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 digitos 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 digito 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

```dart
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;
}

Run | Debug
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")
}
```

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;
}
}

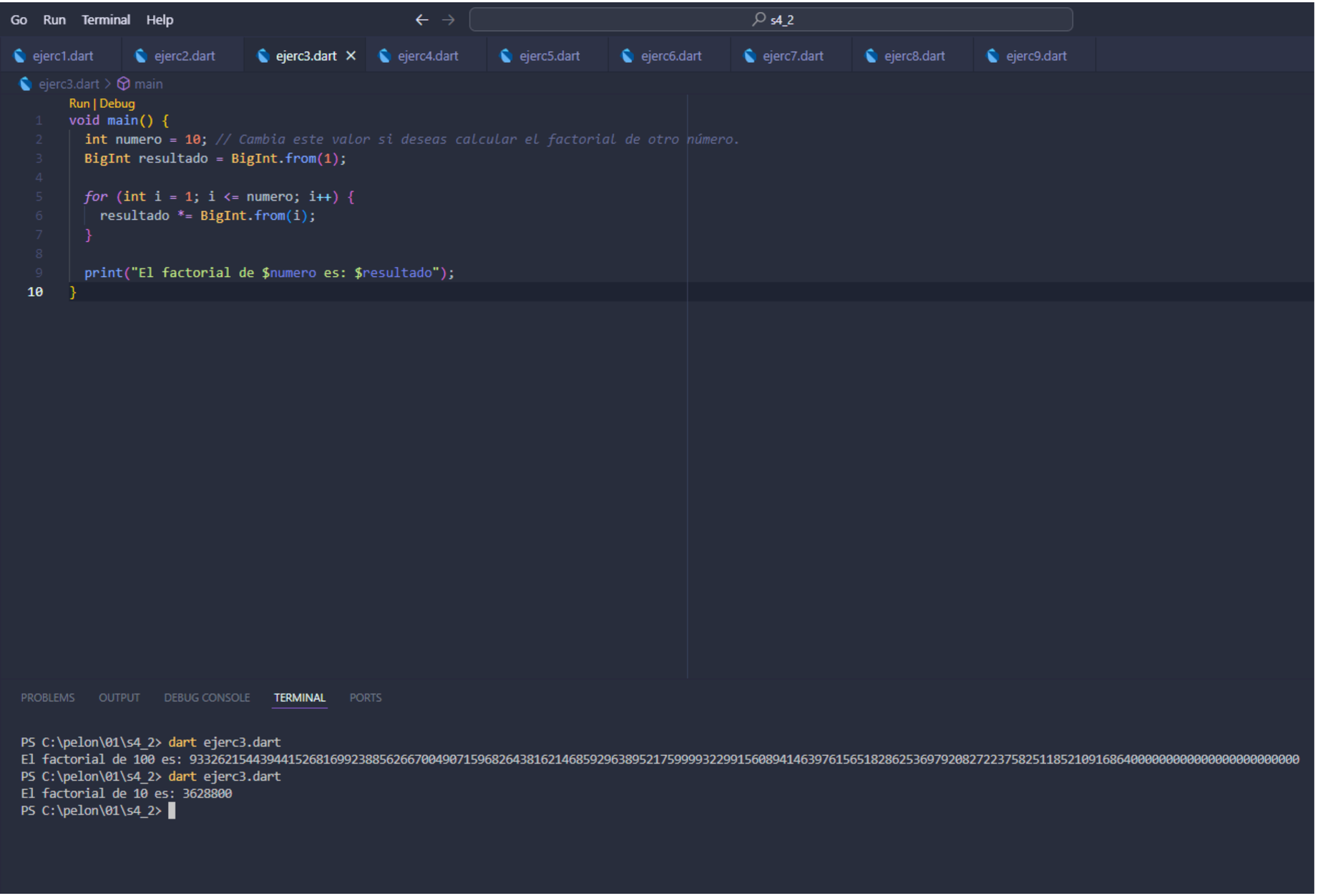
print("La sumatoria de los números primos entre $inicio y $fin es: $sumatoriaPrimos");
}

void main() {
int n = 10; // obtener más o menos términos de la secuencia

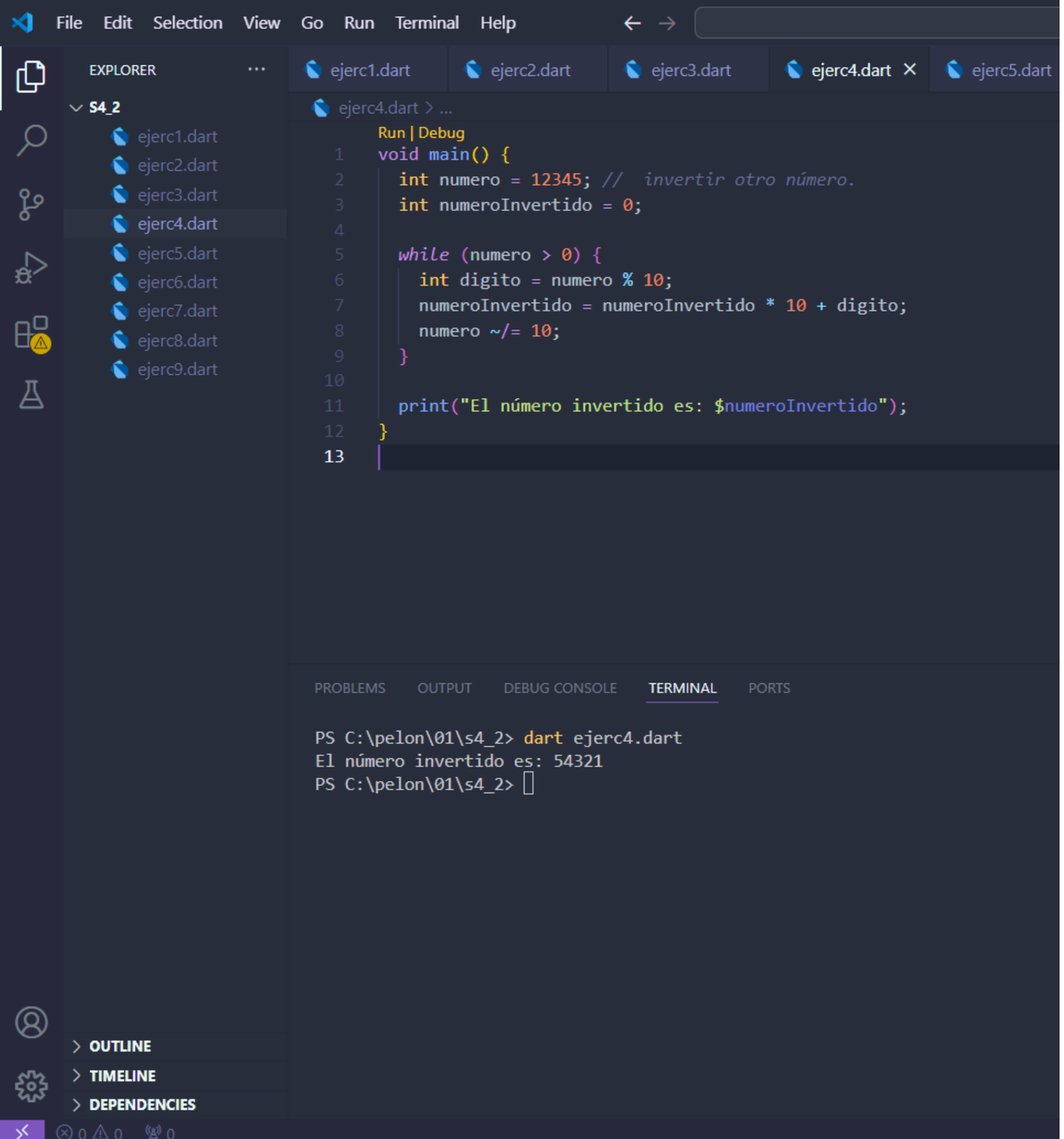
int a = 0;
int b = 1;

print("Secuencia de Fibonacci de $n términos:");

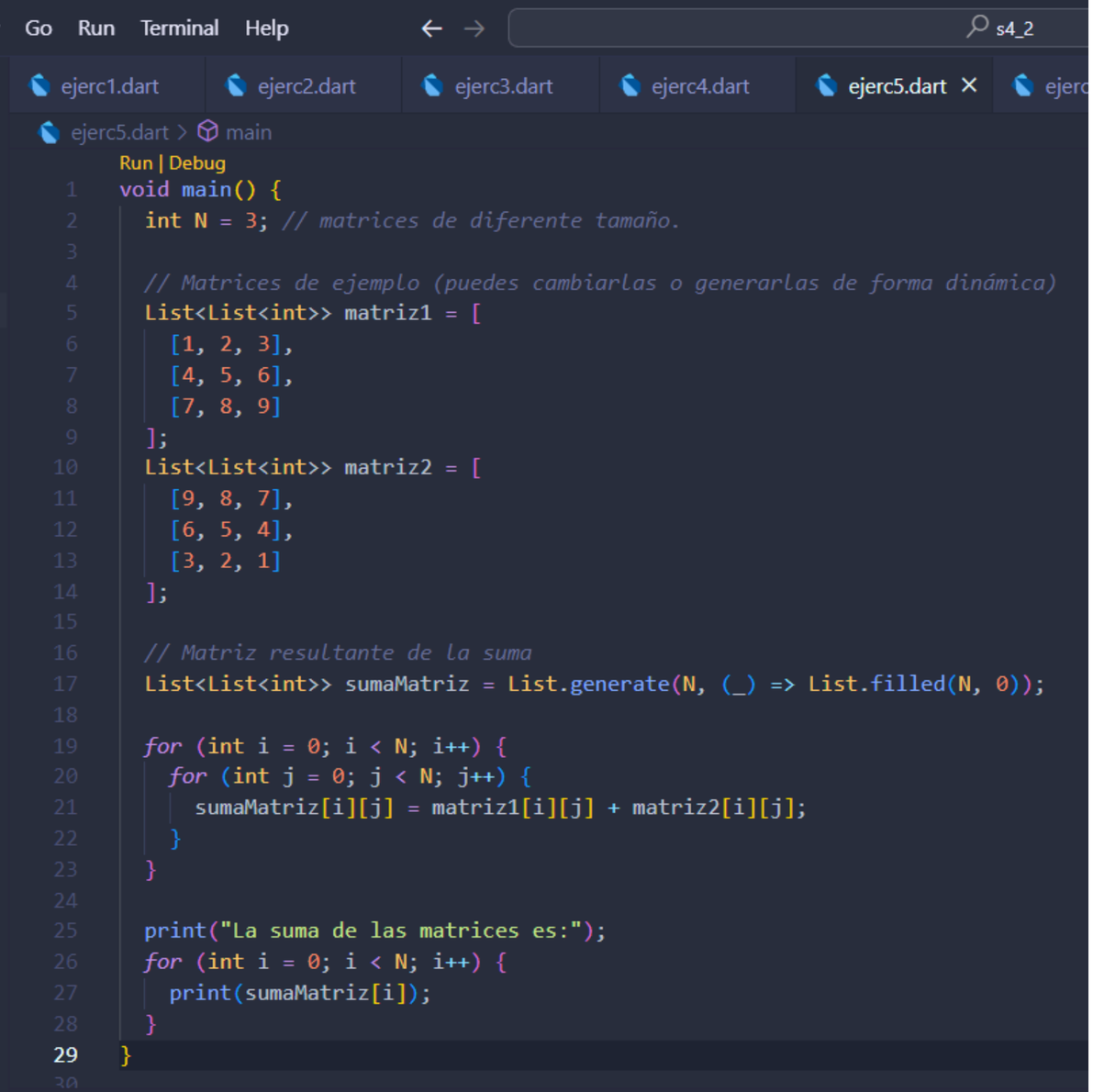
for (int i = 0; i < n; i++) {
print(a);
int temp = a + b;
a = b;
b = temp;
}
}




```
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");
}
```

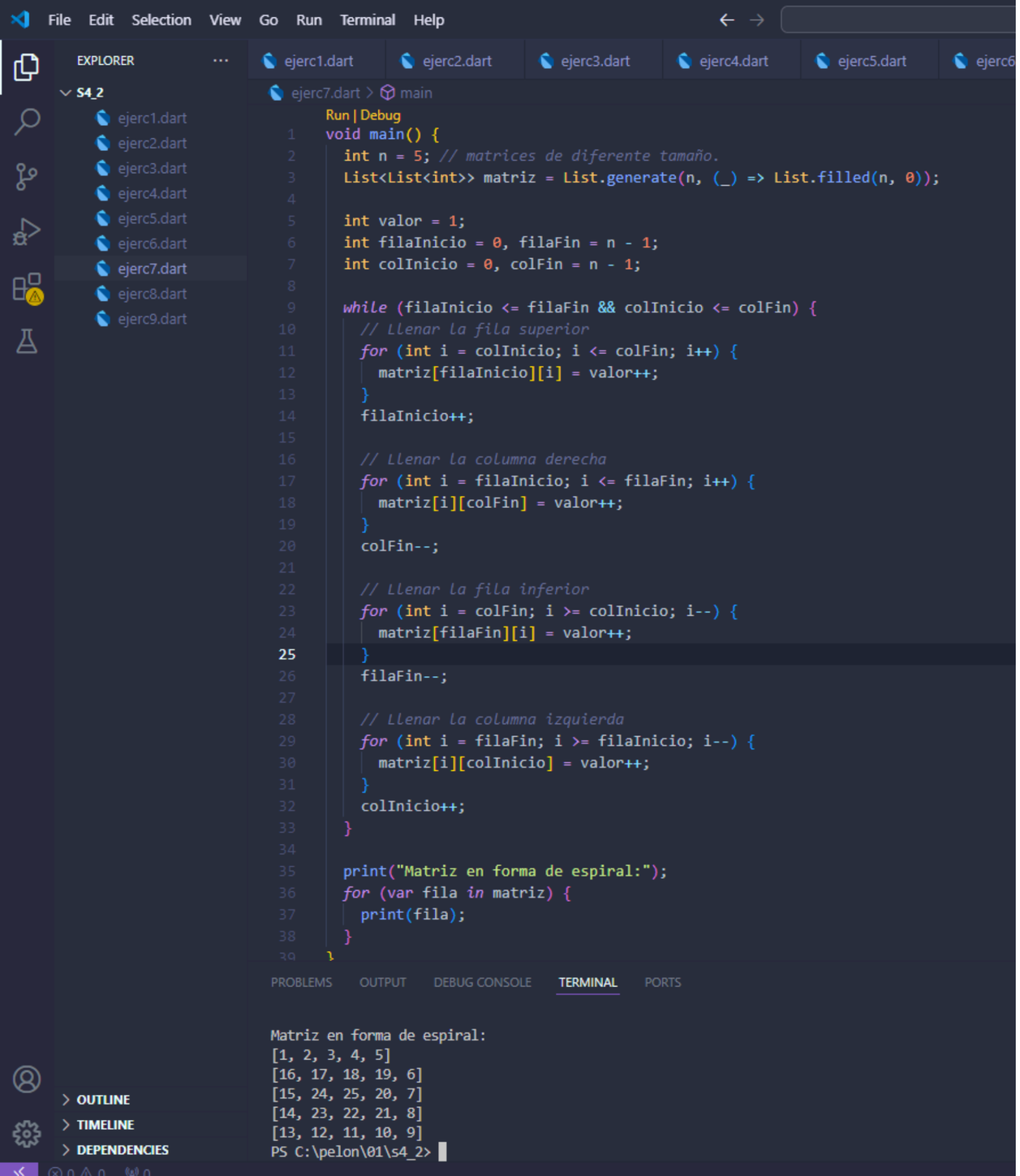



```
void main() {
int numero = 12345; //  invertir otro número.
int numeroInvertido = 0;

while (numero > 0) {
int digito = numero % 10;
numeroInvertido = numeroInvertido * 10 + digito;
numero ~/= 10;
}

print("El número invertido es: $numeroInvertido");
}
```

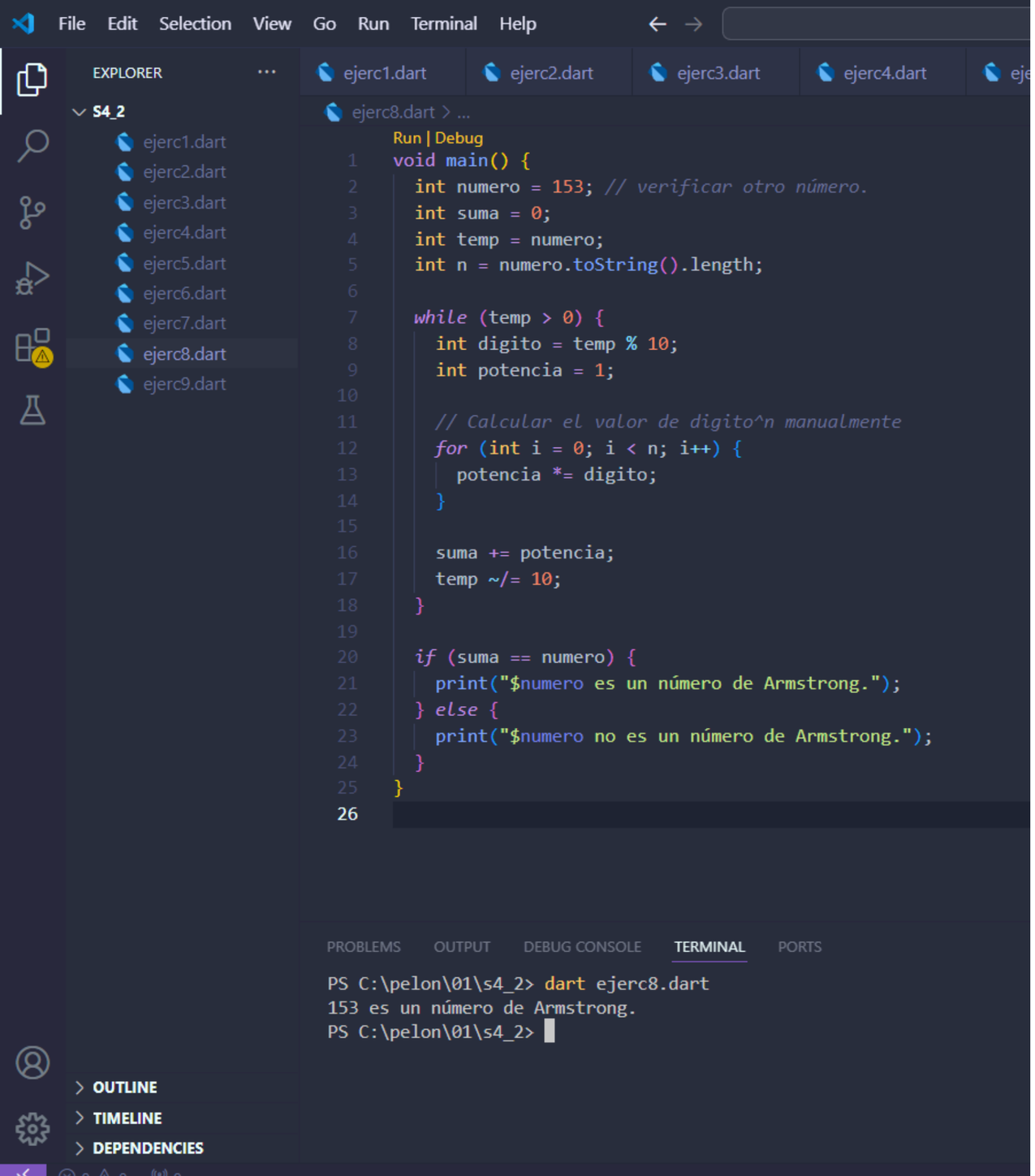



```
void main() {
int N = 3; // matrices de diferente tamaño.

// 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]);
}
}
```

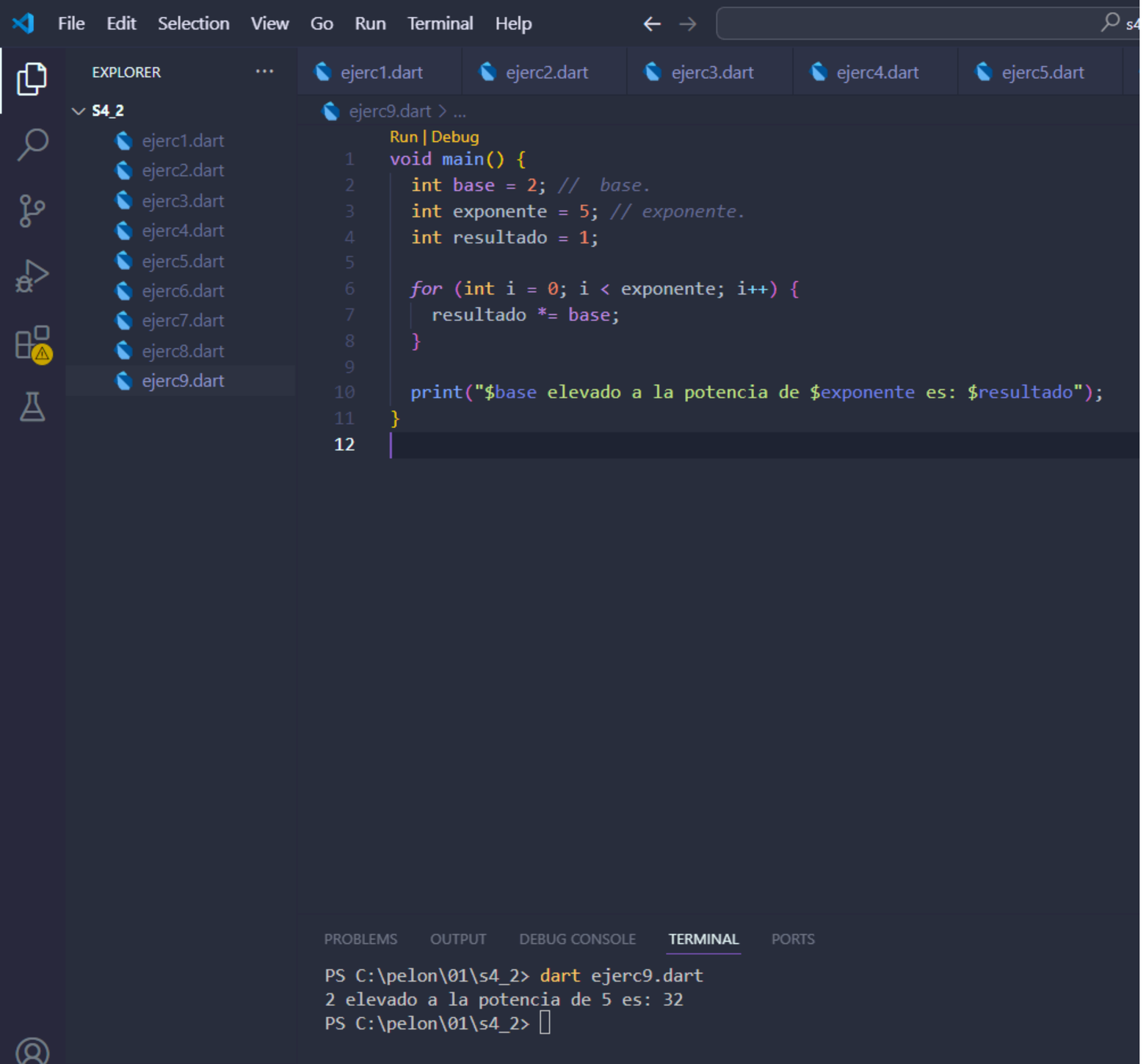

```
void main() {
print("Números perfectos entre 1 y 10,000:");

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);
}
}
}
```




```
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");
}
```