**Nombre y Apellido: César Barreto C.I: 30293326**

1. Dado una secuencia de números naturales, finalizada con un cero, construya un algoritmo que determine: el mayor, el menor y el promedio de los números perfectos y a su vez malvados, según la definición dada a continuación.

**Número perfecto:** es todo número natural que es igual a la suma de sus divisores propios (es decir, todos sus divisores excepto el propio número). Por ejemplo, 6 es un número perfecto ya que sus divisores propios son 1, 2, y 3 y se cumple que 1+2+3=6.

**Número malvado:** todo número natural cuya expresión en base 2 (binaria) contiene un número par de unos. Por ejemplo, el 12 y el 15 son números malvados ya que 12=1100 contiene dos (2) unos en su representación binaria y 15=1111 contiene cuatro (4) unos en su representación binaria (dos y cuatro son pares).

**Nota:** Para llevar un número natural a binario, hacer divisiones sucesivas entre 2 y tomar el resto.

**Análisis del problema**

|  |  |  |
| --- | --- | --- |
| **Entrada** | **Proceso** | **Salida** |
| Número natural: Num | Pedir los datos de entrada  Determinar el número mayor y menor de la secuencia, según los parámetros dados.  Calcular el promedio, según los parámetros dados.  Mostrar los datos de salida. | Mostrar el número mayor y menor de la secuencia.  Mostrar el promedio de los números perfectos y malvados de la secuencia. |

**Pseudocodigo**

**Cabecera:** SecuenciadeNumeros;

**Declaraciones**

Mayor, Menor, Cont\_Uno, Num, Num\_Perf, i, Salv\_num, Prom\_Parcial, Cont\_PM: Entero;

Prom\_PM: Real;

**Cuerpo**

**Inicio**

Mayor <- 0;

Menor <- 30000;

Cont\_PM <- 0;

Prom\_Parcial<- 0;

Salv\_Menor <- Menor;

Repetir

Escribir (“Ingrese una secuencia de numeros naturales (Para finalizar la secuencia, debe ingresar un cero (0)) :”);

Leer (Num);

Si (Num < 0) Entonces

Escribir (“Usted ha ingresado un valor invalido.”);

Fin-Si;

Hasta (Num >= 0);

Fin-Repetir hasta;

Repetir Mientras (Num <> 0) Hacer

Num\_Perf <- 0;

Para i, 1, (num – 1,), +1 Hacer

Si (Num Mod i = 0) Entonces

Num\_Perf <- Num\_Perf + i;

Fin-Si;

Fin-Repetir para;

Cont\_Uno <- 0;

Salv\_num <- num;

Repetir

Si (Num Mod 2 <> 0) Entonces

Cont\_Uno <- Cont\_Uno + 1;

Num <- Num Div 2;

Hasta (Num = 1);

Fin-Repetir hasta;

Cont\_Uno <- Cont\_Uno + 1;

Si (Cont\_Uno Mod 2 = 0) O (Num\_Perf = Salv\_num) Entonces

Prom\_Parcial <- Prom\_Parcial + Salv\_num;

Cont\_PM <- Cont\_PM + 1;

Si (Salv\_Num > Mayor) Entonces

Mayor <- Salv\_num;

Fin-Si;

Si (Salv\_Num < Menor) Entonces

Menor <- Salv\_Num;

Fin-Si;

Fin-Si;

Escribir (“Ingrese el siguiente numero (Recuerde que para finalizar la secuencia, ingrese un cero(0)):”);

Leer (Num);

Fin-Repetir mientras

Si (Menor = Salv\_Menor) Entonces

Menor <- 0;

Fin-Si;

Si (Cont\_PM <> 0) Entonces

Prom\_PM <- Prom\_Parcial / Cont\_PM;

Fin-Si;

Escribir (“Numero mayor de la secuencia: “, Mayor);

Escribir (“Numero menor de la secuencia: “, Menor);

Escribir (“Promedio de los numeros perfectos y malvados de la secuencia: “, Prom\_PM);

**Fin.**

**Código fuente de Pascal**

Program SecuenciadeNumeros;

Uses

Crt;

Var

Mayor, Menor, Cont\_Uno, Num, Num\_Perf, i, Salv\_num, Prom\_Parcial,

Cont\_PM, Salv\_Menor: Integer;

Prom\_PM: Real;

Begin

Mayor := 0;

Menor := 30000;

Cont\_PM := 0;

Prom\_Parcial:= 0;

Salv\_Menor <- Menor;

//Validando que la entrada sea un numero natural

Repeat

clrscr;

Writeln ('Ingrese una secuencia de numeros naturales (Para finalizar la secuencia, debe ingresar un cero (0)) :');

Readln (Num);

If (Num < 0) Then

Begin

Writeln ('Usted ha ingresado un valor invalido.');

Readkey;

End;

Until (Num >= 0);

While (Num <> 0) Do

Begin

//Determinando si el numero ingresado es un numero perfecto

Num\_Perf := 0;

For i:= 1 To (num - 1) Do

Begin

If (Num Mod i = 0) Then

Num\_Perf := Num\_Perf + i;

End;

//Determinando los valores binarios del número ingresado

Cont\_Uno := 0;

Salv\_num := num;

Repeat

If (Num Mod 2 <> 0) Then

Cont\_Uno := Cont\_Uno + 1;

Num := Num Div 2;

Until (Num = 1) ;

Cont\_Uno := Cont\_Uno + 1;

//Determinando si el numero ingresado es perfecto y a su vez malvado

If (Cont\_Uno Mod 2 = 0) Or (Num\_Perf = Salv\_num)Then

Begin

Prom\_Parcial := Prom\_Parcial + Salv\_num;

Cont\_PM := Cont\_PM + 1;

//Determinando el numero mayor y menor de los números perfectos y malvados

if (Salv\_Num > Mayor) Then

Mayor := Salv\_num;

If (Salv\_Num < Menor) Then

Menor := Salv\_Num;

End;

Writeln ('Ingrese el siguiente numero (Recuerde que para finalizar la secuencia, ingrese un cero(0)):');

Readln (Num);

End;

//Calculo del promedio de numeros perfectos y malvados de la secuencia

If (Menor = Salv\_Menor) Then

Menor := 0;

If (Cont\_PM <> 0) Then

Prom\_PM := Prom\_Parcial / Cont\_PM;

Writeln ('Numero mayor de la secuencia: ', Mayor);

Writeln ('Numero menor de la secuencia: ', Menor);

Writeln ('Promedio de los numeros perfectos y malvados de la secuencia: ',

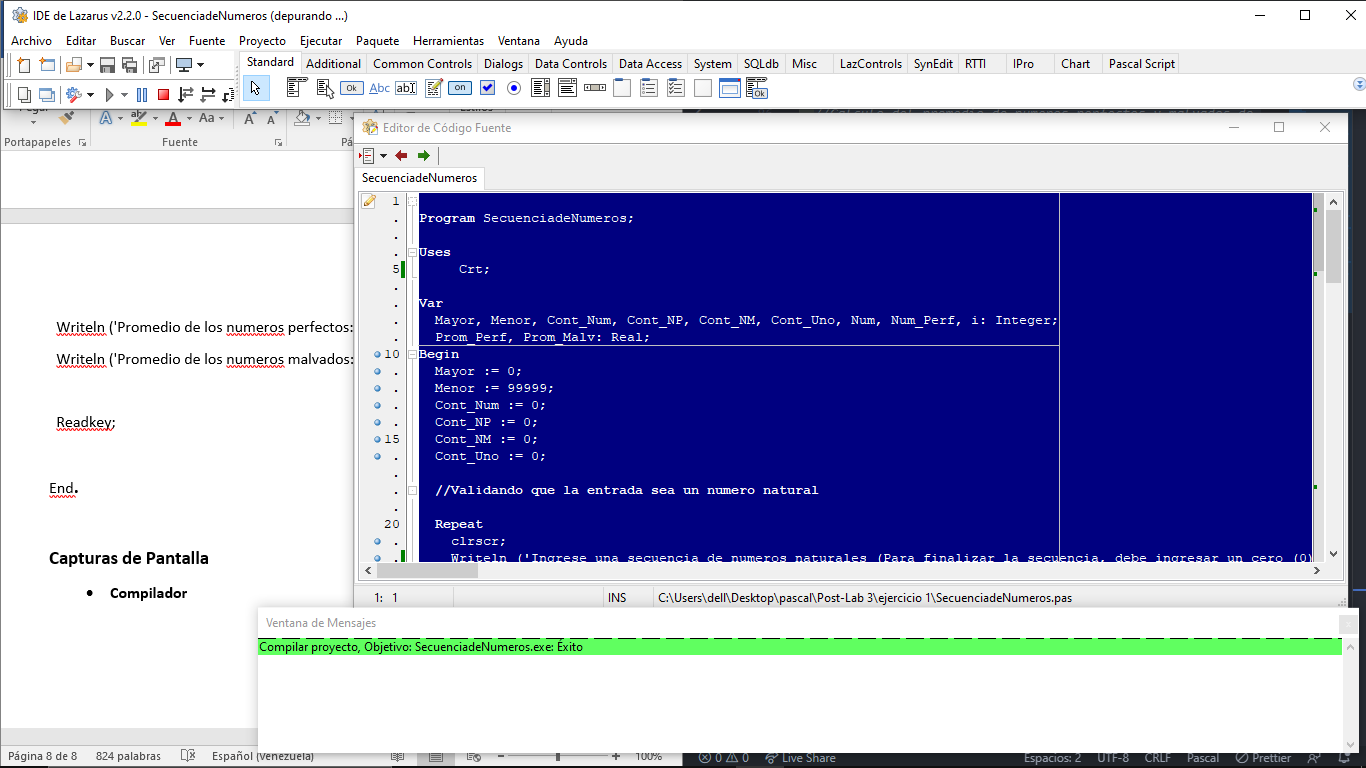
Prom\_PM:0:2);

Readkey;

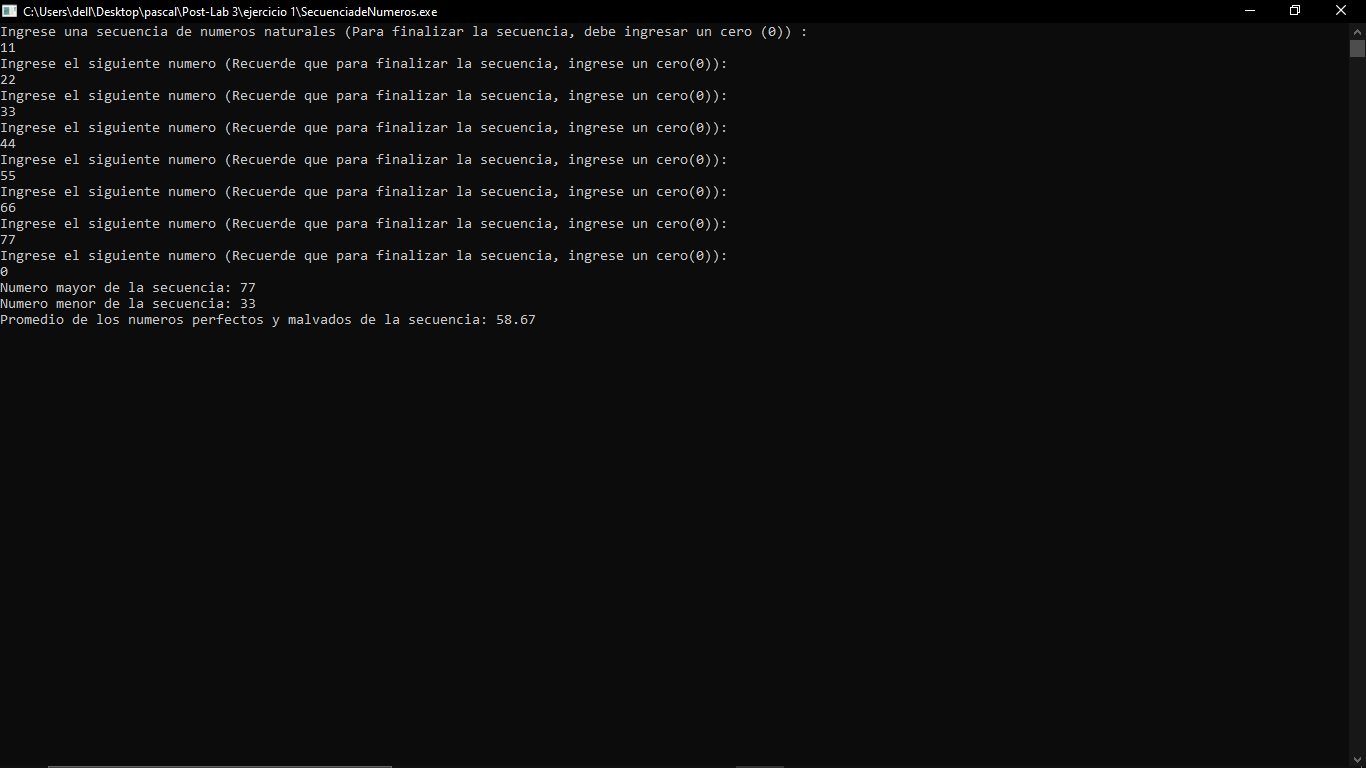
End.

**Capturas de Pantalla**

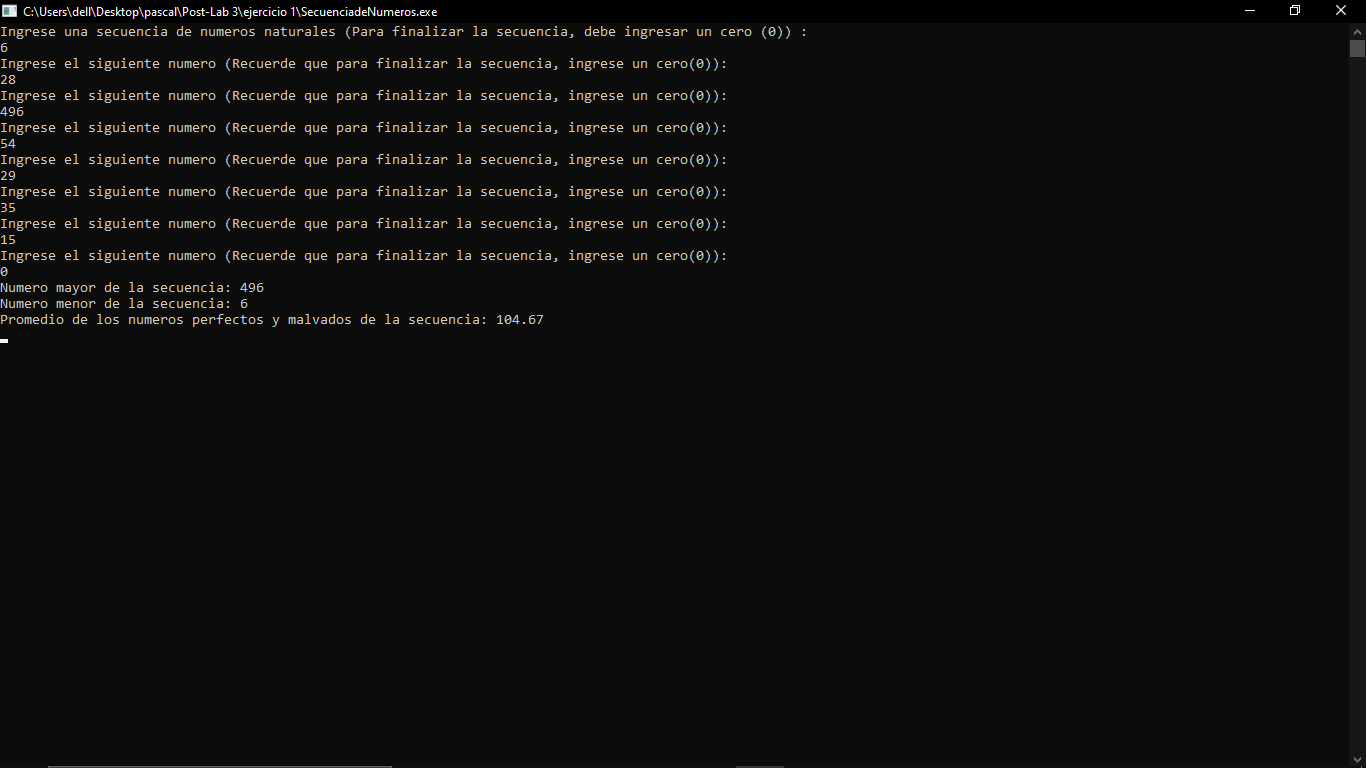
* **Compilador**



* **Ejemplo 1**



* **Ejemplo 2**



1. Construya un programa que lea los elementos de una secuencia de números enteros que termina con un 0. Si la secuencia es estrictamente descendente el programa debe escribir el mensaje **“Su lista de N elementos es descendente”**, donde N es la cantidad de elementos de la lista, incluyendo el 0. Si la secuencia no es estrictamente descendente debe escribir el mensaje **“Su lista de N elementos no es descendente**, ya que aparece **A** seguido de **B”**, donde **A** y **B** son dos números que aparecen en la secuencia, **A** inmediatamente antes de **B** y **B** no es menor que **A.**

Por ejemplo, si la secuencia en la entrada es 10, 9, 4, 2, 0, la salida debe ser: Su lista de 5 elementos es descendente.

Por otra parte, si la secuencia en la entrada es 8, 6, 5, 7, 6, 8, 0, la salida debe ser: Su lista de 7 elementos NO es descendente, ya que aparece 5 seguido de 7.

De igual manera para la lista 6, 3, 3, 0, la salida debe ser: Su lista de 4 elementos NO es descendente ya que aparece 3 seguido de 3.

**Nota: Para la corrida de este programa usar los datos de los ejemplos dados**.

**Análisis del problema**

|  |  |  |
| --- | --- | --- |
| **Entrada** | **Proceso** | **Salida** |
| Número de la secuencia: Num | Pedir los datos de entrada  Determinar si la secuencia es descendente o no, siguiendo los parámetros dados.  Mostar los datos de salida | Mostrar si la secuencia es descendente o no. |

**Pseudocodigo**

**Cabecera:** SecuenciaDescendente;

**Declaraciones**

Num, Num\_A, Num\_B, Cont\_Num: Entero;

Corte: Booleano;

**Cuerpo**

**Inicio**

Cont\_Num <- 0;

Corte <- Verdadero;

Num\_A <- 99999;

Repetir

Escribir (“Ingrese una secuencia de números enteros. Para finalizar la secuencia, ingrese un cero (0):”);

Leer (Num);

Cont\_Num <- Cont\_Num + 1;

Si (Num\_A <= Num) Y (Corte =true) Entonces

Corte <- falso;

Num\_B <- Num;

Fin-Si;

Si (Corte = Verdadero) Entonces

Num\_A := Num;

Fin-Si;

Hasta (Num = 0);

Fin- Repetir Hasta;

Si (Corte = Verdadero) Entonces

Escribir (“Su lista de “, Cont\_Num, “elementos es descendente.”);

Sino

Escribir (“Su lista de “, Cont\_Num, “ elementos NO es descendente, ya que aparece “, Num\_A, “ seguido de “, Num\_B,”.”);

Fin-Si;

**Fin.**

**Código Fuente Pascal**

Program SecuenciaDescendente;

Uses

Crt;

Var

Num, Num\_A, Num\_B, Cont\_Num: Integer;

Corte: Boolean;

Begin

Cont\_Num := 0;

Corte := true;

Num\_A:= 99999;

Repeat

Writeln ('Ingrese una secuencia de numeros enteros. Para finalizar la secuencia, ingrese un cero (0): ');

Readln(Num);

//Contador de los números ingresados

Cont\_Num := Cont\_Num + 1;

//Determinando si el numero ingresado, rompe la secuencia descendente

If (Num\_A <= Num) And (Corte =true) Then

Begin

Corte := false;

Num\_B := Num;

End;

If (Corte=true) Then

Num\_A := Num;

Until (Num = 0);

If (Corte = true) Then

Writeln('Su lista de ', Cont\_Num,' elementos es descendente.')

Else

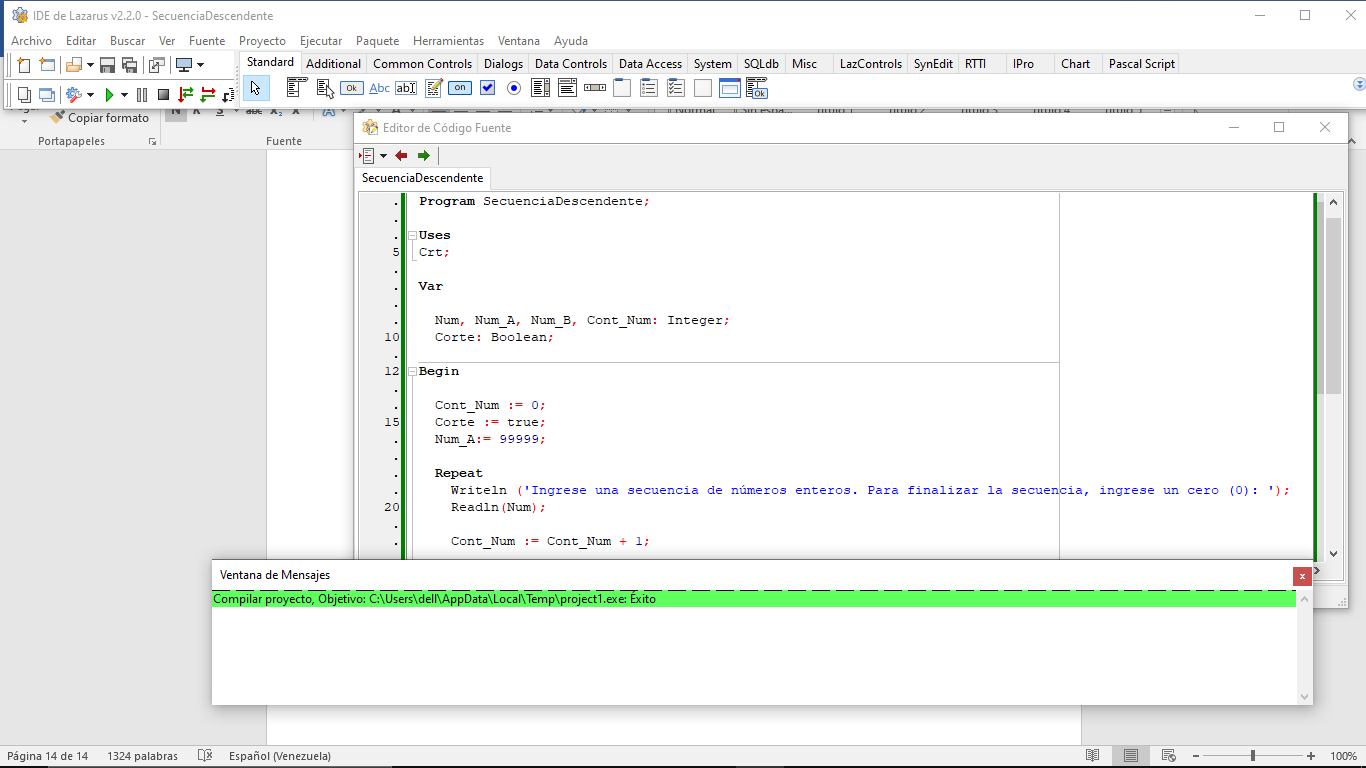
Writeln('Su lista de ', Cont\_Num, ' elementos NO es descendente, ya que aparece ', Num\_A, ' seguido de ', Num\_B,'.');

Readkey;

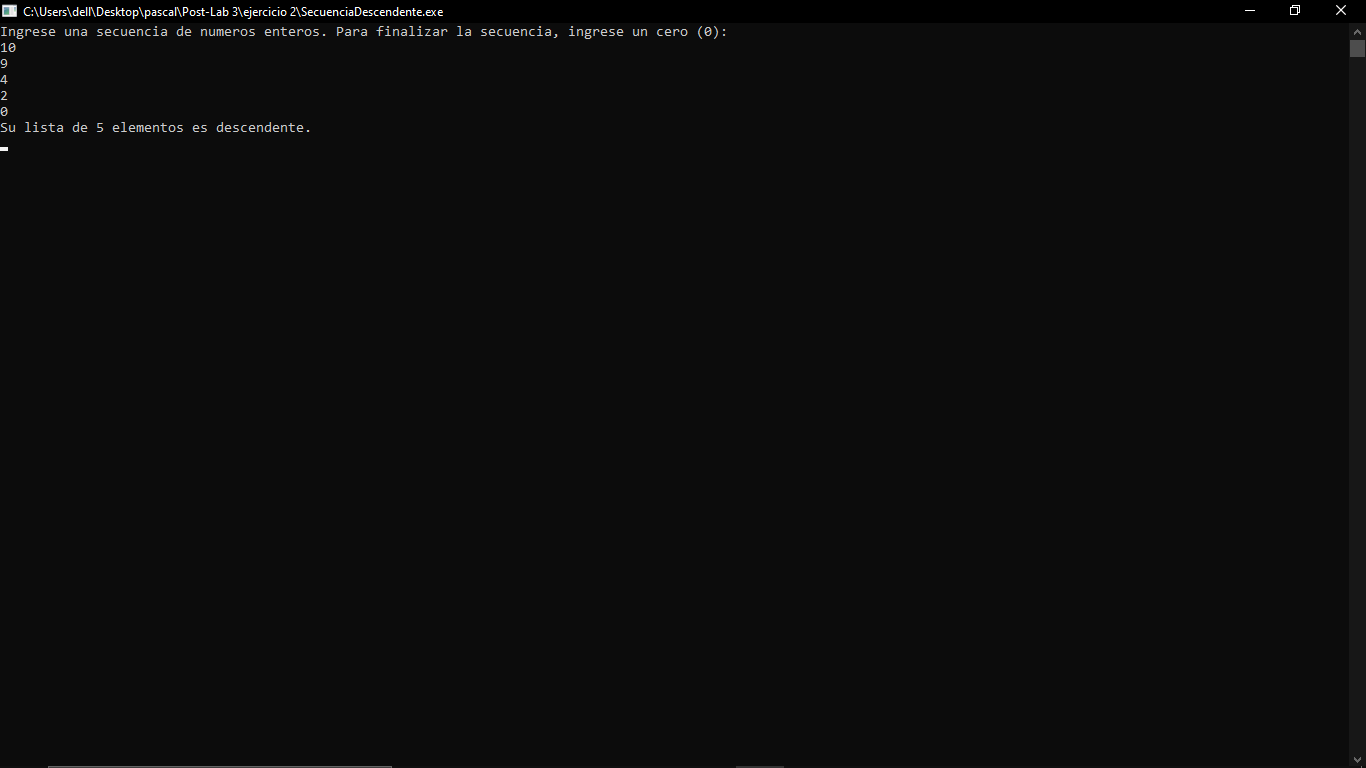
End**.**

**Capturas de pantalla**

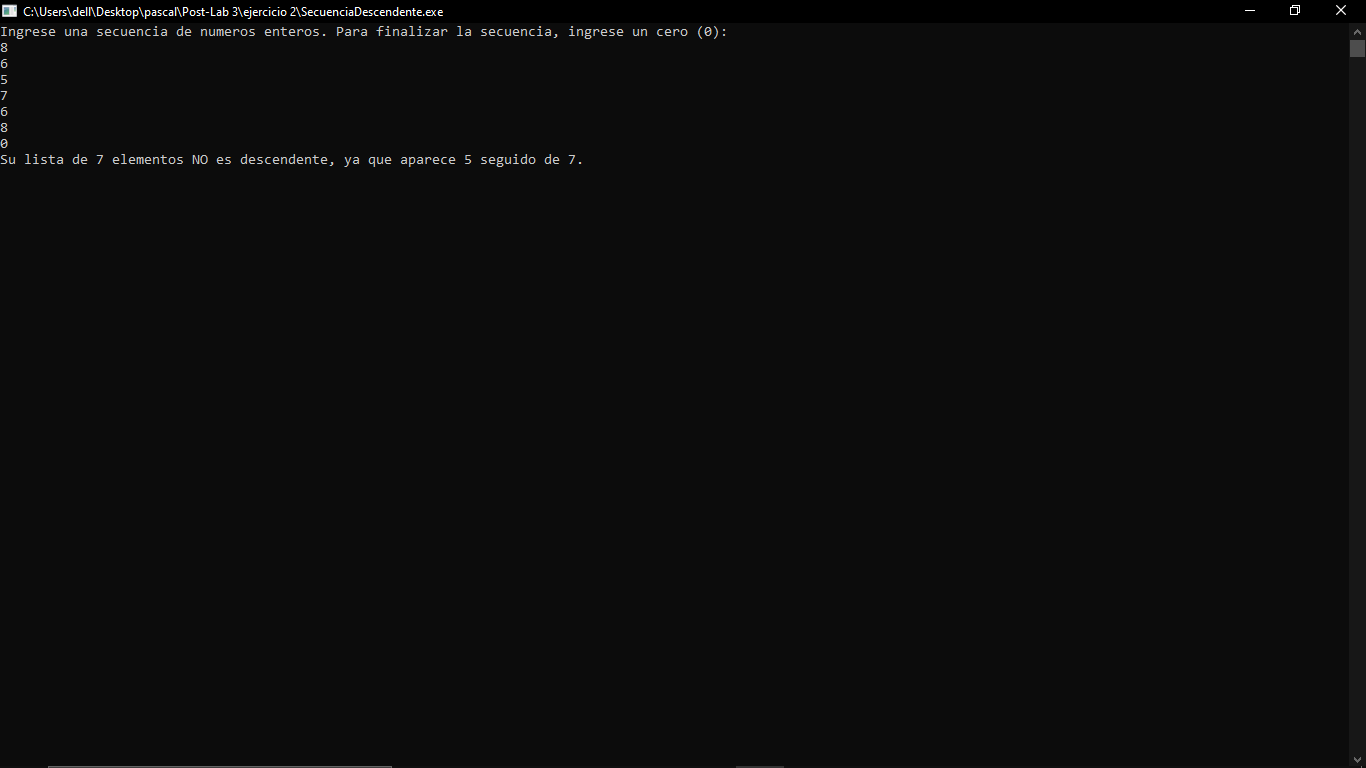
* **Compilador**



* **Ejemplo 1**



* **Ejemplo 2**



* **Ejemplo 3**

