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: Alan Giovanni Nieto Becerril

FECHA: 17/04/25

Índice

Introducción

Los diagramas de flujo son representaciones que describen un proceso o algoritmo. Son utilizados en numerosos campos, no solo el informático, para planificar, mejorar y comunicar procesos que suelen ser complicados de entender si se comunican de forma hablada o escrita. Estos diagramas utilizan distintas figuras como rectángulos, óvalos y diamantes para representar distintos procesos de la secuencia. Además, emplean flechas conectoras para determinar cual es el flujo de la información. Los diagramas pueden describir simples o múltiples rutas, al igual que pueden ser a mano o computarizados. Los símbolos de diagramas de flujo más comunes pueden representar una terminal, un proceso, un documento, una decisión, los datos de entrada o salida, los datos almacenados, la flecha de flujo, algún comentario, un proceso predefinido, una referencia dentro y fuera de la página, entre otros. Existen numerosos tipos de diagramas de flujo que se aplican a diferentes áreas, algunos serían los diagramas de flujo de documentos, de flujo de datos, de flujo de sistemas, de flujo de programas, etc.

Descripción

En la actividad anterior, realizamos tres distintos algoritmos para tres calculadoras diferentes. El objetivo de esta actividad es realizar el diagrama de flujo de cada algoritmo de la actividad anterior, se realizarán en la plataforma “lucidchart”. Se debe recordar que la primera calculadora se llama ‘primos’ y tiene la función de leer el número ingresado por el usuario y establecer si se trata de un número primo o no. Un número primo solo es divisible entre 1 y entre sí mismo. La segunda calculadora, lleva por nombre ‘par/impar’ y esta recibe 10 números enteros del usuario y determina cuales son pares y cuales son impares. Serán pares cuando al dividir entre dos no queda ningún residuo. La tercera calculadora, se denomina ‘al revés’ y tiene el objetivo de recibir un número entero de 4 dígitos proporcionado por el usuario, después utiliza divisiones para invertir el orden del número. Por ejemplo, si se ingresa el número 1234, debe dar como salida al número 4321.

Justificación

La importancia de realizar diagramas de flujo después de haber descrito los algoritmos de la actividad pasada y antes de pasar el proceso a un lenguaje de programación (C) recae en la facilidad de comunicar la secuencia que se quiere seguir. Como se mencionó anteriormente, los diagramas de flujo nos permitirán explicar a alguna otra persona el flujo que sigue un programa o algoritmo. Estos diagramas se pueden usar para explicar detalladamente cual es la lógica en la estructura de un programa. También ayuda a darnos a nosotros mismos una perspectiva general y una guía que nos ayude al momento de escribir el código, que realizaremos en nuestro proyecto final. Antes de realizar nuestro diagrama de flujo, se tuvo que conocer cual es el objetivo de nuestras calculadoras y analizar los procedimientos que se requieren para cumplir tal objetivo. Se deben respetar ciertas reglas al momento de elaborar los diagramas, como la orientación (de arriba hacia abajo y de izquierda a derecha) para que pueda comunicar de forma eficiente el proceso.

Desarrollo

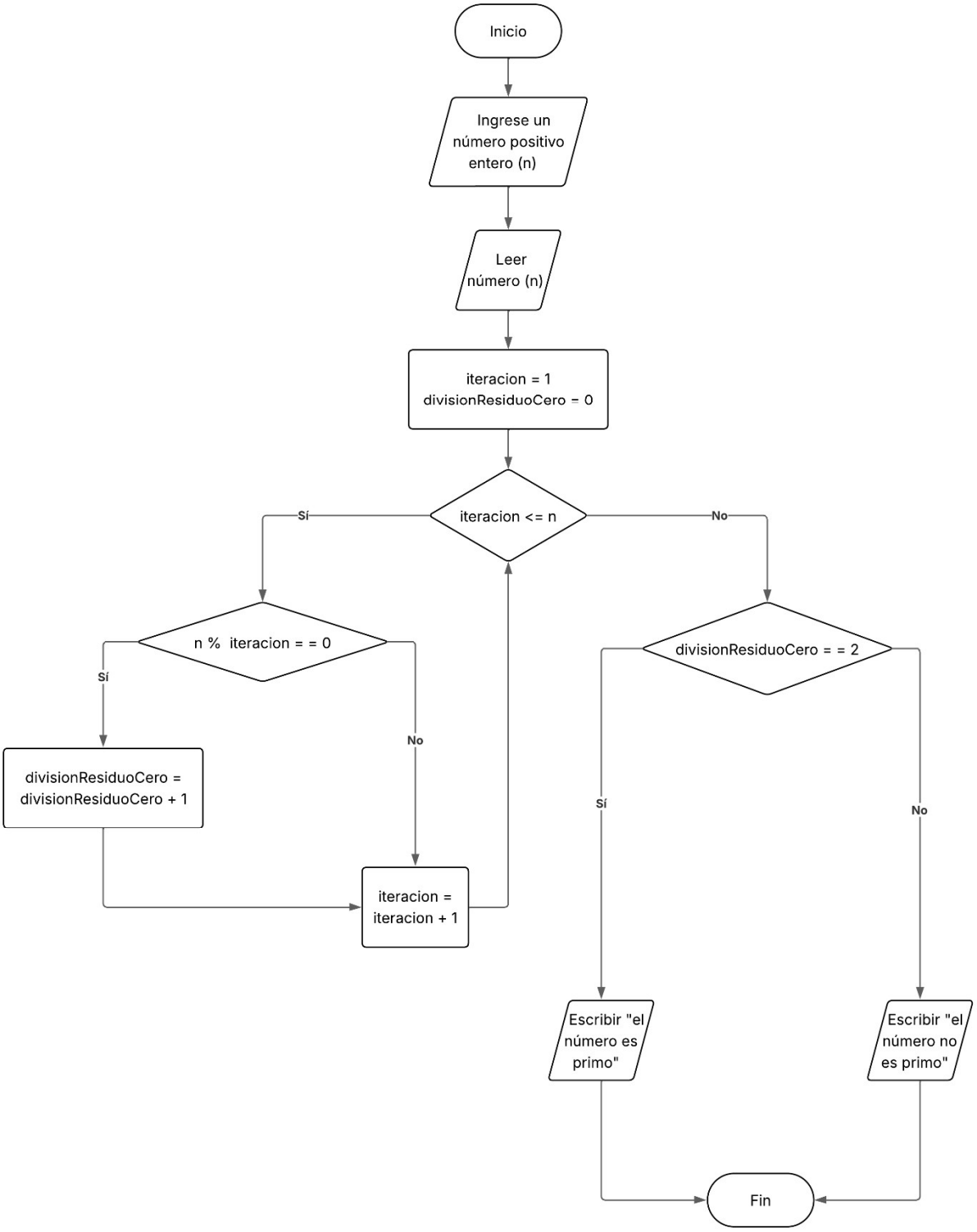

Figura 1. Diagrama de flujo ‘Primos’.

En el diagrama anterior se utiliza un óvalo con la palabra inicio para dar comienzo a la secuencia de pasos, Después, encontramos dos paralelogramas que indican una entrada de datos, en esta parte estamos solicitándole al usuario que ingrese un número con el que se trabajará. A continuación viene la inicialización de variables dentro de un rectángulo que simboliza un proceso. Luego viene un rombo de decisión donde la computadora debe responder si la iteración es menor o igual al número ingresado por el usuario. En caso positivo vamos a otro rombo donde se determina si el residuo de la división del número del usuario entre el valor de la iteración es igual a 0, en caso afirmativo divisionResiduoCero incrementa una unidad al igual que iteración, en caso contrario solo iteración incrementa una unidad. Regresamos al primer rombo de decisión donde tenemos un nuevo valor de iteración y se evalúa si es menor o igual al número del usuario. Ahora, en caso negativo, vamos al rombo de decisión de la derecha donde en caso de que divisionResiduoCero sea igual a dos, va a dar como dato de salida (dentro de un parelelograma) que es un número primo. En cualquier otro caso, dirá que no es número primo. Recordemos que un número primo solo tiene dos divisores, el uno y a sí mismo.

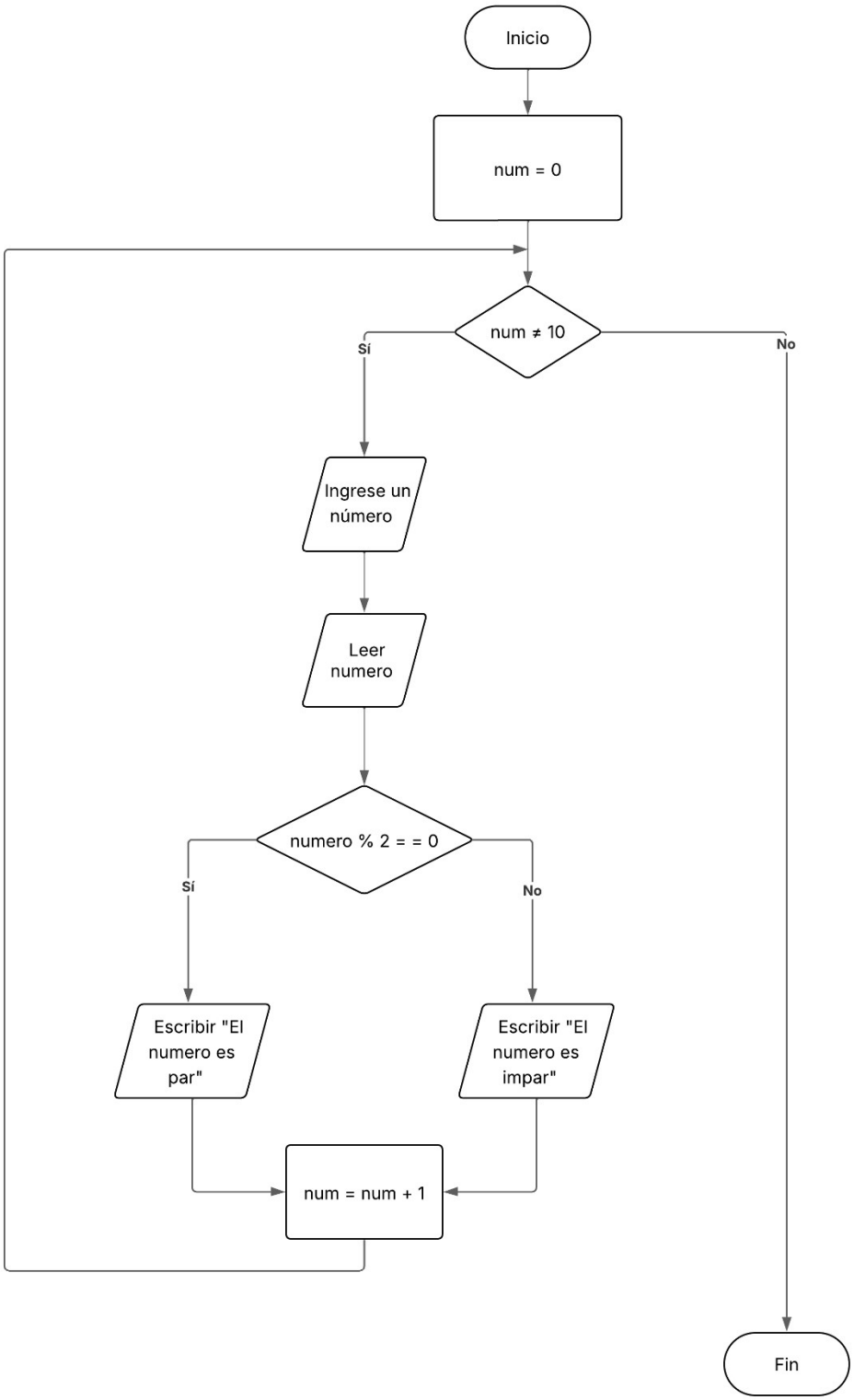

Figura 2. Diagrama de flujo ‘Par/Impar’

El segundo diagrama después de dar inicio, inicializa a la variable num = 0 (dentro de un rectángulo de proceso). Después, entra en un rombo de decisión donde se pregunta si num es desigual a 10. Como al inicio num es igual a 0, se cumplirá la condición y se le solicitará al usuario que ingrese un número. A continuación nos iremos a otro rombo de decisión donde se evalúa si el residuo de la división del número del usuario entre dos es igual a 0. En caso afirmativo, se indica al usuario que es un número par. En caso negativo, se indica al usuario que es un número impar. Luego se le suma una unidad a num y regresa al primer rombo de decisión donde iterará 9 veces más. Cuando num sea igual a 10, la condición de este rombo será falsa, por lo que terminará el algoritmo.

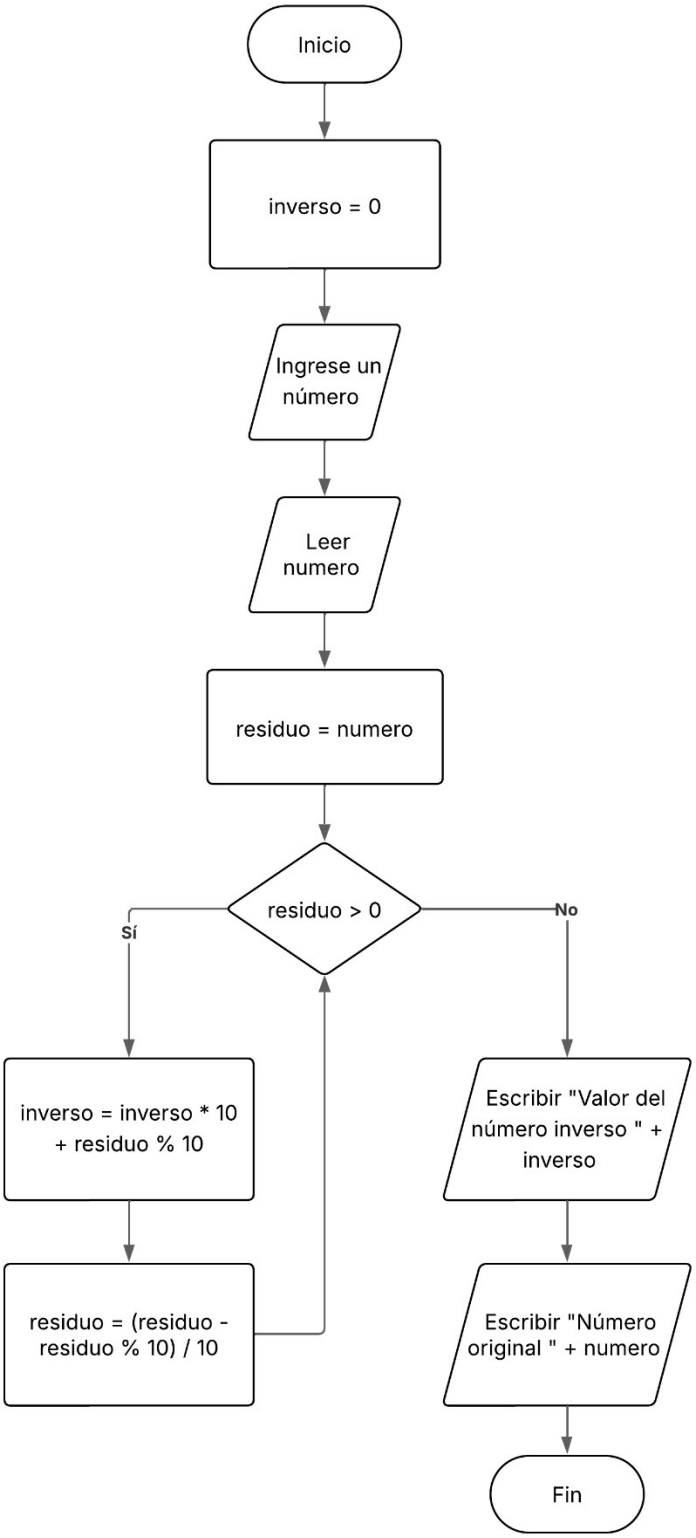

Figura 3. Diagrama de flujo ‘Al revés’

El tercer diagrama, también comienza con un óvalo inicio. Inicializa la variable inverso = 0 dentro de un rectángulo de proceso. En paralelogramas, solicita y lee un número de 4 dígitos ingresado por el usuario que almacena en la variable numero. Después, dentro de otro proceso, le asigna el valor de numero a la variable residuo. Con esta variable puede ir al rombo de decisión donde se evalúa si residuo es mayor que 0. En caso positivo se realizan unas operaciones que actualizan el valor de inverso y de residuo. Posteriormente, se regresa al rombo de decisión anterior y se vuelve a evaluar si residuo es mayor que 0. Cuando residuo ya no sea mayor que 0, se le da como dato de salida al usuario el número invertido y el número original. Finalmente, se acaba el diagrama.

Conclusión

En esta actividad se realizó de manera satisfactoria los tres diagramas de flujo para cada calculadora. Personalmente, me considero una persona muy visual, por lo que entender un diagrama de flujo como estos me parece más sencillo comparado con la lectura de un algoritmo, un pseudocódigo y también de algún script. Me pareció muy acertado el objetivo de las tres actividades de este curso, debido a que siguen una secuencia lógica. Es decir, el objetivo final es programar las calculadoras, pero antes de llegar a ese punto debemos comprender cual va a ser la manera en que cada calculadora funciona, tuvimos que describir estos pasos en un algoritmo y ahora representarlos con el uso de los diagramas. En la vida cotidiana, he utilizado estos diagramas para tomas de decisiones personales que me ayudan a observar distintas consecuencias de mis propias acciones. También los he llegado a observar para realizar algunos trámites, como los casos en que se debe realizar la declaración de impuestos.

Referencias

- Qué es un diagrama de flujo. (s. f.). Lucidchart. https://www.lucidchart.com/pages/es/que-es-un-diagrama-de-flujo

- Diagrama de flujo. (s.f.). Universidad Veracruzana. https://www.uv.mx/personal/aherrera/files/2020/05/DIAGRAMAS-DE-FLUJO.pdf

Liga Github: https://github.com/AlanGNB/IDS