

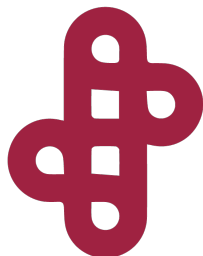


UNIVERSIDAD
ROSARIO CATELLANOS

DISEÑAR Y REPRESENTAR ALGORITMOS

LUIS RENÉ OROROZCO GONZÁLEZ

Ciudad de México, 2023



Primer algoritmo

Ejercicio 1

Diseña un algoritmo que imprima los números naturales ≤ 100 en orden ascendente

A continuación se muestra el diagrama de flujo del algoritmo solicitado, además se añade las capturas de pantalla del pseudocódigo y la pantalla de salida de la ejecución del pseudocódigo.

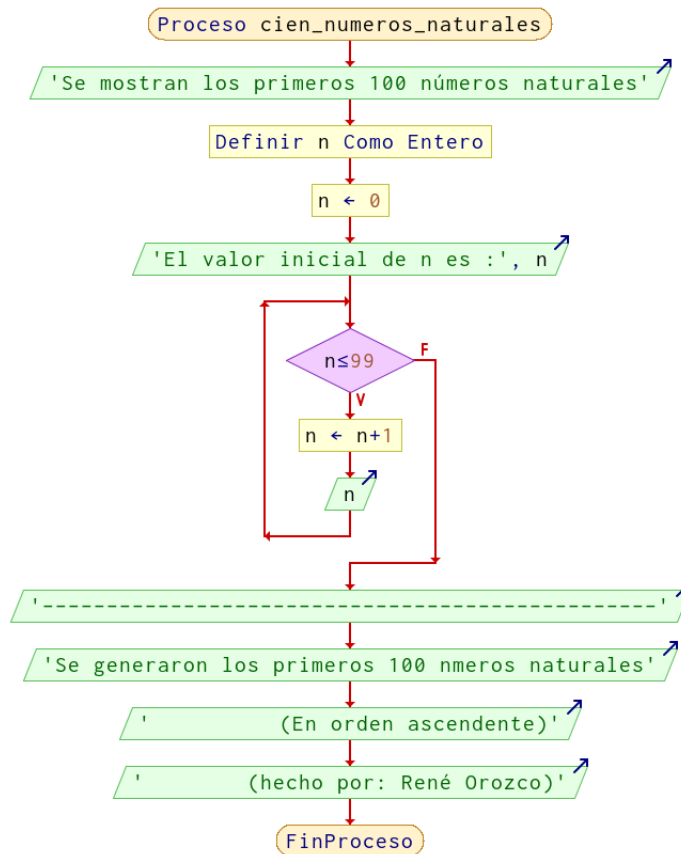


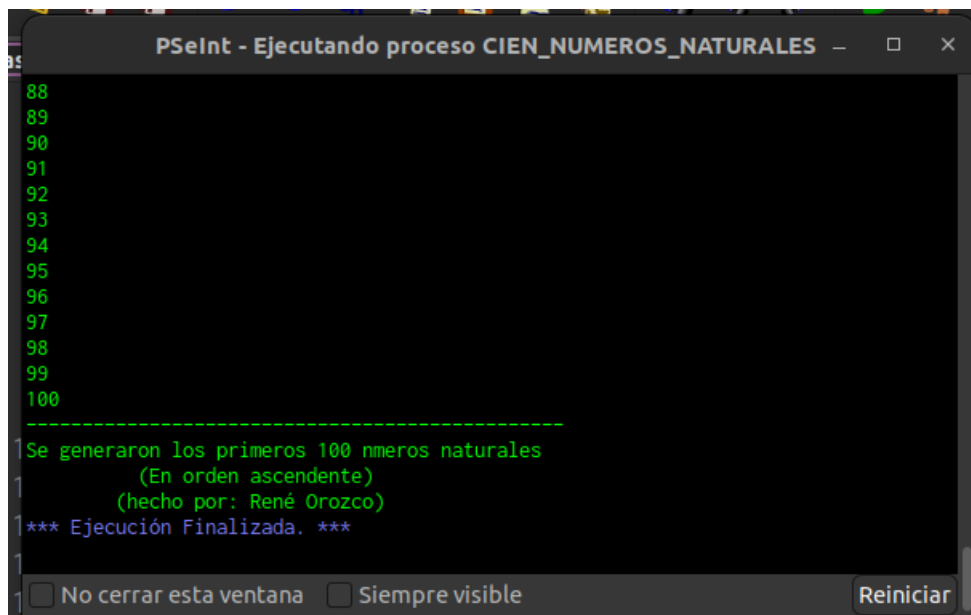
Figura 1: Diagrama del algoritmo para imprimir los primeros 100 números naturales. Desarrollado en pseint.

```

1  Proceso cien_numeros_naturales
2      Escribir 'Se muestran los primeros 100 números naturales';
3      // Declaramos a 'n' como una variable
4      // de tipo entero
5      Definir n Como Entero;
6      // La inicializamos en 0
7      n ← 0;
8      // Definimos un ciclo donde cada vez que
9      // la variable sea menor a 99 se le sumara una unidad
10     // y imprimira el valor actual de 'n'.
11     // en el momento en el que la variable 'n' no cumpla
12     // la condición osea sea mayor a 99 se terminara el bucle
13     Escribir 'El valor inicial de n es :', n;
14     Mientras n≤99 Hacer
15         n ← n+1;
16         Escribir n;
17     FinMientras
18     Escribir '-----';
19     Escribir 'Se generaron los primeros 100 nmeros naturales';
20     Escribir '          (En orden ascendente)';
21     Escribir '          (hecho por: René Orozco)';
22 FinProceso
23

```

Figura 2: Captura de pantalla donde se muestra el pseudocódigo desarrollado.



```

88
89
90
91
92
93
94
95
96
97
98
99
100
-----
Se generaron los primeros 100 nmeros naturales
      (En orden ascendente)
      (hecho por: René Orozco)
*** Ejecución Finalizada. ***

```

☐ No cerrar esta ventana ☐ Siempre visible Reiniciar

Figura 3: Captura de pantalla donde se muestra la salida de ejecución del pseudocódigo.

Segundo algoritmo

Ejercicio 2

Diseña un algoritmo que imprima los números naturales ≤ 100 en orden descendente

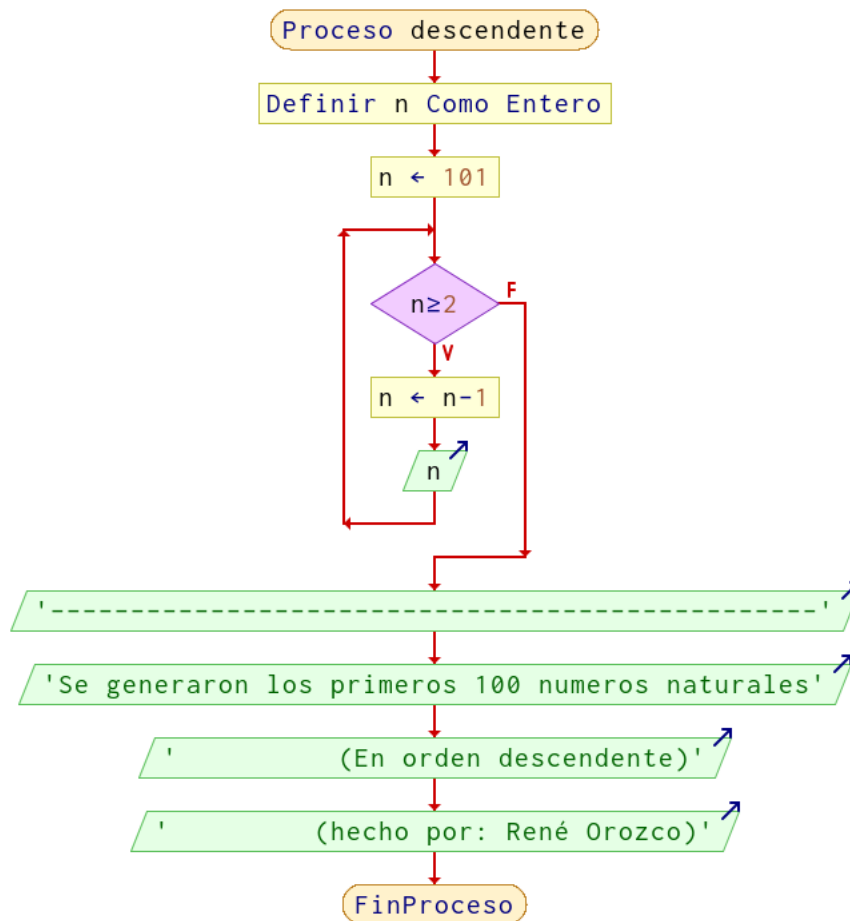


Figura 4: Diagrama del algoritmo para imprimir los primeros 100 números naturales. Desarrollado en pseint

```

1  Proceso descendente
2      // Declaramos a 'n' como una variable de tipo entero
3      // y le damos el valor de 101.
4      Definir n Como Entero;
5      n ← 101;
6      // Definimos un ciclo donde si 'n' es mayor o igual a dos
7      // se llevara acabo una resta de una unidad y esto continuara
8      // hasta que el valor de 'n' ya no cumpla la condición de ciclo.
9      // Por otra parte de manera simulatnea se escribira el valor de '
10     // Lo que debe de mostrarnos como el valor de dicha variable va d
11     // con respecto a su valor inicial.
12     Mientras n ≥ 2 Hacer
13         n ← n-1;
14         Escribir n;
15     FinMientras
16     Escribir '-----';
17     Escribir 'Se generaron los primeros 100 numeros naturales';
18     Escribir '          (En orden descendente)';
19     Escribir '          (hecho por: René Orozco)';
20 FinProceso
21

```

Figura 5: Captura de pantalla donde se muestra el pseudocódigo desarrollado.

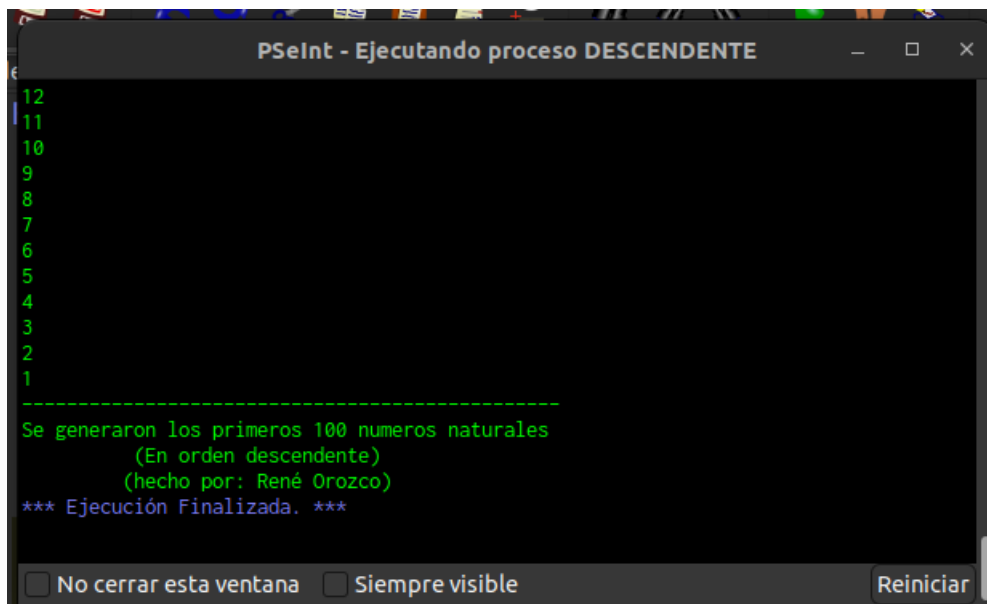


Figura 6: Captura de pantalla donde se muestra la salida de ejecución del pseudocódigo.

Tercer algoritmo

Ejercicio 3

Diseña un algoritmo que imprima 10 números pares < 20 en orden ascendente.

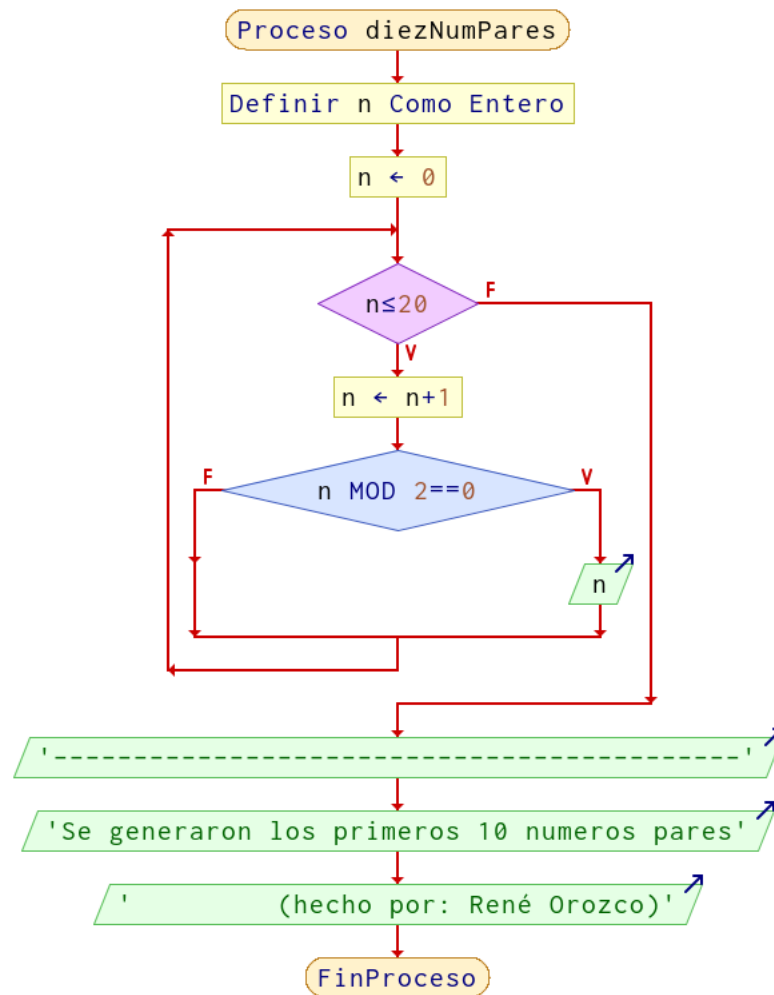


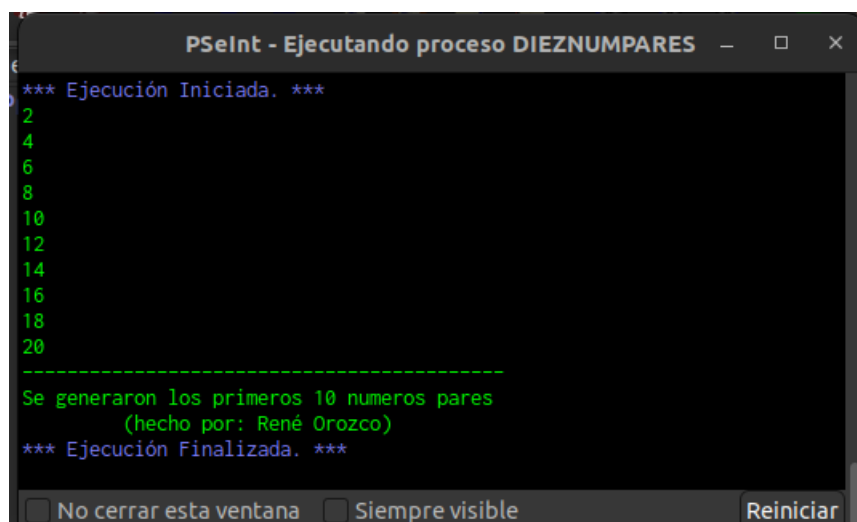
Figura 7: Diagrama del algoritmo para imprimir los primeros 20 números pares. Desarrollado en pseint

```

1  Proceso diezNumPares
2      // Declaramos a la variable 'n' de tipo entero
3      Definir n Como Entero;
4      // la iniciamos en cero
5      n ← 0;
6      // Iniciamos un ciclo donde la variable 'n'
7      // es comparada, tal que si es menor igual a 20
8      // se le suma una unidad.
9      Mientras n≤20 Hacer
10         n ← n+1;
11         // Se declara una estructura condicional donde
12         // se calcula el modulo 2 de la variable 'n'
13         // y aquel valor 'n' que su modulo 2 sea igual a cero
14         // se mandara a imprimir en consola.
15         // Por lo que solo se moostaran los numeros multiplos de 2
16         Si n MOD 2==0 Entonces
17             Escribir n;
18         FinSi
19     FinMientras
20     Escribir '-----';
21     Escribir 'Se generaron los primeros 10 numeros pares';
22     Escribir '          (hecho por: René Orozco)';
23 FinProceso
24

```

Figura 8: Captura de pantalla donde se muestra el pseudocódigo desarrollado.



```

PSeInt - Ejecutando proceso DIEZNUMPARES
*** Ejecución Iniciada. ***
2
4
6
8
10
12
14
16
18
20
-----
Se generaron los primeros 10 numeros pares
(hecho por: René Orozco)
*** Ejecución Finalizada. ***
☐ No cerrar esta ventana ☐ Siempre visible 

```

Figura 9: Captura de pantalla donde se muestra la salida de ejecución del pseudocódigo.

Cuarto algoritmo

Ejercicio 4

Algoritmo que imprima los divisores de un número natural

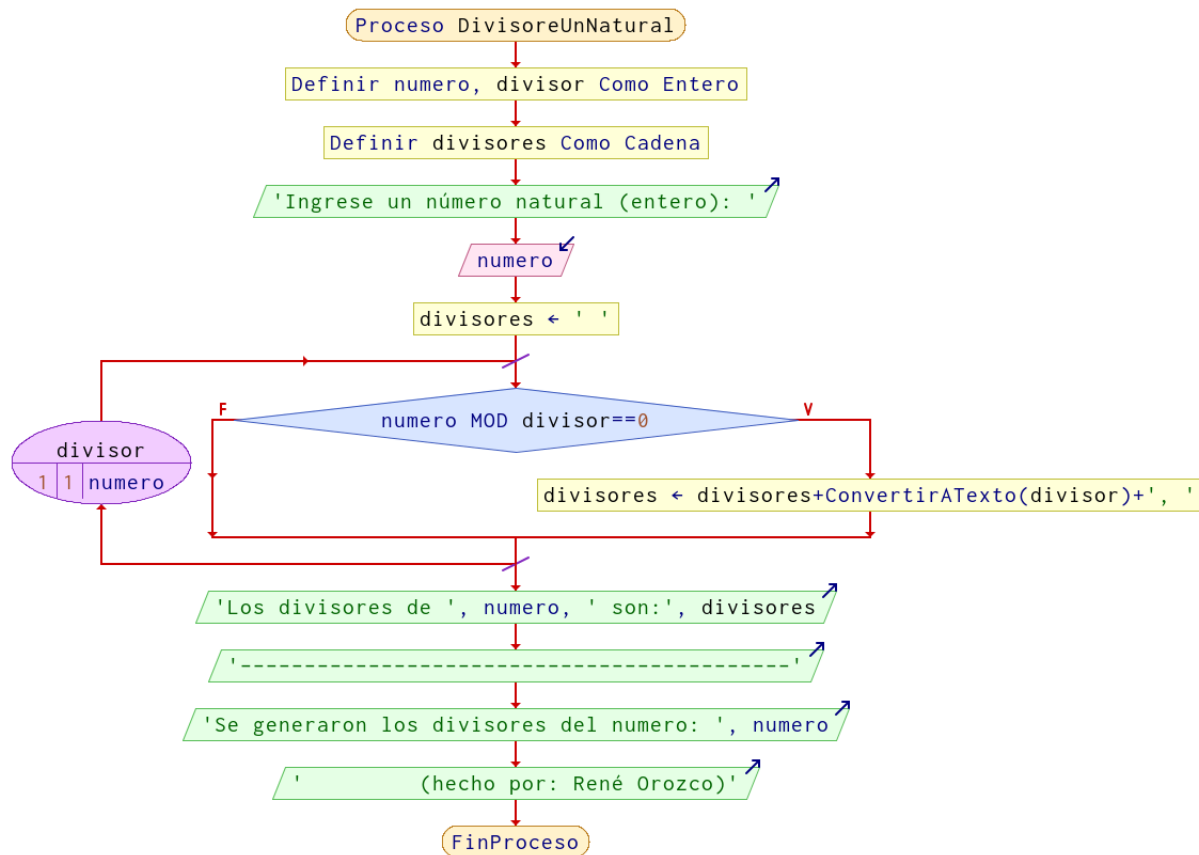


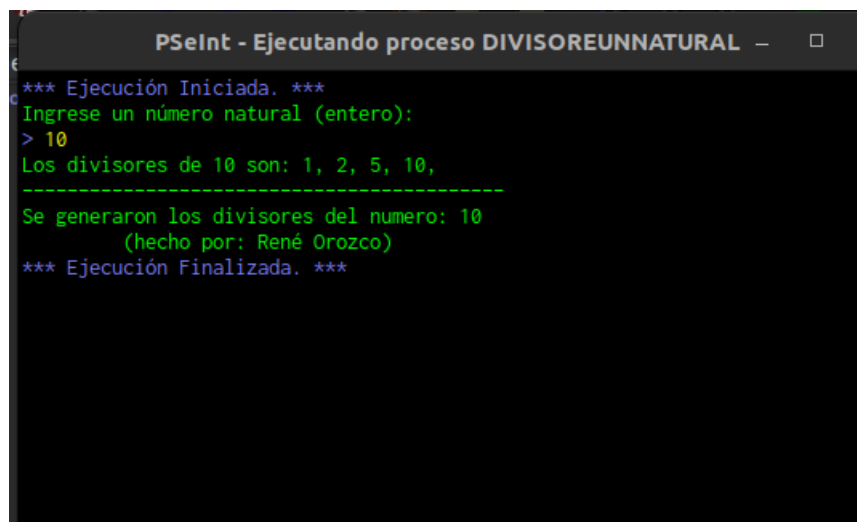
Figura 10: Diagrama del algoritmo para imprimir los divisores de un numero natural. Desarrollado en pseint


```

1  Proceso DivisoreUnNatural
2  // Declaramos dos variables de tipo entero
3  // con nombre 'numero' y 'divisor'
4  Definir numero, divisor Como Entero;
5  // Declaramos otra variable con nombre 'divisores'
6  // de tipo string o Caracter.
7  Definir divisores Como Cadena;
8  // Le solicitamos al usuario que ingrese un numero
9  Escribir 'Ingrese un número natural (entero): ';
10 // Con el comando 'Leer' almacenamos el numero que ingreso
11 // el usuario y lo asociamos a la variable 'numero'
12 Leer numero;
13 // Inicializamos la variable 'divisores' como una cadena vacia
14 divisores ← ' ';
15 // Iniciamos un ciclo donde se asocia el valor '1' a la variable
16 // de tipo entero 'divisor' de tal manera que tiene que recorrer
17 // desde ese valor hasta el numero que ingrese el usuario con un
18 // paso de unidad por unidad y de manera simultanea va a realizar
19 // acción
20 Para divisor←1 Hasta numero Con Paso 1 Hacer
21     // Lo que se tiene que realizar en ese recorrido es una estructura
22     // condicional donde si el resultado del modulo entre el numero
23     // ingresado y el divisor en cuestion es igual es cero, Entonces
24     // la variable de tipo cadena con nombre 'divisores' se le concatenara
25     // el valor que este asociado la variable divisor, pero como en
26     // principio 'divisores' y 'divisor' son variables de diferente tipo
27     // de dato lo que se realiza es una conversión con la Funcion
28     // 'ConvertirATexto()' para que no halla problemas.
29     Si numero MOD divisor==0 Entonces
30         divisores ← divisores+ConvertirATexto(divisor)+', ';
31     FinSi
32 FinPara
33 // Por ultimo se muestran los numeros que cumplen con la condición
34 Escribir 'Los divisores de ', numero, ' son:', divisores;
35 Escribir '-----';
36 Escribir 'Se generaron los divisores del numero: ', numero;
37 Escribir '          (hecho por: René Orozco)';
38 FinProceso

```

Figura 11: Captura de pantalla donde se muestra el pseudocódigo desarrollado.



```

PSeInt - Ejecutando proceso DIVISOREUNNATURAL
*** Ejecución Iniciada. ***
Ingrese un número natural (entero):
> 10
Los divisores de 10 son: 1, 2, 5, 10,
-----
Se generaron los divisores del numero: 10
          (hecho por: René Orozco)
*** Ejecución Finalizada. ***

```

Figura 12: Captura de pantalla donde se muestra la salida de ejecución del pseudocódigo.

Quinto algoritmo

Ejercicio 5

Diseña un algoritmo que valide si un numero natural es primo.

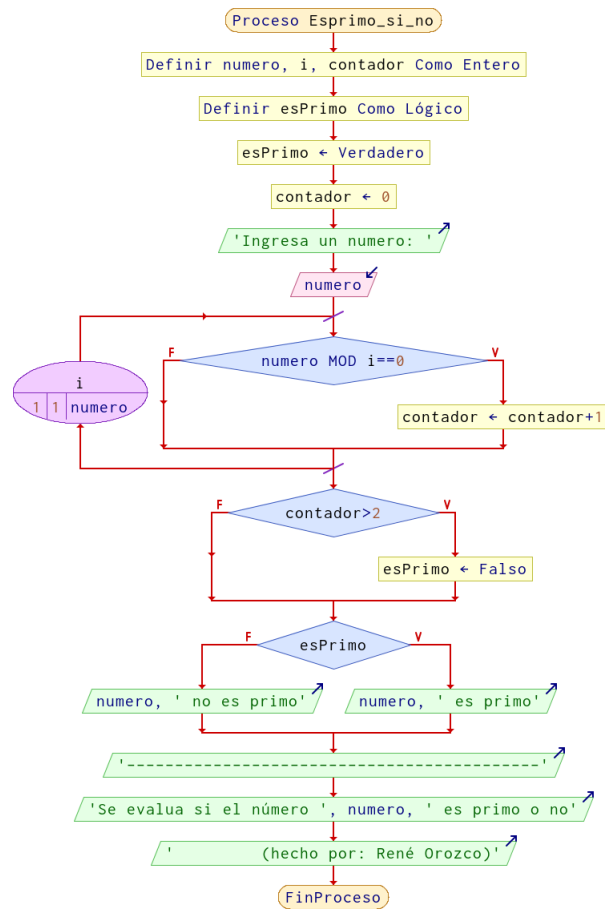


Figura 13: Diagrama del algoritmo para validar si un numero es primo o no. Desarrollado en pseint

```

1  Proceso EsPrimo_si_no
2  // Declaramos tres variables de tipo entero
3  // una con nombre 'numero', otra como 'i' y
4  // la tercera como contador
5  Definir numero, i, contador Como Entero;
6  // Declaramos una variable de tipo booleano
7  // con nombre 'esPrimo'
8  Definir esPrimo Como Lógico;
9  // Inicializamos la variable 'esPrimo' con un valor 'True'
10 esPrimo ← Verdadero;
11 // Inicializamos la variable 'contador' con el valor cero
12 contador ← 0;
13 // Le solicitamos al usuario que ingrese un numero
14 Escribir 'Ingresa un numero: ';
15 // Con el comando 'Leer' almacenamos el numero que ingreso
16 // el usuario y lo asociamos a la variable 'numero'
17 Leer numero;
18 // Iniciamos un ciclo donde recorreremos desde el valor de 'i'
19 // el cual es uno hasta el valor ingresado por el usuario contador
20 // un paso de una unidad y en cada iteración se llevara acabo una acción
21 Para i←1 Hasta numero Con Paso 1 Hacer
22     // Lo que se llevara acabo es que se calculara el modulo
23     // del numero que ingreso el usuario y el valor del contador 'i'
24     // si el resultado es igual a cero entonces a la variable contador
25     // se le agregara una unidad.
26     Si numero MOD i==0 Entonces
27         contador ← contador+1;
28     FinSi
29 FinPara
30 // Despues se evalua la variable 'contador' de la siguiente manera
31 // si la variable contador es mayor a dos entonces la variable
32 // 'esPrimo' cambiara de valor 'True' a 'False'
33 Si contador>2 Entonces
34     esPrimo ← Falso;
35 FinSi
36 // Ya por ultimo se muestra al usuario si el numero que ingreso es o no

```

Figura 14: Captura de pantalla donde se muestra el pseudocódigo.

```

PSeInt - Ejecutando proceso ESPRIMO_SI_NO
*** Ejecución Iniciada. ***
Ingresa un numero:
> 9
9 no es primo
-----
Se evalua si el número 9 es primo o no
    (hecho por: René Orozco)
*** Ejecución Finalizada. ***

☐ No cerrar esta ventana ☐ Siempre visible 

```

Figura 15: Captura de pantalla donde se muestra la salida de ejecución del pseudocódigo.

Sexto algoritmo

Ejercicio 6

Diseña un algoritmo que represente un numero natural en sus factores primos.

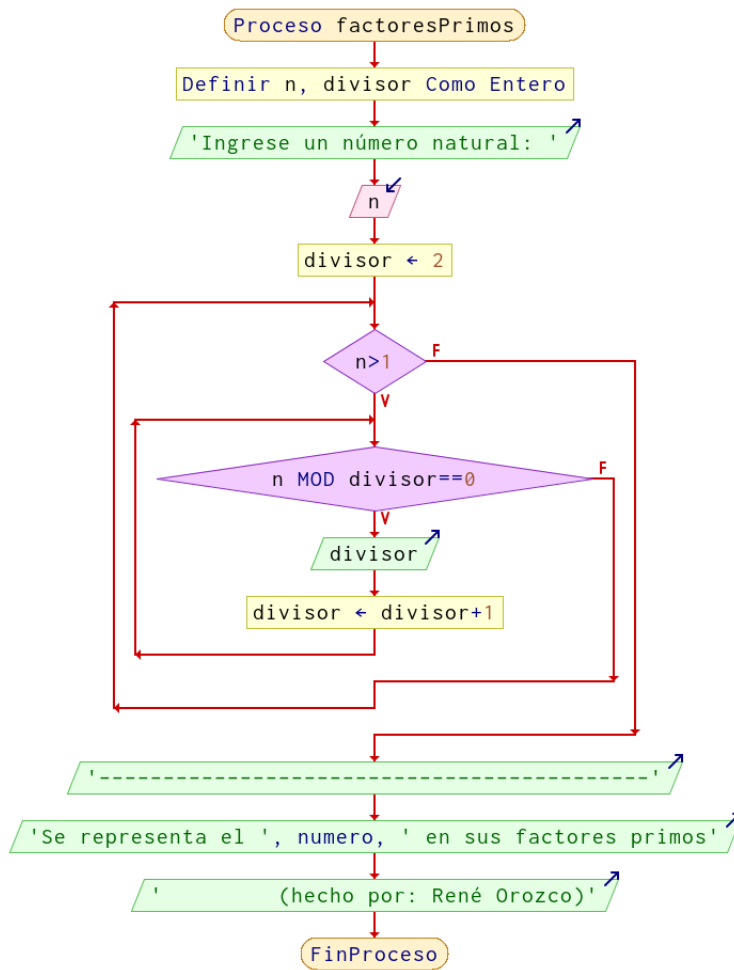


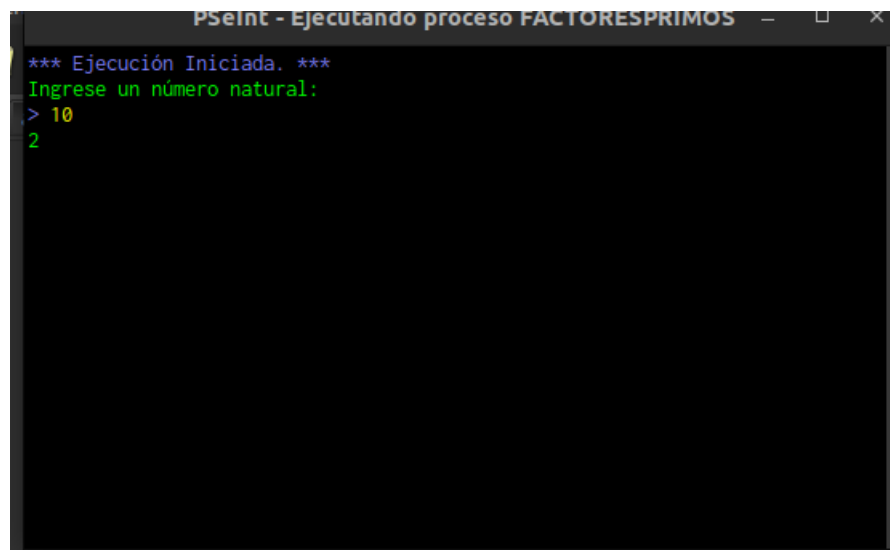
Figura 16: Diagrama del algoritmo para imprimir los factores primos de un numero natural. Desarrollado en pseint

```

1  Proceso factoresPrimos
2      // Declaramos dos variables de tipo entero
3      Definir n, divisor Como Entero;
4
5      // // Le solicitamos al usuario que ingrese un numero
6      Escribir "Ingrese un número natural: ";
7      // Con el comando 'Leer' almacenamos el numero que ingreso
8      // el usuario y lo asociamos a la variable 'n'
9      Leer n;
10
11     // inicializamos la variable 'divisor' con el valor de 2
12     divisor ← 2;
13
14     // Mientras el valor asociado a la variable 'n' sea mayor a 1
15     // se llevara acabo el calculo del modulo entre la variable 'n'
16     // y la variable 'divisor' si el resultado es cero entonces se
17     // el valor de divisor, para despues sumarle una unidad
18     Mientras n > 1 Hacer
19         Mientras n % divisor == 0 Hacer
20             Escribir divisor;
21             divisor ← divisor +1;
22         FinMientras
23     FinMientras
24     Escribir "-----";
25     Escribir "Se representa el ", numero, " en sus factores primos";
26     Escribir "          (hecho por: René Orozco)";
27 FinProceso
28

```

Figura 17: Captura de pantalla donde se muestra el pseudocódigo.



```

PSeInt - Ejecutando proceso FACTORESPRIMOS
*** Ejecución Iniciada. ***
Ingrese un número natural:
> 10
2

```

Figura 18: Captura de pantalla donde se muestra la salida de ejecución del pseudocódigo.