

Actividad | 1 | Algoritmos

Introducción al Desarrollo de Software

Ingeniería en Desarrollo de Software



TUTOR: Sandra Luz Lara Dévora

ALUMNO: Cynthia Jeanette García Torres

FECHA: Sábado, 9 de Agosto de 2025

Índice

Introducción.....	3
Descripción	3
Justificación.....	4
Desarrollo	5
Números primos.....	5
Número par e impar	6
Números invertidos	8
Conclusión	10
Referencias	10

Introducción

En esta actividad presento el desarrollo de tres algoritmos que me han permitido poner en práctica y reforzar conceptos clave de la programación estructurada: identificar si un número es primo, determinar si un número es par o impar, e invertir el orden de las cifras de un número. Para ello utilicé el programa PSeInt, una herramienta que facilita la escritura de pseudocódigo y la comprensión de la lógica de los procesos antes de llevarlos a un lenguaje de programación formal.

La finalidad de este trabajo es no solo cumplir con la implementación correcta de cada algoritmo, sino también comprender de manera clara la lógica que hay detrás de ellos. A lo largo del documento compartiré el pseudocódigo, una breve explicación de su funcionamiento y ejemplos de ejecución obtenidos durante las pruebas. Este ejercicio ha sido una oportunidad para afianzar mis conocimientos, desarrollar un pensamiento lógico más preciso y familiarizarme con estructuras como ciclos y condicionales, indispensables en el mundo de la programación.

Descripción

Esta actividad representa una oportunidad para poner en práctica, de forma estructurada y consciente, los conocimientos adquiridos en la creación de algoritmos. El trabajo se desarrolló en el entorno de PSeInt, el cual permite plasmar de manera clara y ordenada la lógica de un programa antes de implementarlo en un lenguaje formal. El

objetivo fue resolver tres problemas específicos: determinar si un número es primo, identificar si es par o impar, e invertir el orden de sus cifras.

Cada uno de estos ejercicios requirió analizar el problema, comprender su lógica y traducirla en una serie de pasos que la computadora pueda interpretar. Para ello, fue fundamental utilizar estructuras de control como condicionales y bucles, que permiten evaluar condiciones y repetir instrucciones de forma controlada. Personalmente, esta práctica me permitió reforzar mi pensamiento lógico, mejorar mi atención a los detalles y afianzar la capacidad de diseñar soluciones eficientes. Además, me brindó la satisfacción de comprobar que la lógica planteada funcionaba correctamente en cada prueba realizada.

Justificación

El uso de algoritmos desarrollados en pseudocódigo, como los que se presentan en esta actividad, constituye una estrategia efectiva para abordar problemas de programación, especialmente en etapas de aprendizaje. Trabajar con PSeInt permite centrarse en la lógica y la estructura de la solución sin distracciones propias de la sintaxis de un lenguaje formal. Esto facilita comprender de manera clara el flujo de un programa, identificar posibles errores y corregirlos antes de la implementación final.

En el caso de los tres problemas planteados, la solución basada en pseudocódigo ofrece un enfoque ordenado que guía paso a paso el proceso. Este método no solo garantiza que la lógica sea coherente, sino que también optimiza el tiempo de desarrollo al prevenir errores posteriores. Personalmente, considero que esta forma de trabajo

fortalece la capacidad de análisis y fomenta un pensamiento lógico estructurado, habilidades esenciales para cualquier programador que aspire a crear soluciones claras, eficientes y fáciles de mantener.

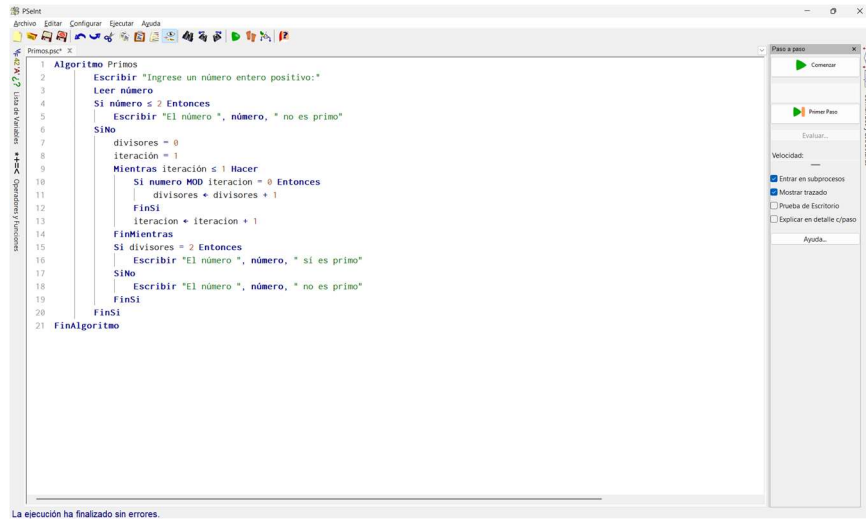
Desarrollo

En el siguiente apartado se presentan los algoritmos elaborados para dar solución a cada uno de los problemas planteados. Cada uno incluye su pseudocódigo escrito en PSeInt, la explicación de la lógica utilizada y una captura de su ejecución.

Números primos

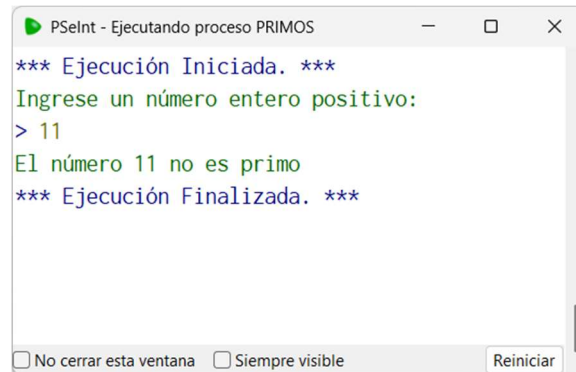
El algoritmo solicita al usuario un número entero positivo. Luego, recorre todos los valores desde 1 hasta el número ingresado y cuenta cuántos divisores exactos tiene. Si el total de divisores es exactamente 2, se considera primo (divisible solo por 1 y por sí mismo). En caso contrario, se determina que no lo es.

Figura 1. *Pseudocódigo del algoritmo de números primos escrito en PSeInt.*



Nota. En el código se observa el uso de estructuras condicionales y un ciclo Mientras para evaluar divisores y determinar si el número es primo o no.

Figura 2. Ejecución del algoritmo de números primos.



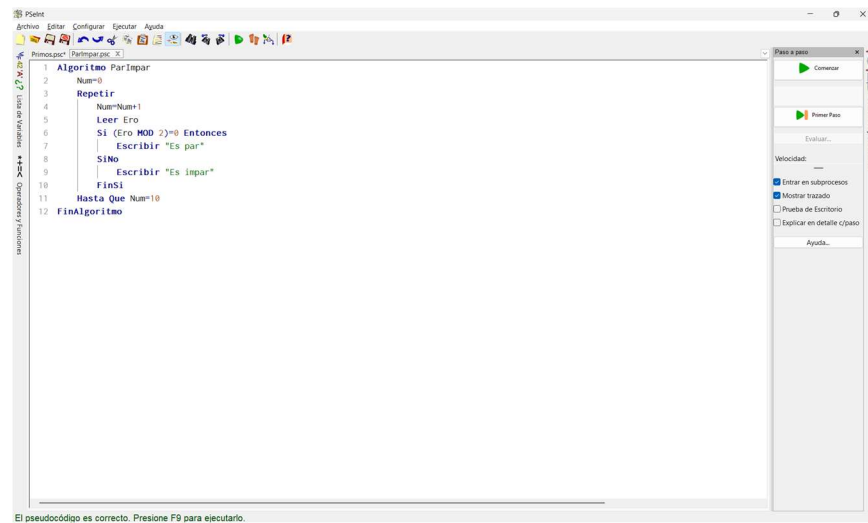
Nota. La ejecución muestra que el número 11 no es primo según la lógica implementada.

Número par e impar

Este algoritmo solicita al usuario diez números enteros, uno por cada iteración del ciclo. Por cada número ingresado, se aplica la operación módulo (MOD 2) para determinar si es divisible entre 2. Si el residuo es igual a cero, el número es clasificado como par; de

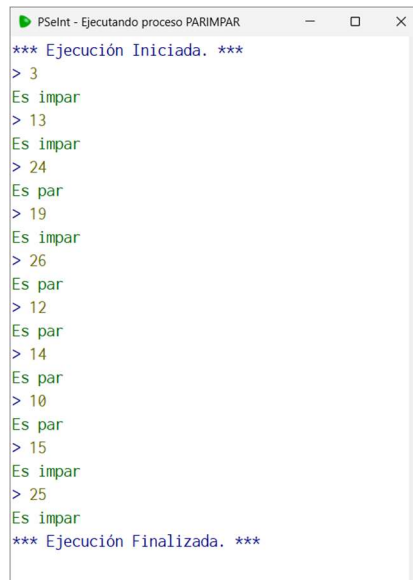
lo contrario, se clasifica como impar. El proceso se repite hasta completar el número de intentos establecidos, utilizando la estructura Repetir...Hasta Que.

Figura 3. Pseudocódigo del algoritmo para clasificar un número como par o impar en PSeInt.



Nota. El código utiliza una estructura repetitiva para solicitar diez números y evaluar cada uno mediante el operador módulo.

Figura 4. Ejecución del algoritmo de par o impar con diferentes entradas.



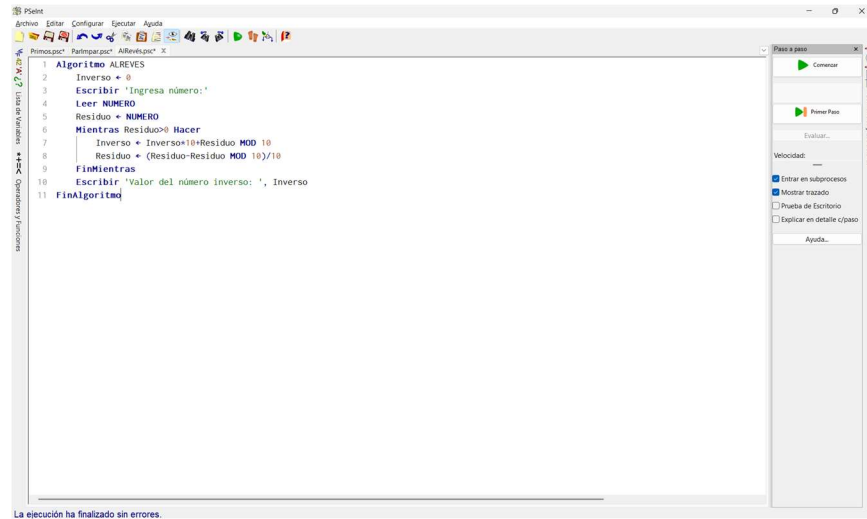
```
PSeInt - Ejecutando proceso PARIMPAR
*** Ejecución Iniciada. ***
> 3
Es impar
> 13
Es impar
> 24
Es par
> 19
Es impar
> 26
Es par
> 12
Es par
> 14
Es par
> 10
Es par
> 15
Es impar
> 25
Es impar
*** Ejecución Finalizada. ***
```

Nota. Se muestra en pantalla la clasificación de cada número ingresado, alternando entre los resultados “Es par” y “Es impar” según corresponda.

Números invertidos

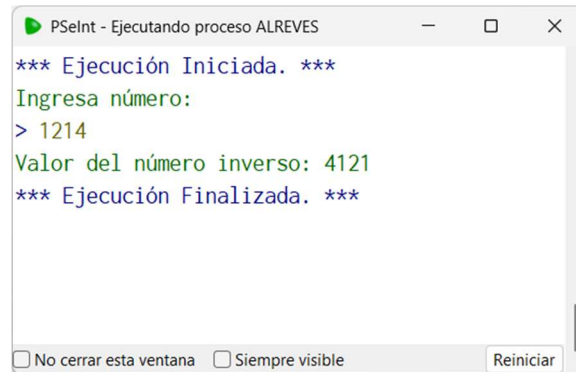
Este algoritmo solicita al usuario un número entero y calcula su valor invertido, es decir, coloca las cifras en orden contrario. Para lograrlo, emplea un ciclo que obtiene el último dígito del número usando la operación módulo (MOD 10) y lo agrega al acumulador multiplicado previamente por 10. Luego, reduce el número original mediante división entera hasta que este se vuelva cero. Al finalizar, muestra el resultado invertido en pantalla.

Figura 5. Pseudocódigo del algoritmo de inversión de números en PSeInt.



Nota. En el código se observa el uso de un ciclo Mientras para extraer y reposicionar dígitos hasta invertir por completo el número original.

Figura 6. Ejecución del algoritmo de números invertidos con el número 1214.



Nota. El resultado mostrado es 4121, que corresponde al número ingresado con sus cifras en orden inverso.

Conclusión

Desarrollar esta actividad me permitió no solo reforzar conceptos fundamentales de programación, sino también poner en práctica la capacidad de analizar y resolver problemas de forma estructurada. Implementar algoritmos para identificar números primos, clasificar números como pares o impares e invertir cifras me ayudó a comprender de manera más profunda cómo se aplican las estructuras condicionales, los ciclos y las operaciones matemáticas en soluciones reales.

En el contexto laboral, estas habilidades se traducen en una mayor capacidad para crear procesos eficientes, depurar errores y optimizar recursos, algo clave en áreas como el análisis de datos, la automatización o el desarrollo de software. En lo personal, trabajar con este tipo de ejercicios fomenta la paciencia, la atención a los detalles y el pensamiento lógico, cualidades útiles para enfrentar cualquier reto cotidiano. Esta experiencia reafirma que la programación no es solo escribir código, sino también cultivar una forma de pensar clara, ordenada y enfocada en encontrar soluciones efectivas.

Referencias

Gómez, P. (2023). PSeInt (Versión 20230501) [Software]. <https://pseint.sourceforge.net/>

Academia Global-MX. (2025, 17 de julio). Introducción al desarrollo de software #1 –

Pantalla compartida con vista del orador [Grabación de Zoom]. Academia Global-MX Zoom.