

Ejercicio 1: Sumatoria de números primos en un rango

Escribe un programa que solicite dos números y calcule la sumatoria de los números primos que existen entre esos dos valores. Utiliza un bucle for o while para recorrer los números en el rango y verifica si son primos.

Ejercicio 2: Números de Fibonacci hasta N términos

Implementa un programa que genere la secuencia de Fibonacci hasta un número n de términos ingresado por el usuario. Utiliza un bucle while o for para ir generando los números de la secuencia.

Ejercicio 3: Factorial de números grandes

Escribe un programa que calcule el factorial de un número grande (por ejemplo, 100) utilizando estructuras repetitivas y el tipo de datos BigInteger para manejar grandes números.

Ejercicio 4: Inversión de un número

Crea un programa que invierta los dígitos de un número entero ingresado por el usuario, utilizando un bucle while para extraer y reordenar los dígitos.

Ejercicio 5: Suma de matrices NxN

Escribe un programa que solicite dos matrices de tamaño N x N (donde N es proporcionado por el usuario) y luego realice la suma de las dos matrices utilizando bucles anidados for.

Ejercicio 6: Número perfecto

Implementa un programa que encuentre y muestre todos los números perfectos entre 1 y 10,000. Un número perfecto es aquel que es igual a la suma de sus divisores propios. Usa un bucle para iterar y otro para encontrar los divisores de cada número.

Ejercicio 7: Matriz de espiral

Crea un programa que imprima una matriz cuadrada de tamaño nxn en forma de espiral. Utiliza bucles anidados para recorrer las posiciones de la matriz en el orden adecuado.

Ejercicio 8: Verificación de un número Armstrong  
Escribe un programa que verifique si un número de n dígitos ingresado por el usuario es un número de Armstrong (o narcisista). Utiliza un bucle for para separar y elevar cada dígito a la potencia correspondiente.

Ejercicio 9: Cálculo de potencias usando multiplicación repetida

Crea un programa que calcule la potencia de un número usando multiplicación repetida, es decir, sin utilizar la función Math.pow(). El programa debe solicitar una base y un exponente, y luego calcular la potencia utilizando un bucle while o for.

Terminal Help ← → s4\_2

ejerc1.dart X

ejerc1.dart > ...

1 bool esPrimo(int numero) {  
2   if (numero < 2) return false;  
3   for (int i = 2; i <= numero ~/ 2; i++) {  
4     if (numero % i == 0) return false;  
5   }  
6   return true;  
7 }  
8  
9 Run | Debug  
10 void main() {  
11   int inicio = 10; // rango que uno desea  
12   int fin = 50;    // rango que uno desea  
13  
14   int sumatoriaPrimos = 0;  
15  
16   for (int i = inicio; i <= fin; i++) {  
17     if (esPrimo(i)) {  
18       sumatoriaPrimos += i;  
19     }  
20   }  
21   print("La sumatoria de los números primos entre \$inicio y \$fin es: \$sumatoriaPrimos")  
22 }  
23

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\pelon\01\s4\_2> dart ejerc1.dart  
La sumatoria de los números primos entre 10 y 50 es: 311  
PS C:\pelon\01\s4\_2>

```
bool esPrimo(int numero) {  
  if (numero < 2) return false;  
  for (int i = 2; i <= numero ~/ 2; i++) {  
    if (numero % i == 0) return false;  
  }  
  return true;  
}  
  
void main() {  
  int inicio = 10; // rango que uno desea  
  int fin = 50;    // rango que uno desea  
  
  int sumatoriaPrimos = 0;  
  
  for (int i = inicio; i <= fin; i++) {  
    if (esPrimo(i)) {  
      sumatoriaPrimos += i;  
    }  
  }  
  
  print("La sumatoria de los números primos entre $inicio y $fin es:  
    $sumatoriaPrimos");  
}
```

Terminal Help ← → s4\_2

ejerc1.dart

ejerc2.dart ×

ejerc2.dart > main

Run | Debug

1 void main() {  
2   int n = 10; // obtener más o menos términos de la secuencia  
3  
4   int a = 0;  
5   int b = 1;  
6  
7   print("Secuencia de Fibonacci de \$n términos:");  
8  
9   for (int i = 0; i < n; i++) {  
10     print(a);  
11     int temp = a + b;  
12     a = b;  
13     b = temp;  
14   }  
15 }  
16

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Secuencia de Fibonacci de 10 términos:  
0  
1  
1  
2  
3  
5  
8  
13  
21  
34  
PS C:\pelon\01\s4\_2>

```
void main() {  
  int n = 10; // obtener más o menos  
               términos de la secuencia  
  
  int a = 0;  
  int b = 1;  
  
  print("Secuencia de Fibonacci de $n  
        términos:");  
  
  for (int i = 0; i < n; i++) {  
    print(a);  
    int temp = a + b;  
    a = b;  
    b = temp;  
  }  
}
```

Go Run Terminal Help

← → s4\_2

ejerc1.dart

ejerc2.dart

ejerc3.dart ✕

ejerc4.dart

ejerc5.dart

ejerc6.dart

ejerc7.dart

ejerc8.dart

ejerc9.dart

ejerc3.dart > main

Run | Debug

1 void main() {

2   int numero = 10; // Cambia este valor si deseas calcular el factorial de otro número.

3   BigInt resultado = BigInt.from(1);

4

5   for (int i = 1; i <= numero; i++) {

6     resultado \*= BigInt.from(i);

7   }

8

9   print("El factorial de \$numero es: \$resultado");

10 }

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

PS C:\pelon\01\s4\_2> dart ejerc3.dart

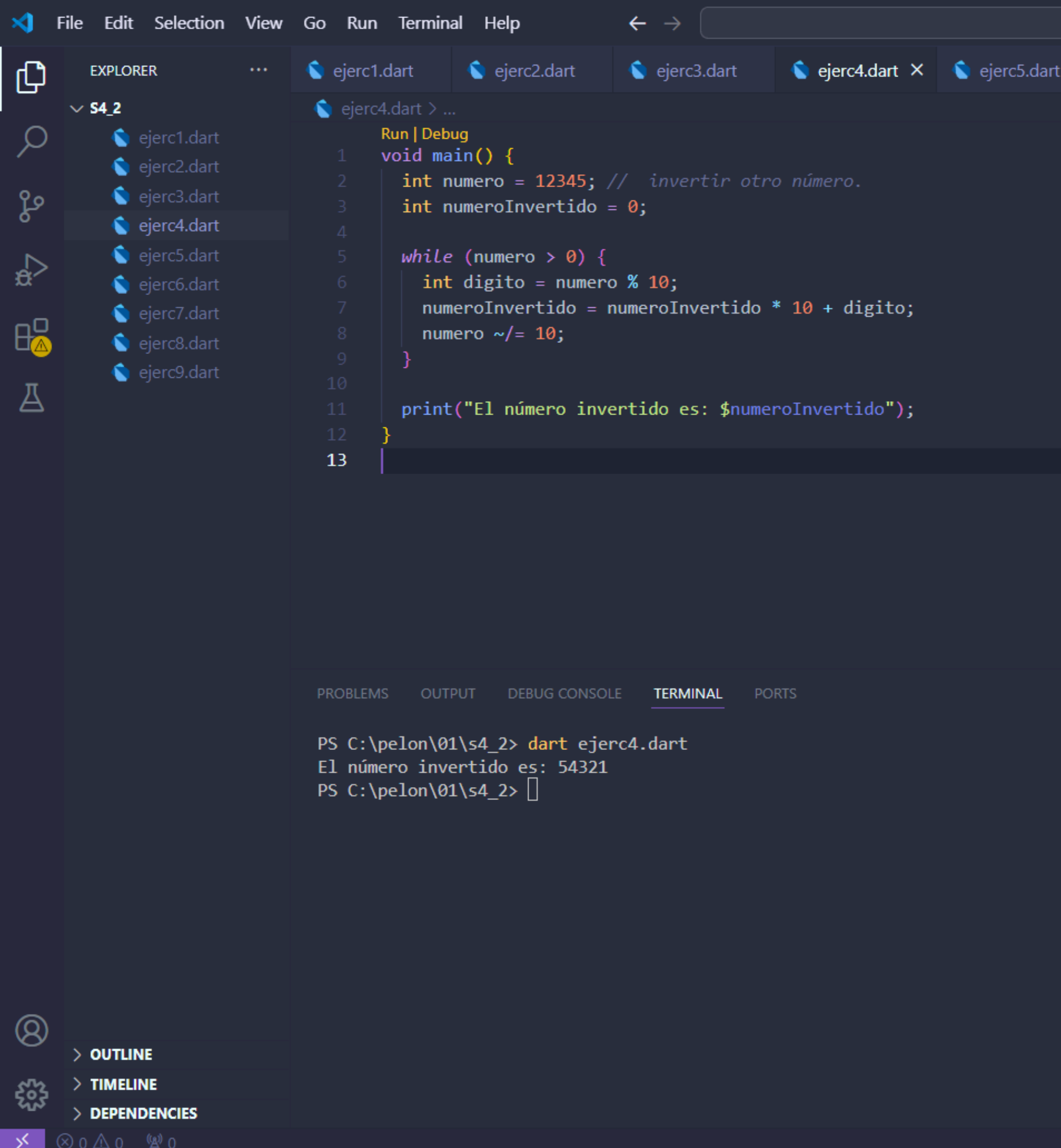
El factorial de 100 es: 9332621544394415268169923885626670049071596826438162146859296389521759999322991560894146397615651828625369792082722375825118521091686400000000000000000000

PS C:\pelon\01\s4\_2> dart ejerc3.dart

El factorial de 10 es: 3628800

PS C:\pelon\01\s4\_2>

```
void main() {  
  int numero = 10; // Cambia  
este valor si deseas calcular el  
factorial de otro número.  
  BigInt resultado =  
    BigInt.from(1);  
  
  for (int i = 1; i <= numero; i++)  
  {  
    resultado *= BigInt.from(i);  
  }  
  
  print("El factorial de $numero  
es: $resultado");  
}
```



```
void main() {  
int numero = 12345; // invertir otro número.  
int numeroInvertido = 0;  
  
while (numero > 0) {  
int digito = numero % 10;  
numeroInvertido = numeroInvertido * 10 + digito;  
numero ~/= 10;  
}  
  
print("El número invertido es: $numeroInvertido");  
}
```

```
Go Run Terminal Help
ejerc1.dart ejerc2.dart ejerc3.dart ejerc4.dart ejerc5.dart x ejerc6.dart
ejerc5.dart > main
Run | Debug
1 void main() {
2   int N = 3; // matrices de diferente tamaño.
3
4   // Matrices de ejemplo (puedes cambiarlas o generarlas de forma dinámica)
5   List<List<int>> matriz1 = [
6     [1, 2, 3],
7     [4, 5, 6],
8     [7, 8, 9]
9   ];
10  List<List<int>> matriz2 = [
11    [9, 8, 7],
12    [6, 5, 4],
13    [3, 2, 1]
14  ];
15
16  // Matriz resultante de la suma
17  List<List<int>> sumaMatriz = List.generate(N, (_) => List.filled(N, 0));
18
19  for (int i = 0; i < N; i++) {
20    for (int j = 0; j < N; j++) {
21      sumaMatriz[i][j] = matriz1[i][j] + matriz2[i][j];
22    }
23  }
24
25  print("La suma de las matrices es:");
26  for (int i = 0; i < N; i++) {
27    print(sumaMatriz[i]);
28  }
29 }
30
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
La suma de las matrices es:
[10, 10, 10]
[10, 10, 10]
[10, 10, 10]
PS C:\pelon\01\s4_2>
```

```
void main() {
  int N = 3; // matrices de diferente tamaño.

  // Matrices de ejemplo (puedes cambiarlas o generarlas de forma dinámica)
  List<List<int>> matriz1 = [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
  ];
  List<List<int>> matriz2 = [
    [9, 8, 7],
    [6, 5, 4],
    [3, 2, 1]
  ];

  // Matriz resultante de la suma
  List<List<int>> sumaMatriz = List.generate(N, (_) => List.filled(N, 0));

  for (int i = 0; i < N; i++) {
    for (int j = 0; j < N; j++) {
      sumaMatriz[i][j] = matriz1[i][j] + matriz2[i][j];
    }
  }

  print("La suma de las matrices es:");
  for (int i = 0; i < N; i++) {
    print(sumaMatriz[i]);
  }
}
```

```
Selection View Go Run Terminal Help
ejerc1.dart
ejerc2.dart
ejerc3.dart
ejerc4.dart
ejerc6.dart > ...
Run | Debug
1 void main() {
2   print("Números perfectos entre 1 y 10,000:");
3
4   for (int num = 1; num <= 10000; num++) {
5     int sumaDivisores = 0;
6
7     for (int i = 1; i <= num ~/ 2; i++) {
8       if (num % i == 0) {
9         sumaDivisores += i;
10      }
11    }
12
13    if (sumaDivisores == num) {
14      print(num);
15    }
16  }
17 }
18

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\pelon\01\s4_2> dart ejerc6.dart
Números perfectos entre 1 y 10,000:
6
28
496
8128
PS C:\pelon\01\s4_2> 
```

```
void main() {
print("Números perfectos entre 1 y 10,000:");

for (int num = 1; num <= 10000; num++) {
    int sumaDivisores = 0;

    for (int i = 1; i <= num ~/ 2; i++) {
        if (num % i == 0) {
            sumaDivisores += i;
        }
    }

    if (sumaDivisores == num) {
        print(num);
    }
}
}
```



File Edit Selection View Go Run Terminal Help

EXPLORER

ejerc1.dart ejerc2.dart ejerc3.dart ejerc4.dart ejerc5.dart ejerc6.dart ejerc7.dart ejerc8.dart ejerc9.dart

S4\_2

ejerc7.dart > main

```
Run | Debug
void main() {
  1  int n = 5; // matrices de diferente tamaño.
  2  List<List<int>> matriz = List.generate(n, (_) => List.filled(n, 0));
  3
  4
  5  int valor = 1;
  6  int filaInicio = 0, filaFin = n - 1;
  7  int colInicio = 0, colFin = n - 1;
  8
  9  while (filaInicio <= filaFin && colInicio <= colFin) {
 10    // Llenar la fila superior
 11    for (int i = colInicio; i <= colFin; i++) {
 12      matriz[filaInicio][i] = valor++;
 13    }
 14    filaInicio++;
 15
 16    // Llenar la columna derecha
 17    for (int i = filaInicio; i <= filaFin; i++) {
 18      matriz[i][colFin] = valor++;
 19    }
 20    colFin--;
 21
 22    // Llenar la fila inferior
 23    for (int i = colFin; i >= colInicio; i--) {
 24      matriz[filaFin][i] = valor++;
 25    }
 26    filaFin--;
 27
 28    // Llenar la columna izquierda
 29    for (int i = filaFin; i >= filaInicio; i--) {
 30      matriz[i][colInicio] = valor++;
 31    }
 32    colInicio++;
 33  }
 34
 35  print("Matriz en forma de espiral:");
 36  for (var fila in matriz) {
 37    print(fila);
 38  }
 39}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Matriz en forma de espiral:  
[1, 2, 3, 4, 5]  
[16, 17, 18, 19, 6]  
[15, 24, 25, 20, 7]  
[14, 23, 22, 21, 8]  
[13, 12, 11, 10, 9]  
PS C:\pelon\01\s4\_2>

OUTLINE  
TIMELINE  
DEPENDENCIES

```
void main() {
  int n = 5; // matrices de diferente tamaño.
  List<List<int>> matriz = List.generate(n, (_) => List.filled(n, 0));

  int valor = 1;
  int filaInicio = 0, filaFin = n - 1;
  int colInicio = 0, colFin = n - 1;

  while (filaInicio <= filaFin && colInicio <= colFin) {
    // Llenar la fila superior
    for (int i = colInicio; i <= colFin; i++) {
      matriz[filaInicio][i] = valor++;
    }
    filaInicio++;

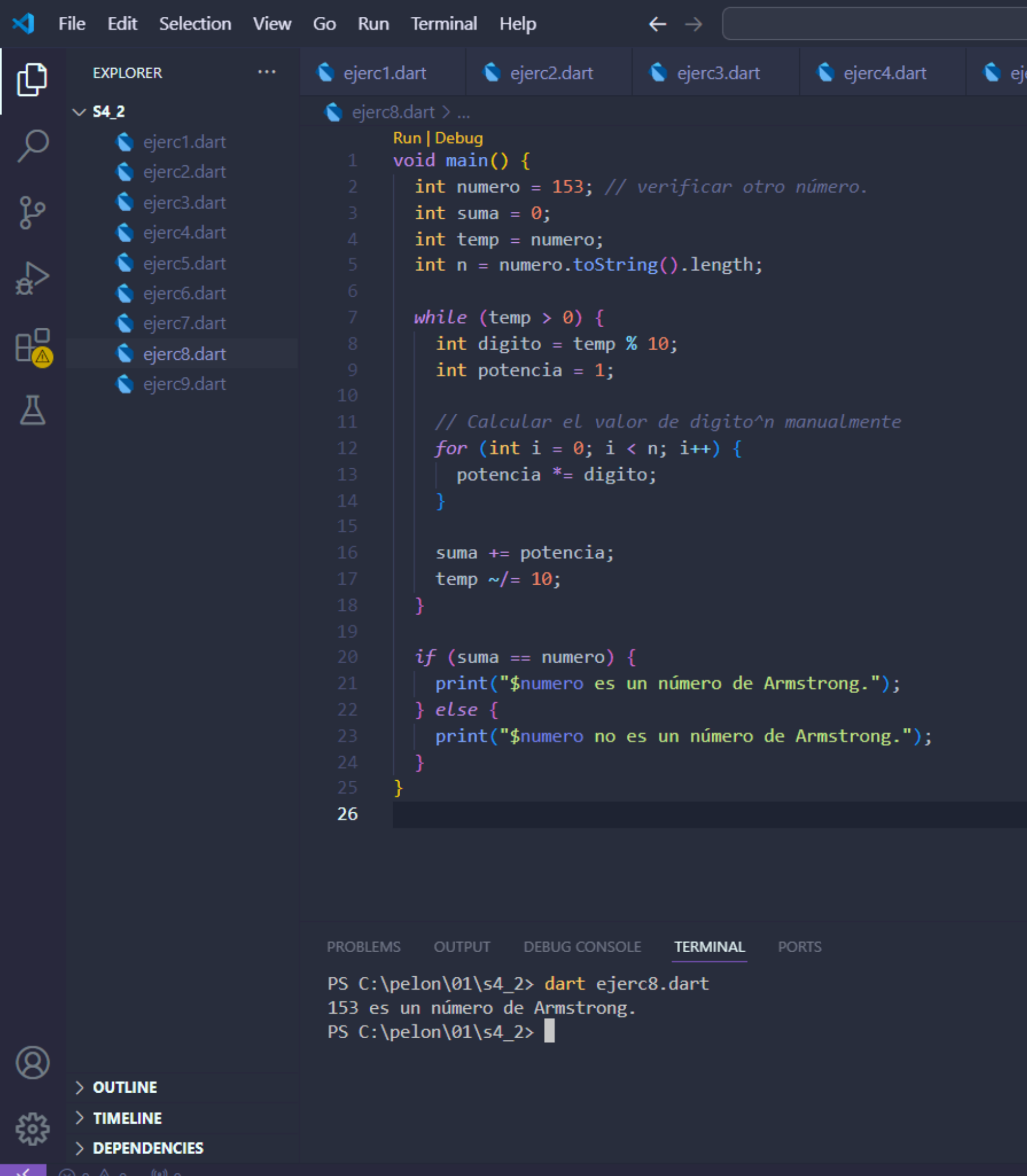
    // Llenar la columna derecha
    for (int i = filaInicio; i <= filaFin; i++) {
      matriz[i][colFin] = valor++;
    }
    colFin--;

    // Llenar la fila inferior
    for (int i = colFin; i >= colInicio; i--) {
      matriz[filaFin][i] = valor++;
    }
    filaFin--;

    // Llenar la columna izquierda
    for (int i = filaFin; i >= filaInicio; i--) {
      matriz[i][colInicio] = valor++;
    }
    colInicio++;
  }

  print("Matriz en forma de espiral:");
  for (var fila in matriz) {
    print(fila);
  }
}
```





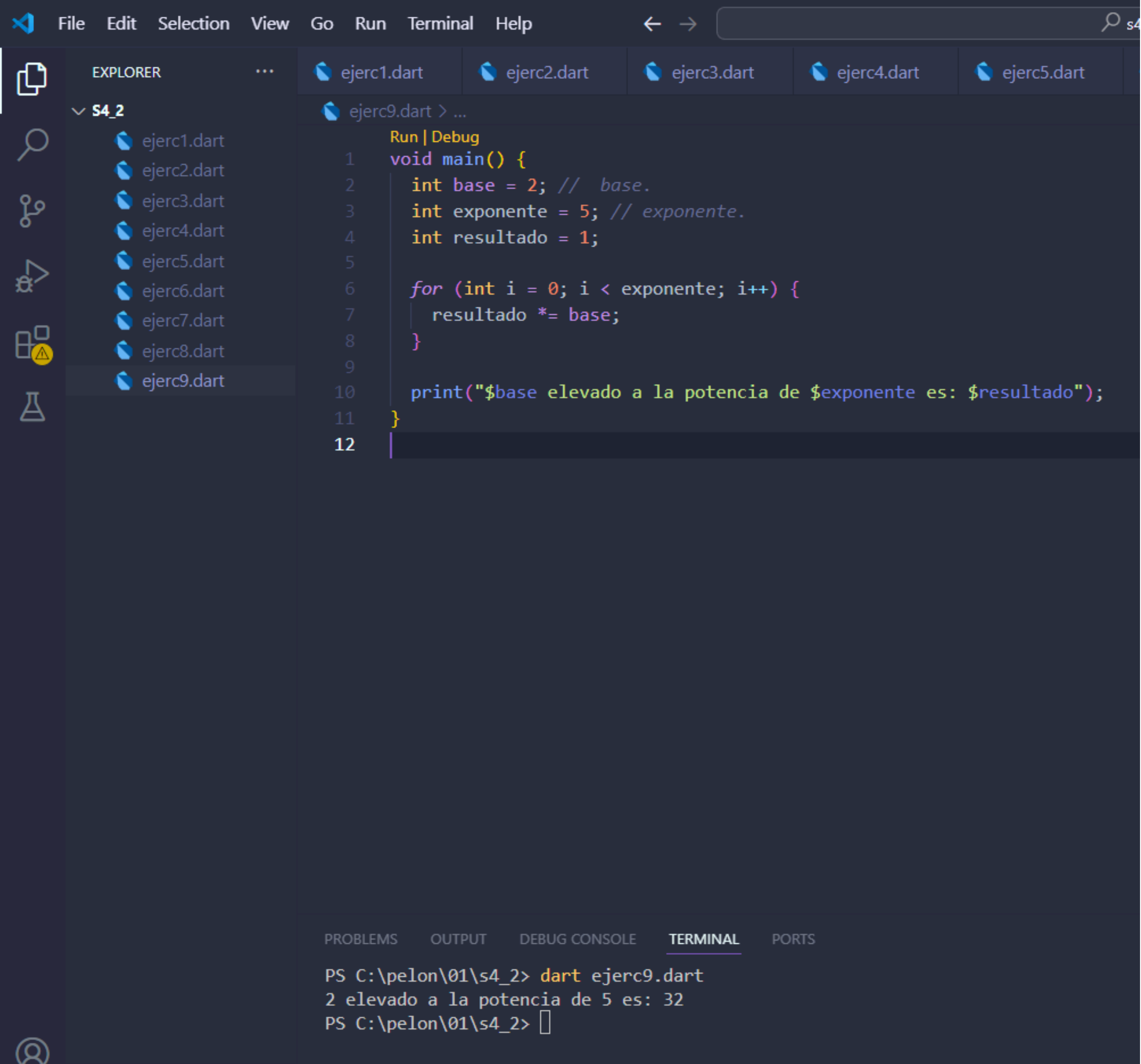
```
void main() {
  int numero = 153; // verificar otro número.
  int suma = 0;
  int temp = numero;
  int n = numero.toString().length;

  while (temp > 0) {
    int digito = temp % 10;
    int potencia = 1;

    // Calcular el valor de digito^n manualmente
    for (int i = 0; i < n; i++) {
      potencia *= digito;
    }

    suma += potencia;
    temp ~/= 10;
  }

  if (suma == numero) {
    print("$numero es un número de Armstrong.");
  } else {
    print("$numero no es un número de Armstrong.");
  }
}
```



```
void main() {  
    int base = 2; // base.  
    int exponente = 5; // exponente.  
    int resultado = 1;  
  
    for (int i = 0; i < exponente; i++) {  
        resultado *= base;  
    }  
  
    print("$base elevado a la potencia de  
        $exponente es: $resultado");  
}
```