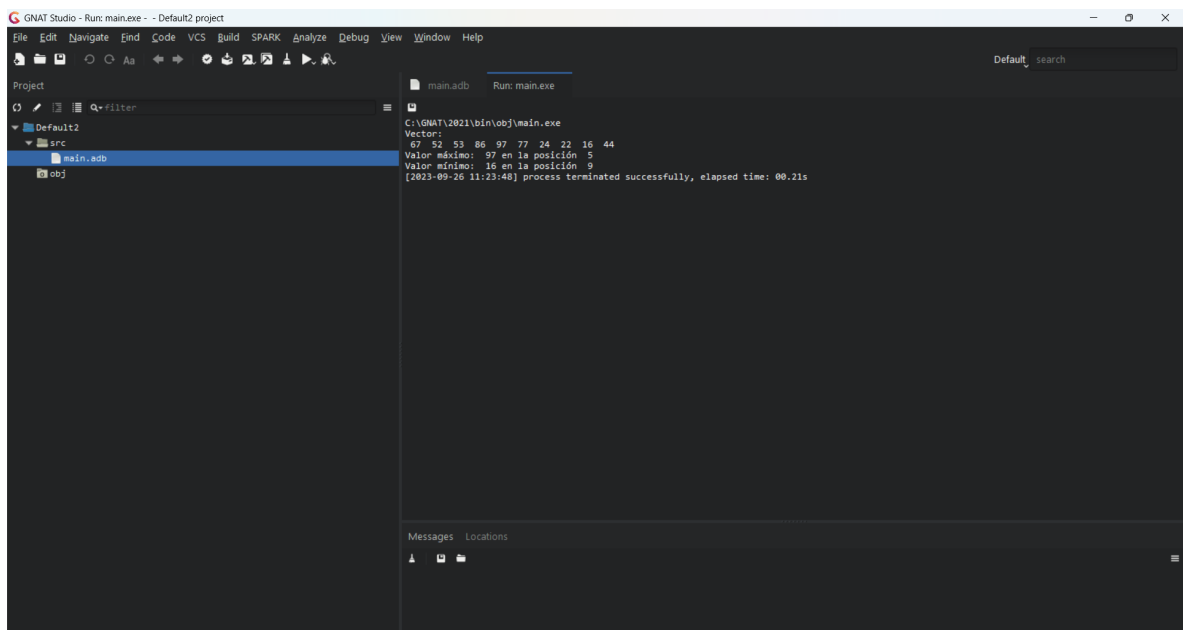
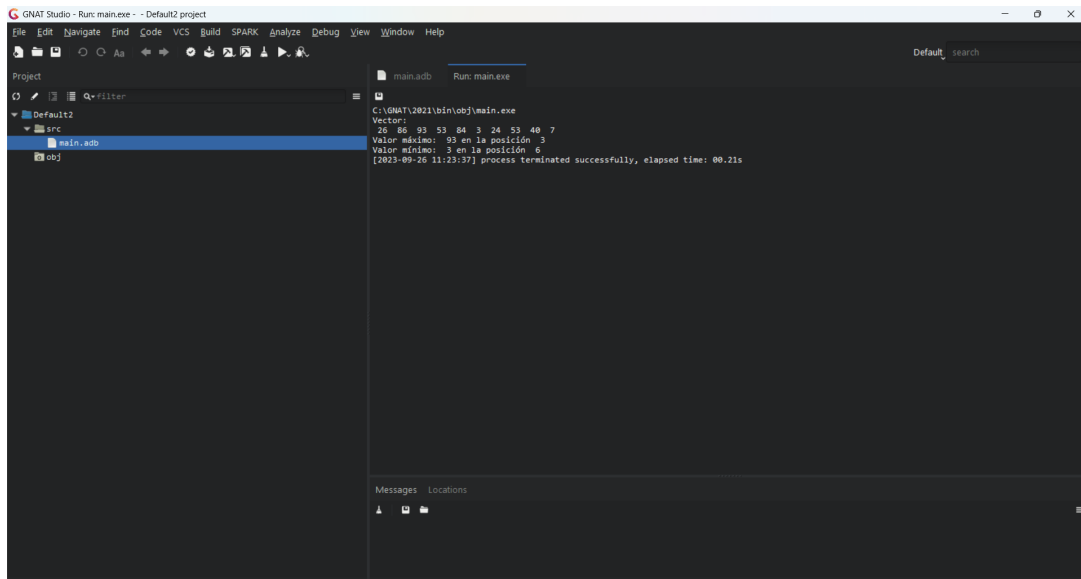
Messages Locations
gprbuild -d -PC:\GNAT\2021\bin\default1.gpr C:\GNAT\2021\bin\src\main.adb
Link
[[link]] main.adb
[2023-09-26 10:57:30] process terminated successfully, elapsed time: 00.73s
```

Ejercicio 2

Implementar en Ada un programa que declare inicialmente un vector de enteros de tamaño 10 y lo inicializa con valores aleatorios. Luego, el programa deberá encontrar el valor máximo y mínimo en el vector e imprimir ambos valores junto con sus posiciones en el vector

1. Declaramos el vector de enteros con 10 elementos.
2. Para inicializar con valores aleatorios utilizamos la librería `Ada.Numerics.Float Random` y creamos una función la cual generará ese número aleatorio que buscamos.
3. En el main recorremos el rango del vector y en cada iteración llamamos a la función para darle a cada índice del vector un número aleatorio.
4. Volvemos a recorrer el vector esta vez buscando el número más grande y más pequeño de nuestro vector cuando hayamos comparado con todo nuestro vector.
5. Enseñaremos el valor más grande y más pequeño del vector y la posición en la que se encuentran.

La mayor dificultad que hemos encontrado a la hora de realizar este ejercicio ha sido generar los números aleatorios, ya que nos estaban dando problemas pero tras utilizar la librería más adecuada nos ha acabado funcionando.

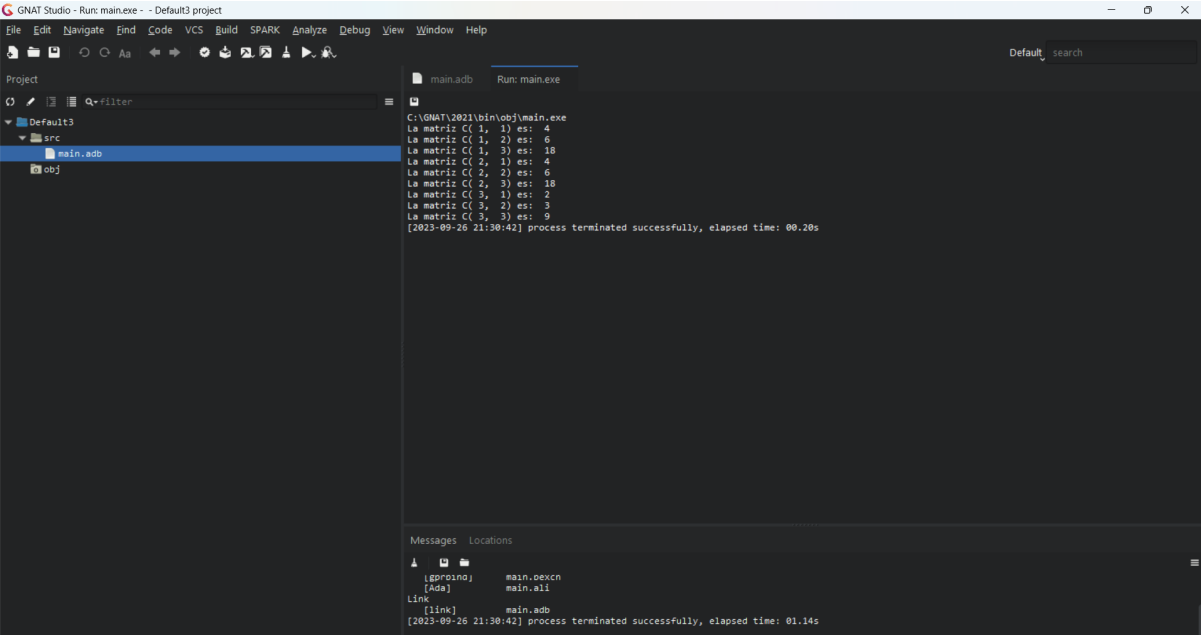


Ejercicio 3

Implementar en Ada un programa que declare dos matrices bidimensionales A y B de dimensiones 3x3. Inicializa ambas matrices con valores enteros. Luego, el programa debe realizar la multiplicación de matrices A y B y almacenar el resultado en una tercera matriz C. Finalmente, muestra la matriz C resultante por pantalla

1. Declarar 3 arrays bidimensionales A,B,C
2. 2 procedimientos de llenado de las matrices
3. Llenar la matriz A recorriendo la matriz con dos bucles fors anidados si los índices son iguales es decir la diagonal principal dar valor = 1 , si son distintas valor = 0
4. Llenar matriz B manualmente con números
5. La matriz C es el resultado de la multiplicación AxB con tres bucles anidados recorriendo las filas y columnas de las matrices y haciendo las multiplicaciones
6. Mostramos los valores de la matriz C con dos bucles anidados

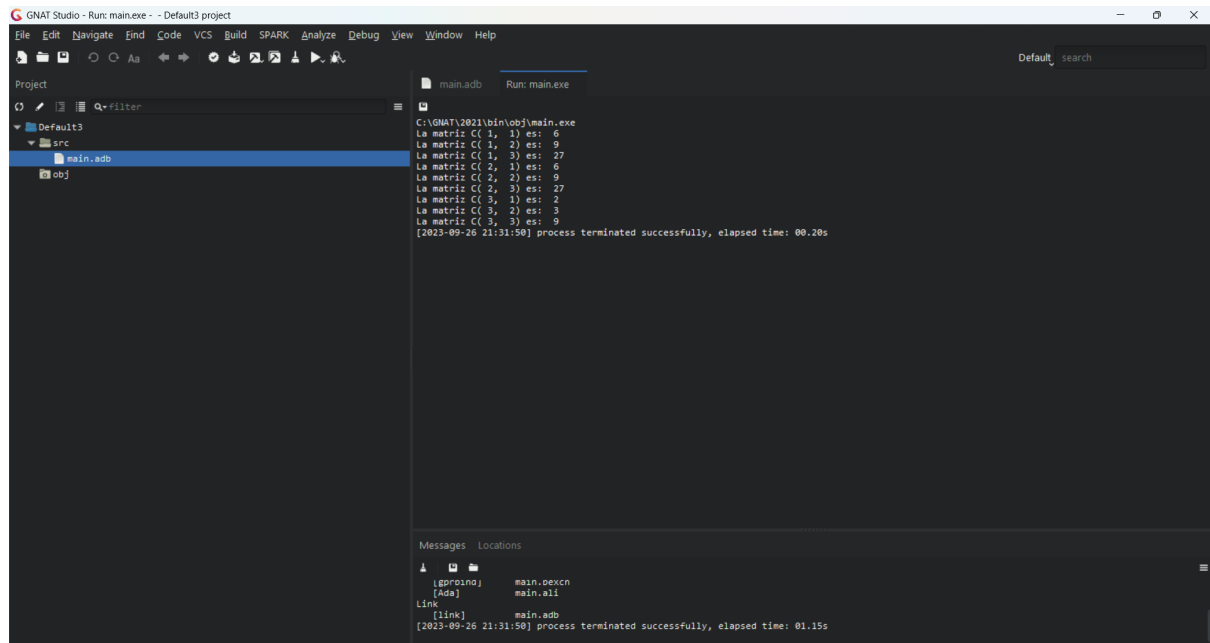
ejemplo con matriz a diagonal principal 1 y demas 2



```
GNAT Studio - Run: main.exe - - Default3 project
File Edit Navigate Find Code VCS Build SPARK Analyze Debug View Window Help
Project
  Default3
    src
      main.adb
    obj
main.adb Run: main.exe
C:\GNAT\2021\bin\obj\main.exe
La matriz C (1, 1) es: 4
La matriz C (1, 2) es: 6
La matriz C (1, 3) es: 18
La matriz C (2, 1) es: 4
La matriz C (2, 2) es: 6
La matriz C (2, 3) es: 18
La matriz C (3, 1) es: 2
La matriz C (3, 2) es: 3
La matriz C (3, 3) es: 9
[2023-09-26 21:30:42] process terminated successfully, elapsed time: 00.28s

Messages Locations
[pproindaj] main.pexcn
[Ada] main.ali
Link
[link] main.adb
[2023-09-26 21:30:42] process terminated successfully, elapsed time: 01.14s
```

ejemplo diagonal principal 1 resto 3



The screenshot shows the GNAT Studio interface. On the left, the 'Project' pane displays a tree structure for 'Default3' with subfolders 'src' and 'obj', and a file 'main.adb'. The main editor area shows the 'Run: main.exe' console output. The output displays a 3x3 matrix with values 6, 9, 27 on the diagonal and 0 elsewhere, followed by a successful termination message. The bottom pane shows the 'Messages' section with a table of messages and their locations.

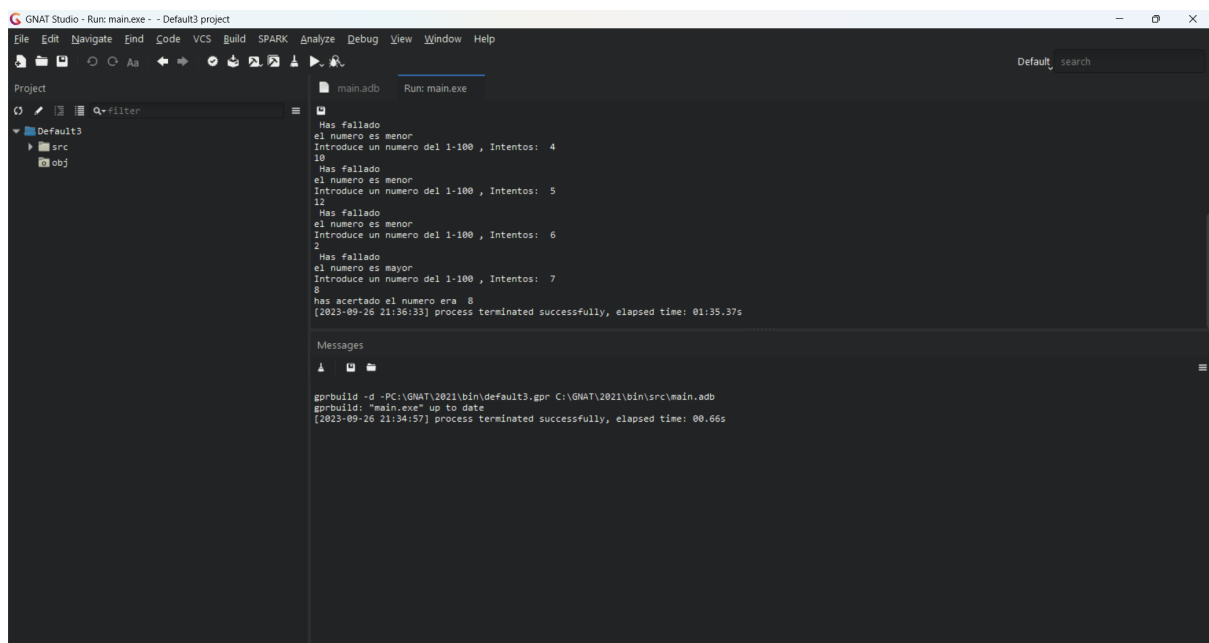
```
C:\GNAT\2021\bin\obj\main.exe
La matriz C(1, 1) es: 6
La matriz C(1, 2) es: 9
La matriz C(1, 3) es: 27
La matriz C(2, 1) es: 6
La matriz C(2, 2) es: 9
La matriz C(2, 3) es: 27
La matriz C(3, 1) es: 2
La matriz C(3, 2) es: 3
La matriz C(3, 3) es: 9
[2023-09-26 21:31:50] process terminated successfully, elapsed time: 00.20s
```

Messages	Locations
[gprcno] main.pexcn	
[Ada] main.ali	
Link	main.adb
[2023-09-26 21:31:50] process terminated successfully, elapsed time: 01.15s	

Ejercicio 4

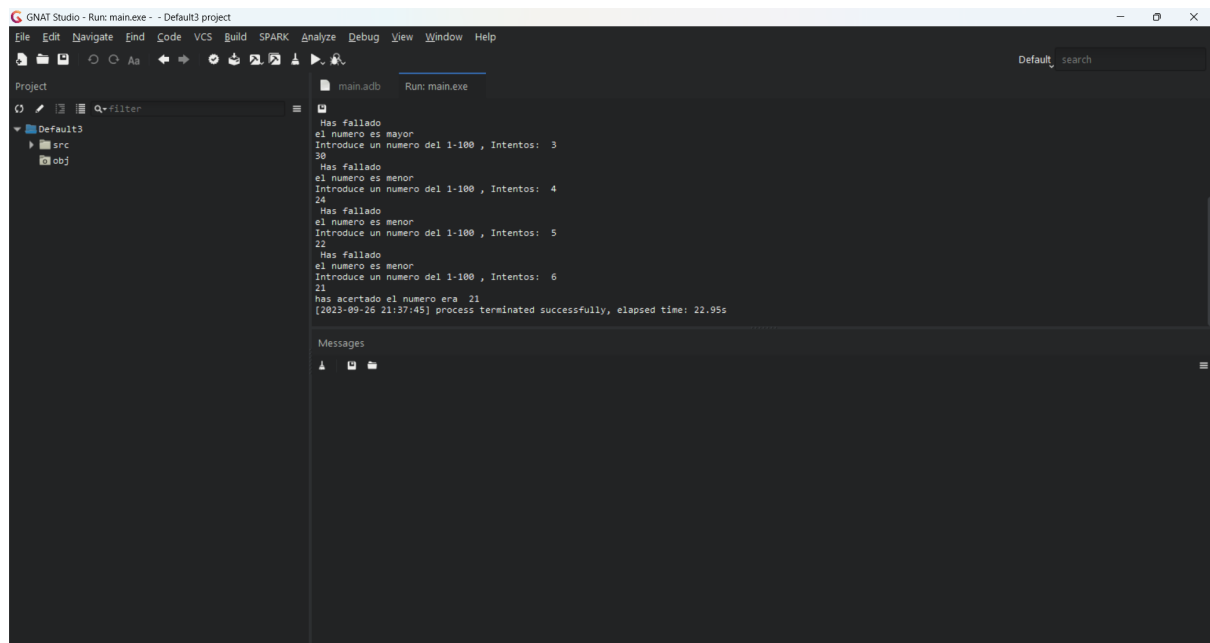
Implementar en Ada un programa que simule un juego de adivinar un número secreto. El programa generará un número secreto aleatorio entre 1 y 100, y luego permitirá al usuario ingresar números hasta que adivine el número secreto. El programa debe proporcionar pistas de si el número ingresado es mayor o menor que el número secreto. Lleva un registro del número de intentos y muestra el número de intentos cuando el usuario adivina el número

Para este ejercicio hemos usado la misma función de generar números aleatorios y le hemos pedido al usuario que introduzca un número positivo entre un rango si el número no es positivo o entre el intervalo que pida introducir de nuevo un número y por cada vez que escribe por pantalla se suma un intento más hasta que lo adivine



```
GNAT Studio - Run: main.exe - - Default3 project
File Edit Navigate Find Code VCS Build SPARK Analyze Debug View Window Help
Project
  Default3
    src
    obj
Run: main.exe
  Has fallado
  el numero es menor
  Introduce un numero del 1-100 , Intentos: 4
  10
  Has fallado
  el numero es menor
  Introduce un numero del 1-100 , Intentos: 5
  12
  Has fallado
  el numero es menor
  Introduce un numero del 1-100 , Intentos: 6
  2
  Has fallado
  el numero es mayor
  Introduce un numero del 1-100 , Intentos: 7
  8
  has acertado el numero era 8
  [2023-09-26 21:36:33] process terminated successfully, elapsed time: 01:35.37s

Messages
  gprbuild -d -PC:\GNAT\2021\bin\default3.gpr C:\GNAT\2021\bin\src\main.adb
  gprbuild: "main.exe" up to date
  [2023-09-26 21:34:57] process terminated successfully, elapsed time: 00.66s
```

Ejercicio 5

Implementar en Ada un programa que gestione una lista de contactos. Cada contacto se representa con un registro que contiene nombre, número de teléfono y dirección de correo electrónico. El programa debe permitir al usuario agregar nuevos contactos, buscar un contacto por nombre o número de teléfono, y eliminar contactos. Debe utilizar tipos de datos compuestos y estructuras de datos dinámicas para gestionar la lista de contactos.

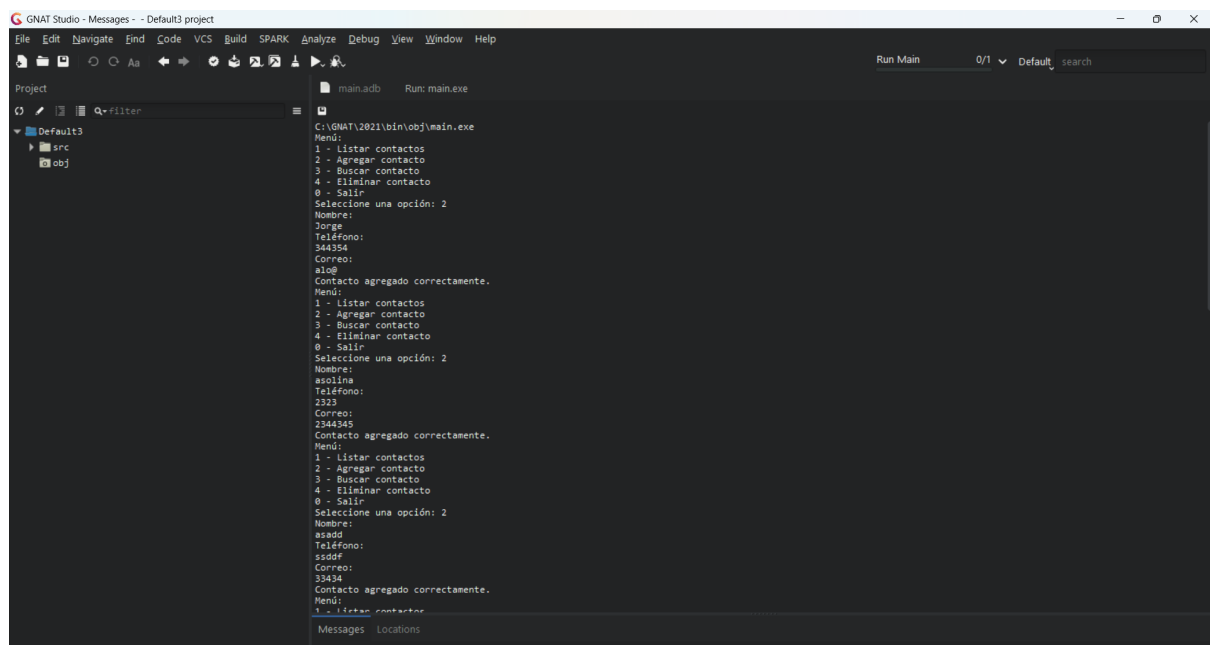
GuardarContacto: Permite al usuario agregar nuevos contactos, verificando si la lista de contactos está llena antes de agregar uno nuevo.

buscarPorNombre: Permite al usuario buscar un contacto por nombre y muestra la información si se encuentra.

BuscarPorTlf: Permite al usuario buscar un contacto por número de teléfono y muestra la información si se encuentra.

EliminarContacto: Permite al usuario eliminar un contacto proporcionando su nombre o número de teléfono.

El programa se ejecuta en un bucle de case donde el usuario puede seleccionar diferentes opciones, como listar, agregar, buscar, eliminar contactos o salir del programa. Si el usuario ingresa una opción no válida, el programa mostrará un mensaje de error.



```
GNAT Studio - Messages - --Default3 project
File Edit Navigate Find Code VCS Build SPARK Analyze Debug View Window Help
Project: main.adb Run: main.exe
C:\GNAT\2021\bin\obj\main.exe
Menu:
1 - Listar contactos
2 - Agregar contacto
3 - Buscar contacto
4 - Eliminar contacto
0 - Salir
Seleccione una opción: 2
Nombre:
Jorge
Teléfono:
344354
Correo:
aloe@
Contacto agregado correctamente.
Menu:
1 - Listar contactos
2 - Agregar contacto
3 - Buscar contacto
4 - Eliminar contacto
0 - Salir
Seleccione una opción: 2
Nombre:
asolina
Teléfono:
2323
Correo:
2344345
Contacto agregado correctamente.
Menu:
1 - Listar contactos
2 - Agregar contacto
3 - Buscar contacto
4 - Eliminar contacto
0 - Salir
Seleccione una opción: 2
Nombre:
asadd
Teléfono:
ssddf
Correo:
33434
Contacto agregado correctamente.
Menu:
1 - Listar contactos
Messages Locations
```

GNAT Studio - Messages - - Default3 project

File Edit Navigate Find Code VCS Build SPARK Analyze Debug View Window Help

Run Main 0/1 Default search

Project

- Default3
 - src
 - obj

main.adb Run: main.exe

```
o - Jeaxr
Seleccione una opción: 4
Introduce el nombre o telefono del usuario que quiere ser eliminado:
Jorge
contacto eliminado
Menu:
1 - Listar contactos
2 - Agregar contacto
3 - Buscar contacto
4 - Eliminar contacto
0 - Salir
Seleccione una opción: 2
Nombre:
224224
Telefono:
233434
Correo:
232423
Contacto agregado correctamente.
Menu:
1 - Listar contactos
2 - Agregar contacto
3 - Buscar contacto
4 - Eliminar contacto
0 - Salir
Seleccione una opción: 1
Contacto 1:
Nombre: asolina
Telefono: 2323
Correo: 2344345
Contacto 2:
Nombre: asadd
Telefono: ssddf
Correo: 33434
Contacto 3:
Nombre: 224224
Telefono: 233434
Correo: 232423
Menu:
1 - Listar contactos
2 - Agregar contacto
3 - Buscar contacto
4 - Eliminar contacto
0 - Salir
Seleccione una opción:
```

Messages Locations