



## Actividad | 1 | Algoritmos

### Introducción al Desarrollo de Software

---

Ingeniería en Desarrollo de Software



TUTOR:

ALUMNO: Alejandro Norzagaray Martínez

FECHA: 28/03/2025

## ÍNDICE

Introducción .....	3
Descripción .....	4
Justificación .....	5
Desarrollo .....	6
Conclusión .....	15
Referencias.....	16

## Introducción

Entendemos por pseudocódigo que está compuesto por diversas expresiones, las cuales son matemáticas, lógica y variables. También conocido como lenguaje falso.

Las principales características del pseudocódigo son

- Representar algoritmos
- Pueden ser ejecutados en una computadora
- Siguen un diseño descendente (de arriba hacia abajo)

También así está presente el algoritmo que es una serie finita de instrucciones dadas para resolver un problema, efectuar un cálculo o realizar una actividad.

El algoritmo se representa gráficamente mediante un diagrama de flujo. Donde su objetivo principal es el resolver o dar una solución a un problema de manera clara, que se pueda desarrollar en cualquier lenguaje de programación.

Sus principales características son:

- Es finito (con inicio y final)
- Tiene una secuencia
- Es preciso
- Tiene lógica

Constando así de 3 partes un algoritmo:

- Input (entrada)
- Proceso
- Output(salida)

Todo algoritmo debe seguir estos tres pasos en el orden antes mencionado.

Por otra parte, los diagramas de flujos son la representación gráfica de un algoritmo. Simbolizando un proceso con el apoyo de diferentes figuras específicas, la cuáles son sus partes fundamentales.

Su principal objetivo es ordenar, de la mejor forma, los pasos a seguir, del algoritmo.

Las características principales del diagrama de flujo son:

- Se compone de bloque de información con flechgas que indican el flujo del proceso
- Tienen un inicio y un fin
- Se simbolizan para su mejor entendimiento
- Permiten leer un proceso extenso de manera rápida
- Sirve como herramienta de control para organizar información

El lenguaje C es un lenguaje de programación que se utiliza para desarrollar sistemas operativos y aplicaciones. Es un lenguaje de bajo nivel que se considera uno de los mas importantes de la actualidad.

## Descripción

A lo largo de esta actividad comprenderemos como se realiza un algoritmo, su funcionamiento y su estructura, así también como es que esto facilita la tarea o resuelve un problema, veremos como se desarrolla el diagrama de flujo de cada uno de los problemas que tendremos presente como podrían ser números primos, números par e impar y los números invertidos o al revés. Son algunas de los ejemplos que veremos a continuación.

También se verá cómo es que estructura de un algoritmo, a un diagrama de flujo y al final llega a los que se conoce como código en lenguaje C.

Como es que las palabras cambian para que el código pueda funcionar sin problemas y las principales características de cada una de los apartados para que todo pueda funcionar de acuerdo a las características que pide cada uno de ellos explicadas a detalle para que no haya ningún problema cuando se requiera ejecutarlos.

## **Justificación**

Entender la importancia de como funciona cada una de estas herramientas ayuda mucho al usuario para poder desarrollar, comprender, resolver los diferentes uso que se le pueden dar en lo que seria la programación, desde lo mas básico que seria un algoritmo que funciona como un borrador, donde plasmas la idea a generar, seguido de un diagrama de flujo donde todo ya tiene un orden, una estructura cual seguir para poder por ultimo buscar el lenguaje que mas te convenga y realizarlo en el lenguaje correspondiente.

Al saber utilizar correctamente un algoritmo podrás generas con mas facilidad lo que serian las ideas para un proyecto sin complicar el proceso.

Con un buen manejo de diagrama de flujo podrías facilitar a las personas que entiendan tus ideas ya si puedan conectar para un futuro proyecto.

Con un buen manejo del un lenguaje de programación podrás crear todo eso que alguna vez ideaste como un algoritmo en un código que funcione correctamente y facilite el trabajo el programa diseñado.

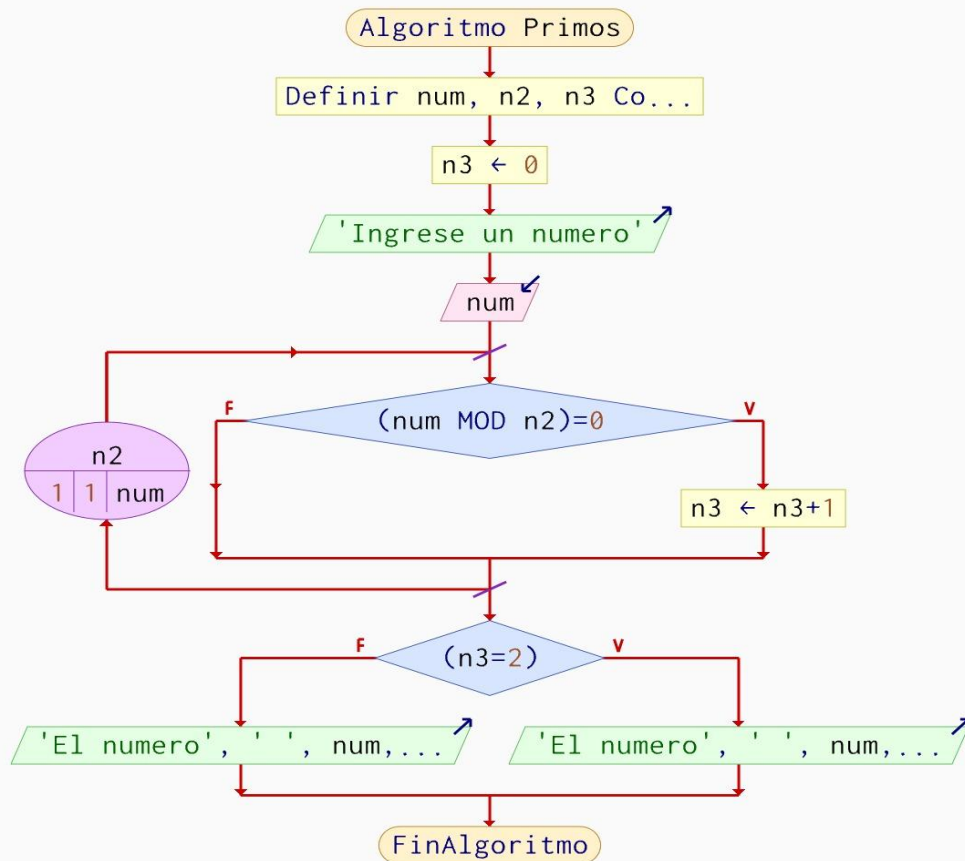
## Desarrollo

```
Algoritmo Primos
  Definir num, n2,n3 Como Entero;
  n3<-0;
  Escribir "Ingrese un numero";
  leer num;

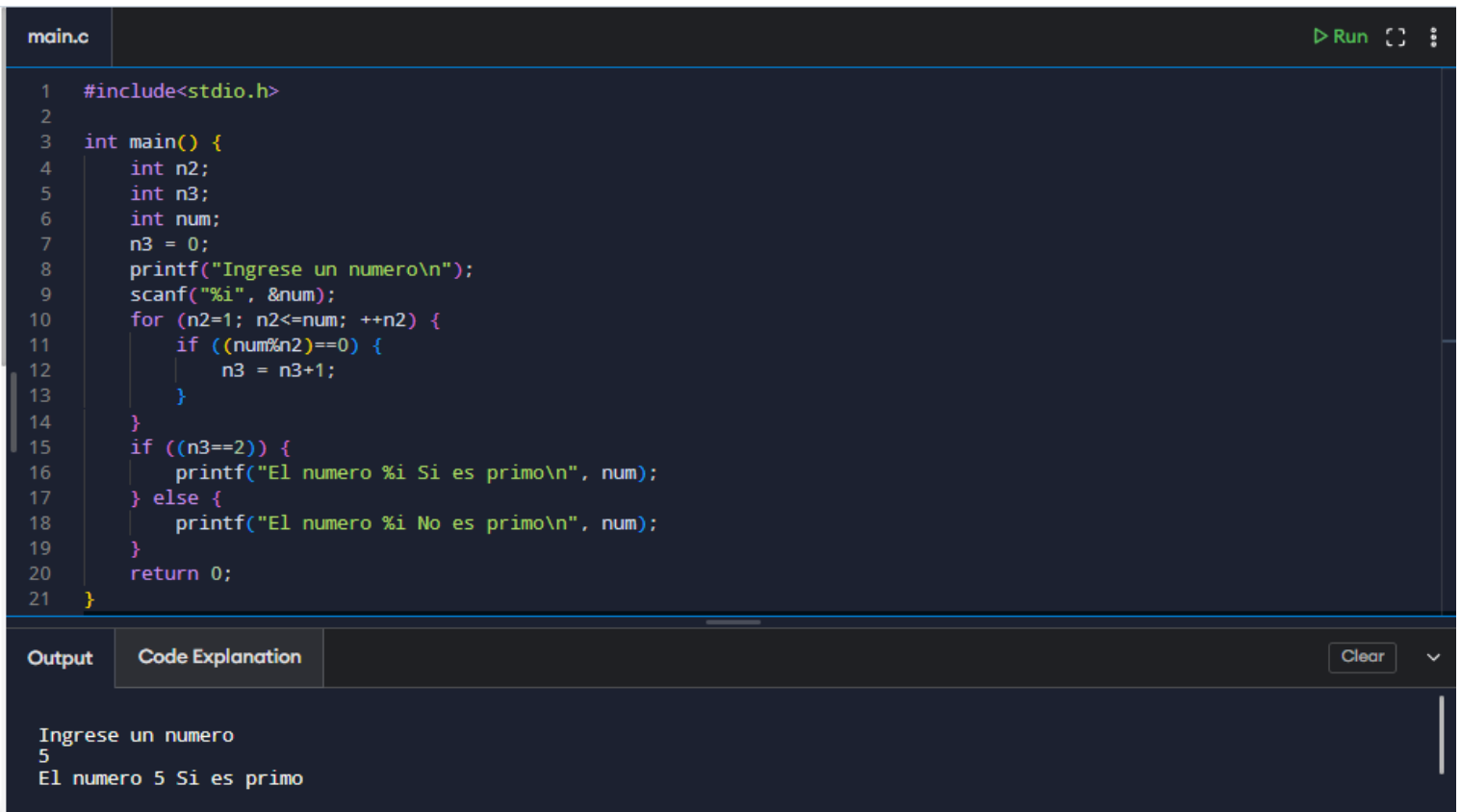
  Para n2<-1 Hasta num Con Paso 1 Hacer
    si (num mod n2)= 0 Entonces
      n3<-n3+1
    FinSi

  FinPara
  si (n3=2) Entonces
    Escribir "El numero" " " num " " "" "Si es primo"
  sino
    Escribir "El numero" " " num " " "No es primo"
  FinSi
FinAlgoritmo
```

Como podemos apreciar en la anterior figura vemos el ejemplo de como seria un algoritmo, es simplemente una input ( entrada), un proceso que en este caso es para saber si un numero es primo o no es primo y una output (salida).



En esta figura podemos apreciar como es que se desglosa un diagrama de flujo que vendría siendo la segunda parte de nuestro proceso, como podemos observar, tiene diferentes formas conectadas con líneas, cada una de estas formas representa una función. Tanto así también podemos observar que es igual al algoritmo, mantienen la misma función sin embargo esta tiene una peculiar forma que lo puede hacer más entendible a simple vista ya que cuenta con sus respectivas formas geométricas y sus líneas que señalan la dirección a la que dirigen si las condiciones que se están pidiendo se cumplen tanto así como se muestra el comienzo y el final del algoritmo de una forma más sencilla de entender.



The image shows a code editor window with a dark theme. The title bar at the top left says 'main.c' and the top right has a green 'Run' button and some icons. The code is written in C and is a program to check if a number is prime. It includes the `<stdio.h>` header, declares variables `n2`, `n3`, and `num`, and initializes `n3` to 0. It prompts the user to enter a number, reads it into `num`, and then uses a for loop to check for divisors from 1 to `num`. If a divisor is found, `n3` is incremented. After the loop, it checks if `n3` is 2 (indicating a prime) and prints the result. The output section at the bottom shows the program's execution: it prompts 'Ingrese un numero', the user enters '5', and the program outputs 'El numero 5 Si es primo'.

```
1  #include<stdio.h>
2
3  int main() {
4      int n2;
5      int n3;
6      int num;
7      n3 = 0;
8      printf("Ingrese un numero\n");
9      scanf("%i", &num);
10     for (n2=1; n2<=num; ++n2) {
11         if ((num%n2)==0) {
12             n3 = n3+1;
13         }
14     }
15     if ((n3==2)) {
16         printf("El numero %i Si es primo\n", num);
17     } else {
18         printf("El numero %i No es primo\n", num);
19     }
20     return 0;
21 }
```

**Output** | **Code Explanation** | Clear ▼

```
Ingrese un numero
5
El numero 5 Si es primo
```

En esta figura apreciamos como ya pasamos de un algoritmo realizado en un bloque de notas, a un algoritmo con sus respectivas formas, a lo que ahora viene siento un código en esta ocasión lenguaje C. A simple visto se aprecia que ahora se usan palabras distintas eso se debe a que cada lenguaje tiene su distinta forma de utilizar las palabras para diferentes tipos de acción. Es importante mencionar que antes de empezar cualquier código que seria en lenguaje C es necesario incluid el comando `#include <stdio.h>`, ya que sin ese comando no va funcionar nuestro código.

Es importante saber que en algunos casos los códigos comparten semejanzas en algunas características de signos como vendrían siendo los paréntesis `()`, las comillas `""`, entre otros que esta tiene el mismo funcionamiento.



Algoritmo Par\_Impar

Definir num, n2 Como Real

Escribir "Ingrese un numero"

Leer num;

n2←num mod 2;

si n2=0 Entonces;

Escribir num " " "Es par";

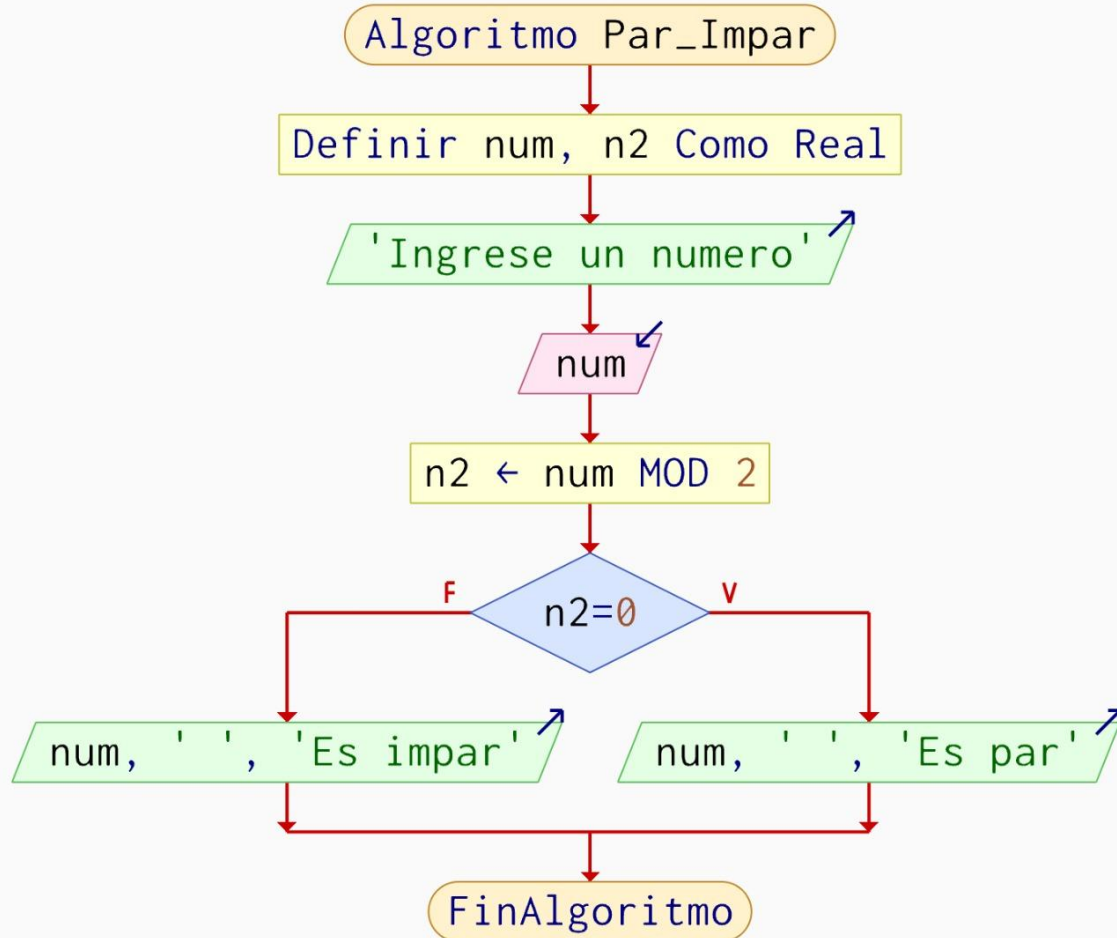
SiNo;

Escribir num " " "Es impar";

FinSi

FinAlgoritmo

En este segundo ejemplo podemos observar como que es un algoritmo parecido pero es totalmente diferente ya que en esta ocasión estamos viendo lo que sería un numero par o impar. Pero cuentan con la misma características principales input (entrada), proceso y output (salida) que seria lo que lo caracteriza como un algoritmo.



Podemos apreciar con mas detalles como es que este algoritmo esta estructurado de una forma más sencilla, como es su funcionamiento, aun que ya mencionado es parecido en estas ocasiones podemos observar que solo hay una condición si es verdadero o falso, cada uno lleva a una respectiva respuesta que el usuario quiere conocer que seria si el numero que ingreso es par o es impar. Se sigue viendo como tiene su entrada, proceso y salida.

main.c	
1	<code>#include&lt;stdio.h&gt;</code>
2	
3	<code>int main() {</code>
4	<code>    int n2;</code>
5	<code>    int num;</code>
6	<code>    printf("Ingrese un numero\n");</code>
7	<code>    scanf("%i", &amp;num);</code>
8	<code>    n2 = (int) num%2;</code>
9	<code>    if (n2==0) {</code>
10	<code>        printf("%i Es par\n", num);</code>
11	<code>    } else {</code>
12	<code>        printf("%i Es impar\n", num);</code>
13	<code>    }</code>
14	<code>    return 0;</code>
15	<code>}</code>

Output	Code Explanation
4	
4 Es par	

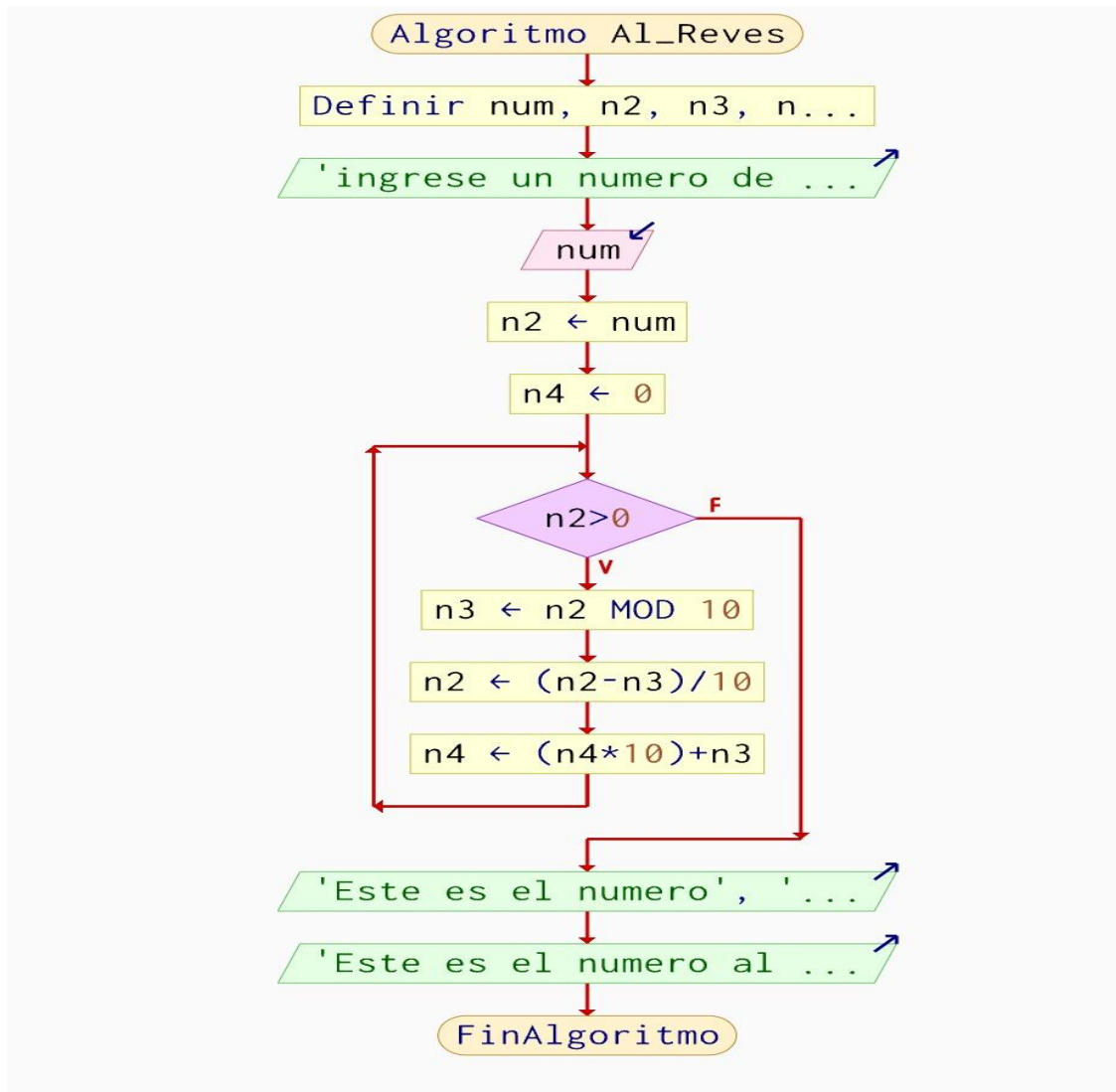
En esta figura podemos observar que nuestro algoritmo , ya antes mostrado en diagrama de flujo, ahora ya esta modificado de palabras para poder verlo funcionar en lo que seria la consola, en este caso lenguaje C.

### Algoritmo Al\_Reves

```
Definir num, n2, n3, n4 Como Entero;  
Escribir "ingrese un numero de 4 dígitos";  
Leer num;  
n2<-num;  
n4<-0;  
  
Mientras n2>0 Hacer  
    n3<-n2 mod 10;  
    n2<-(n2-n3)/10;  
    n4<-(n4*10)+n3;  
FinMientras  
Escribir "Este es el numero", " " num;  
Escribir "Este es el numero al revés" " " n4;
```

### FinAlgoritmo

En la figura ahora mostrada podemos observar como el algoritmo volvió a cambiar, esta vez para voltear o poner al revés el orden de los números esto se consigue haciendo que el usuario nos brinde un valor y nosotros lo que hacemos es que con diferentes variables le damos otro valor al que nos brindó principalmente el usuario y así podremos lograr el resultado requerido por el usuario.



Podemos observar en esta figura que es un poco las larga en tamaño que las anteriores ya que cuenta un pequeño ciclo para poder realizar los cambios necesario para brindarle lo que pide al usuario, dando al final sus número y el numero al revés que solicito.

main.c	
1	<code>#include&lt;stdio.h&gt;</code>
2	
3	<code>int main() {</code>
4	<code>    int n2;</code>
5	<code>    int n3;</code>
6	<code>    int n4;</code>
7	<code>    int num;</code>
8	<code>    printf("ingrese un numero de 4 digitos\n");</code>
9	<code>    scanf("%i", &amp;num);</code>
10	<code>    n2 = num;</code>
11	<code>    n4 = 0;</code>
12	<code>    while (n2&gt;0) {</code>
13	<code>        n3 = n2%10;</code>
14	<code>        n2 = (n2-n3)/10;</code>
15	<code>        n4 = (n4*10)+n3;</code>
16	<code>    }</code>
17	<code>    printf("Este es el numero %i\n", num);</code>
18	<code>    printf("Este es el numero al revés %i\n", n4);</code>
19	<code>    return 0;</code>
20	<code>}</code>

Output	Code Explanation
<code>ingrese un numero de 4 digitos</code>	
<code>4567</code>	
<code>Este es el numero 4567</code>	
<code>Este es el numero al revés 7654</code>	

Como se puede observar en la figura que nuestro algoritmo sigue dando la funcion que se requiere en nuestra consola ahora ya como un codigo en lenguaje C.

## Conclusión

En esta unidad aprendimos los conceptos más básicos y primordiales para crear un algoritmo, un diagrama de flujo y a simple vista lo que es un código en lenguaje C. Observamos los diferentes tipos de ejemplos con los cuales se pueden hacer diferentes operaciones con y como cambia cada algoritmo, diagrama de flujo y código dependiendo lo que se pida por parte del usuario, que cada lenguaje es diferente su forma de caracteres, como nos ayuda un diagrama de flujo a entender de una forma mas sencilla un algoritmo con sus figuras geométricas que representa cada cosa que se hace en el algoritmo y sus respectivas líneas que indican el flujo del algoritmo.

Pudimos observar que al momento de iniciar un código cada uno tiene una forma diferente de comenzar para que este funcione en este caso vimos que lenguaje C se necesita `#include<stdio.h>`.

Pudimos observar que cada una de las herramientas tiene sus características para poder elaborarlas de forma correcta y así poder resolver los problemas de forma más sencilla, también aprendimos a desarrollar cada una de las herramientas usadas.

## Referencias

Félix, H. “*Intérprete para probar un programa escrito en pseudocódigo.*” Universidad Mayor de San Marcos: Perú. ISSN 1560-9146, ISSN-e 1810-9993, Vol. 17, Nº. 1, 2014, págs. 101-110. Recuperado de: <https://dialnet.unirioja.es/servlet/articulo?codigo=8635435>

Sierra, J. (2008). “*Enciclopedia del lenguaje C++. 2ª edición.*” Grupo Editorial RA-MA: España. ISBN: 8499643574, 9788499643571. Obtenido de: [https://books.google.es/books?id=EexFDwAAQBAJ&dq=lenguaje+c%2B%2B&lr=&hl=es&source=gbs\\_navlinks\\_s](https://books.google.es/books?id=EexFDwAAQBAJ&dq=lenguaje+c%2B%2B&lr=&hl=es&source=gbs_navlinks_s)

Del Prado, A. (2014) “*Alternativas para la enseñanza de pseudocódigo y diagrama de flujo.*” Revista Electrónica Iberoamericana de Educación en Ciencias y Tecnología: Argentina. Vol. 5, núm. 3, págs. 102-113. Recuperado de: <https://exactas.unca.edu.ar/riecyt/VOL%205%20NUM%203/F%20%20SI%203%2014%20Trabajo%20Completo%20Fundamentos.pdf>

Lorente, K., (2020). “*Aprendizaje de estructuras de datos mediante algoritmos usando pseudocódigos.*” Revista Electrónica Formación y Calidad Educativa. Vol. 8, núm. 1, págs. 91-102.

Castillo, A. (2014) “*Didactics Procedures System for improving Computational Algorithmization.*” Conferencia Científica Internacional UCIENCIA: Cuba. Obtenido de: [https://www.researchgate.net/profile/Alexander-Gorina-Sanchez/publication/319711014\\_Sistema\\_de\\_Procedimientos\\_Didacticos\\_para\\_perfeccionar\\_la\\_Algoritmizacion\\_Computacional\\_Didactics\\_Procedures\\_System\\_for\\_improving\\_Computational\\_Algorithmization/links/59ba8cea458515bb9c4cb68e/Sistema-de-Procedimientos-Didacticos-para-perfeccionar-la-Algoritmizacion-Computacional-Didactics-Procedures-System-for-improving-Computational-Algorithmization.pdf](https://www.researchgate.net/profile/Alexander-Gorina-Sanchez/publication/319711014_Sistema_de_Procedimientos_Didacticos_para_perfeccionar_la_Algoritmizacion_Computacional_Didactics_Procedures_System_for_improving_Computational_Algorithmization/links/59ba8cea458515bb9c4cb68e/Sistema-de-Procedimientos-Didacticos-para-perfeccionar-la-Algoritmizacion-Computacional-Didactics-Procedures-System-for-improving-Computational-Algorithmization.pdf)

Guevara, C., Párraga, J. (2017). “*DESARROLLO DE UNA HERRAMIENTA PARA EL DISEÑO Y EJECUCIÓN DE DIAGRAMAS DE FLUJO ENFOCADOS A ALGORITMOS COMPUTACIONALES EN PLATAFORMAS MÓVILES. (Tesis pregrado).*” UNIVERSIDAD LAICA ELOY ALFARO DE MANABÍ: Ecuador. Obtenido de: <http://repositorio.uleam.edu.ec/handle/123456789/63>

Saieef, S. (2022) “*EveCode: An Integrated Web Application for Coding Skill Development with an In-Built Compiler and Collaborative Learning Features.*” UNIVERSITY OF LIBERAL ARTS BANGLADESH: Bangladés. Obtenido de: [https://www.researchgate.net/profile/Saieef-Sunny/publication/386214706\\_EveCode\\_An\\_Integrated\\_Web\\_Application\\_for\\_Coding\\_Skill\\_Development\\_with\\_an\\_In-Built\\_Compiler\\_and\\_Collaborative\\_Learning\\_Features/links/6748c121359dcb4d9d3ef360/EveCode-An-Integrated-Web-Application-for-Coding-Skill-Development-with-an-In-Built-Compiler-and-Collaborative-Learning-Features.pdf](https://www.researchgate.net/profile/Saieef-Sunny/publication/386214706_EveCode_An_Integrated_Web_Application_for_Coding_Skill_Development_with_an_In-Built_Compiler_and_Collaborative_Learning_Features/links/6748c121359dcb4d9d3ef360/EveCode-An-Integrated-Web-Application-for-Coding-Skill-Development-with-an-In-Built-Compiler-and-Collaborative-Learning-Features.pdf)