Universidad Corporativa Coppel | Programas Académicos

Ingeniería en Desarrollo de Software

Actividad | 2 | Diagramas de Flujo

Introducción al Desarrollo de Software

Ingeniería en Desarrollo de Software

academia global

TUTOR: Sandra Luz Lara Dévora

ALUMNO: Cynthia Jeanette García Torres

FECHA: Domingo, 10 de Agosto de 2025

Índice

Introducción

Esta actividad tiene como finalidad transformar en diagramas de flujo los algoritmos realizados previamente, aplicando una representación visual del proceso lógico que sigue cada uno. Los diagramas de flujo son herramientas muy útiles para organizar ideas y comprender el camino que sigue un programa, desde su inicio hasta su final.

En este caso, se trabajará con tres problemas: verificar si un número es primo, determinar si es par o impar, y revertir un número de cuatro cifras. Al representar gráficamente estos algoritmos, se puede observar de manera clara cómo se toman decisiones, cómo se repiten procesos y cuál es el orden de las operaciones. Esta forma de visualizar la lógica resulta especialmente útil cuando se está comenzando en el mundo de la programación, ya que ayuda a identificar errores, a depurar procesos y a entender el funcionamiento interno de los programas antes de escribir código en un lenguaje formal como C.

Descripción

La actividad consiste en crear diagramas de flujo que representen visualmente los algoritmos planteados en la actividad anterior. El primer diagrama corresponde al algoritmo de números primos, el cual evalúa si un número ingresado tiene exactamente dos divisores: uno y él mismo. El segundo diagrama representa la clasificación de un número como par o impar, función que se logra mediante el operador módulo. Finalmente, el tercer diagrama aborda la inversión de un número de cuatro cifras, separando cada dígito y reordenándolos en sentido contrario. Cada uno de estos diagramas se construye utilizando

símbolos estándar como óvalos para inicio/fin, rombos para decisiones, y rectángulos para operaciones o procesos.

Esta representación permite visualizar el camino lógico que sigue el programa, lo cual es clave para su comprensión. Además, se incluirá una breve explicación de cada diagrama, detallando la lógica aplicada a cada símbolo, con el propósito de facilitar la interpretación del proceso completo.

Justificación

Representar los algoritmos mediante diagramas de flujo ofrece una ventaja significativa al momento de planificar y desarrollar software. Estas representaciones permiten entender el proceso de forma secuencial y lógica, facilitando la identificación de errores y la mejora en la organización de los pasos.

En el contexto de esta actividad, utilizar diagramas de flujo ayuda a consolidar lo aprendido sobre los tres algoritmos propuestos, permitiendo ver de forma clara cómo se estructura cada solución. Además, este tipo de herramientas resulta útil en entornos académicos y profesionales, ya que ayudan a comunicar ideas de programación de forma simple, incluso a personas que no están familiarizadas con lenguajes de código. Por ello, esta solución es ideal en esta etapa del aprendizaje, donde el objetivo principal es dominar la lógica y el razonamiento detrás de un programa antes de escribir código formal. De esta manera, se fortalece la comprensión del funcionamiento interno de cada algoritmo.

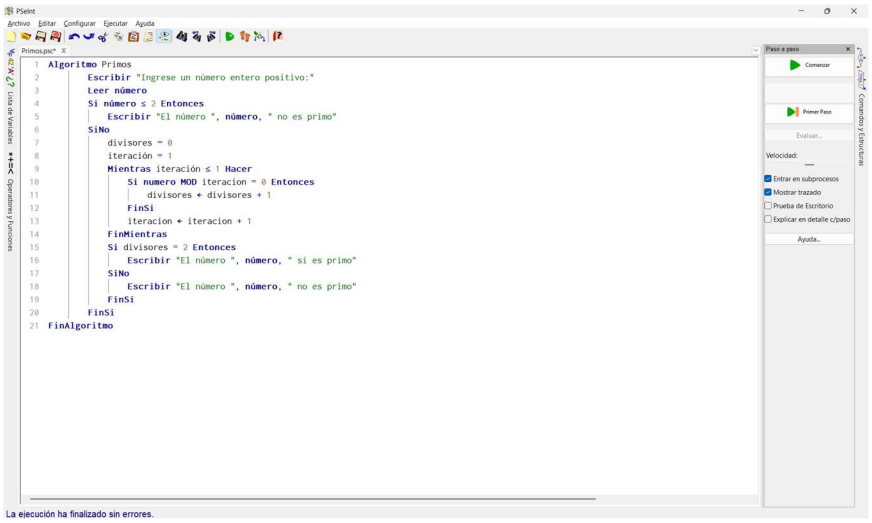
Desarrollo

En esta sección se presentan los diagramas de flujo correspondientes a los algoritmos elaborados en la actividad. Cada uno refleja de manera visual la lógica y el orden de las operaciones que permiten resolver problemas específicos, desde la verificación de números primos, la clasificación de números pares e impares, hasta la inversión de cifras en un número. Estos diagramas facilitan la comprensión de los procesos al mostrar, paso a paso, las decisiones y cálculos realizados, permitiendo identificar con claridad las condiciones, bucles y resultados esperados.

Números primos

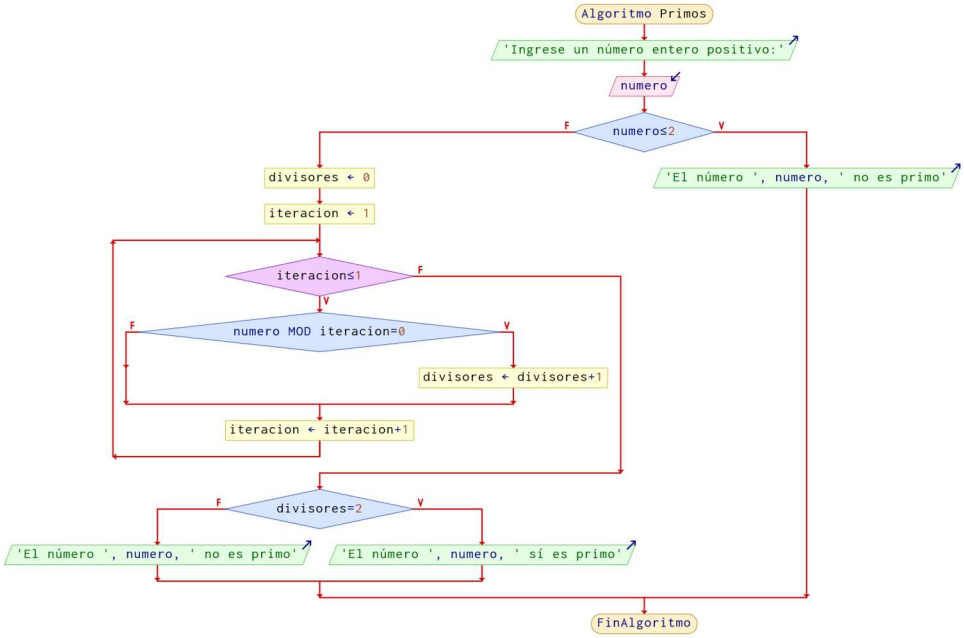
Este diagrama determina si un número entero positivo es primo. Primero verifica si el número es mayor que 2; si lo es, cuenta cuántos divisores exactos tiene al recorrer todos los enteros desde 1 hasta el propio número. Finalmente, compara el conteo con el valor esperado (dos divisores) para emitir el resultado.

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. *Diagrama de flujo del algoritmo "Primos" elaborado en PSDraw.*

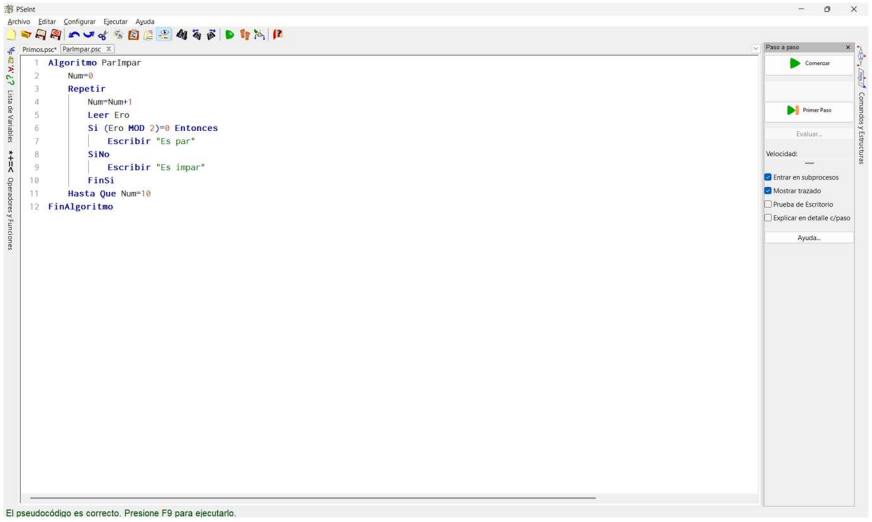


Nota. El diagrama muestra las condiciones y ciclos utilizados para identificar un número primo.

Número par e impar

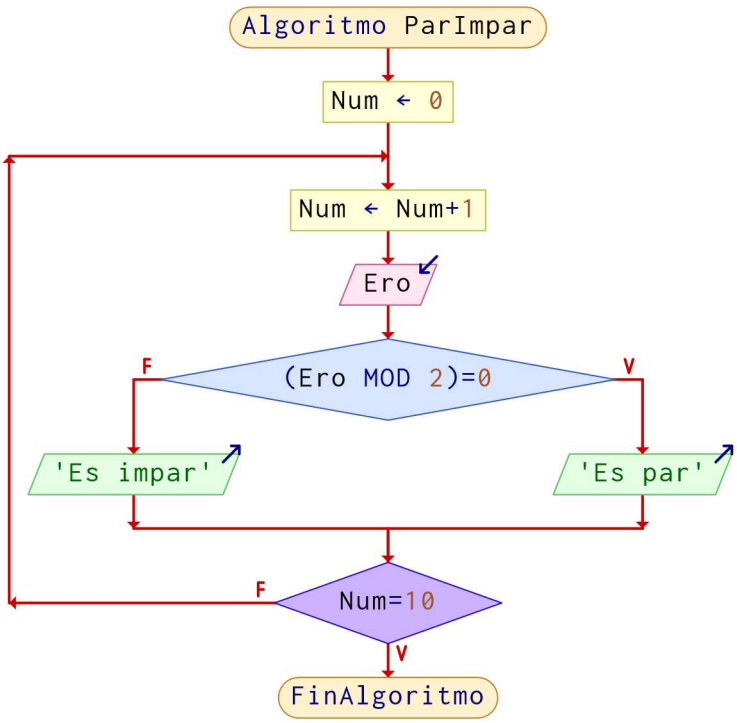
Este diagrama clasifica un conjunto de números como pares o impares. Utiliza la operación módulo para determinar si el residuo al dividir entre 2 es igual a cero (par) o diferente de cero (impar). Repite el proceso hasta alcanzar el límite establecido.

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. *Diagrama de flujo del algoritmo “ParImpar” elaborado en PSDraw.*

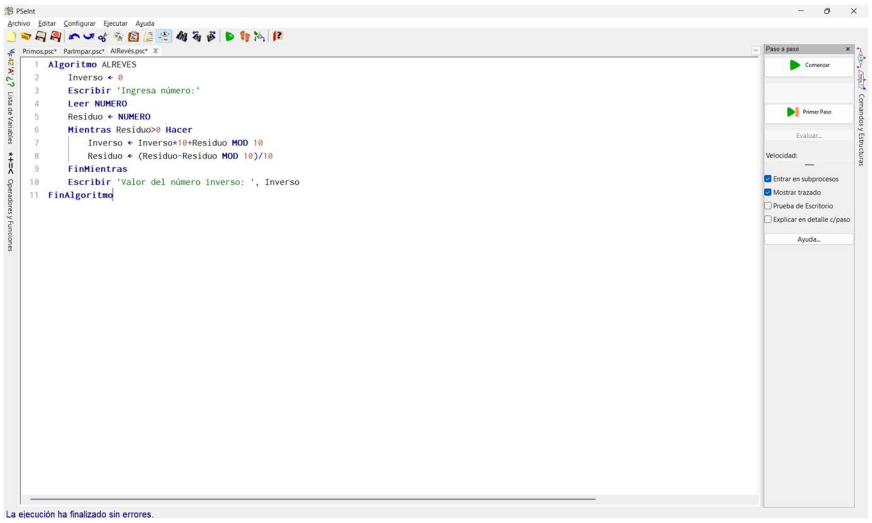


Nota. El diagrama ilustra la estructura condicional y el ciclo repetitivo usado para clasificar los números.

Números invertidos

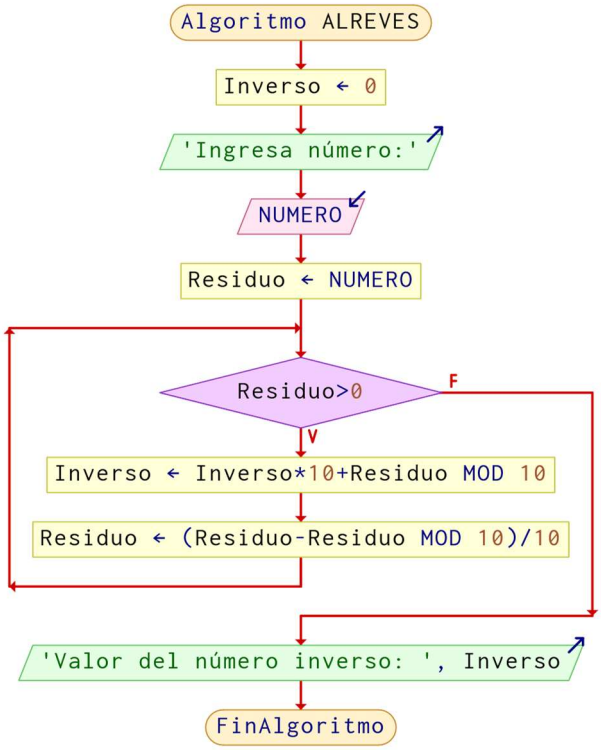
Este diagrama invierte las cifras de un número ingresado por el usuario. Emplea operaciones de módulo y división entera para separar cada dígito desde el último hacia el primero, construyendo así el número invertido.

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. *Diagrama de flujo del algoritmo "ALREVES" elaborado en PSDraw.*



Nota. El diagrama detalla el uso de un ciclo y operaciones matemáticas para invertir un número entero.

Conclusión

La elaboración de diagramas de flujo como parte de esta actividad ha resultado ser una herramienta muy útil para visualizar la lógica que hay detrás de cada algoritmo. Convertir procesos abstractos en representaciones gráficas permite identificar los pasos clave que se deben seguir para resolver un problema, así como las decisiones que influyen en el resultado.

En esta ocasión, trabajar con algoritmos relacionados con números primos, pares/impares y números invertidos ha demostrado cómo tareas matemáticas sencillas pueden descomponerse en procesos claros y ordenados. Los diagramas de flujo también facilitan la documentación del trabajo realizado, haciendo más accesible la comprensión de un programa para otros. Además, constituyen una base sólida para futuros proyectos de programación, ya que enseñan a estructurar correctamente las soluciones antes de comenzar a codificar. En resumen, esta actividad ha sido clave para reforzar habilidades de lógica, análisis y organización, indispensables en el desarrollo de software.

Referencias

PSeInt. (2023). PSDraw [Software]. PSeInt Project. https://pseint.sourceforge.net/

Academia Global-MX. (2025, 24 de julio). I Introducción al Desarrollo de Software#2- Pantalla compartida con vista del orador [Grabación de Zoom].