INTRODUCCIÓN

Para la construcción de un programa es necesario el diseño previo de un algoritmo, de modo que sin algoritmo no puede existir un programa.

Los algoritmos son independientes tanto del lenguaje de programación en que se expresan como de la computadora que los ejecuta. En cada problema el algoritmo se puede expresar en un lenguaje diferente de programación y ejecutarse en una computadora distinta; sin embargo, el algoritmo será siempre el mismo. Así, por ejemplo, haciendo una comparación con la vida diaria, una receta de un plato de cocina se puede expresar en español, inglés o francés, pero cualquiera que sea el lenguaje, los pasos para la elaboración del plato se realizarán sin importar el idioma del cocinero.

En la ciencia de la computación y en la programación, los algoritmos son más importantes que los lenguajes de programación o las computadoras. Un lenguaje de programación es tan sólo un medio para expresar un algoritmo y una computadora es sólo un procesador para ejecutarlo. Tanto el lenguaje de programación como la computadora son los medios para obtener un fin, conseguir que el algoritmo se ejecute y se efectúe el proceso correspondiente.

Dada la importancia del algoritmo, un aspecto muy importante será *el diseño de algoritmos*.

El diseño de la mayoría de los algoritmos requiere creatividad y conocimientos de la técnica de la programación. En esencia, ***la solución de un problema se puede expresar mediante un algoritmo*.**

¿Qué es un algoritmo?

Un algoritmo es una serie de pasos organizados que describe el proceso que se debe seguir, para dar solución a un problema específico.

Otras definiciones de Algoritmo pueden ser:

- *Un **Algoritmo** es un conjunto finito de instrucciones u operaciones que ejecutadas en un orden determinado permiten resolver un problema.*
- *Un **Algoritmo** es una secuencia de instrucciones precisas que llevan a una solución.*

DISEÑO DE ALGORITMOS

En la etapa de análisis en La Metodología de Resolución de Problemas se determina *qué* debe hacer el programa. En la etapa de diseño del algoritmo se determina *cómo* hace el programa la tarea solicitada. Previamente debemos definir el enfoque que utilizaremos para su diseño. Existen enfoques alternativos a los procesos de la solución de problema, se los conoce como Paradigmas de Programación.

Un paradigma de programación representa fundamentalmente enfoques diferentes para la construcción de soluciones a problemas y por consiguiente afectan al proceso completo del desarrollo de programas.

Los paradigmas de programación clásicos son: Procedimental (o Imperativo), funcional, declarativo y orientado a objetos. Durante el curso centraremos nuestro diseño de algoritmos basado en el enfoque procedimental.

El **paradigma imperativo o procedimental** representa el enfoque o método tradicional de programación. Este paradigma define el proceso de programación como el desarrollo de una secuencia de órdenes (*comandos*) que manipulan los datos para producir los resultados deseados. Por consiguiente, el paradigma procedimental señala un enfoque del proceso de programación mediante la realización de un algoritmo que resuelve de modo manual el problema y a continuación expresa ese algoritmo como una secuencia de órdenes.
Dentro de este Paradigma de Programación se encuentran la **programación modular y** la **programación estructurada**.

Dichos enfoques o métodos son los más eficaces para el proceso de diseño en esta parte de introducción a la programación.

La **Programación Modular** se basa en el conocido divide y vencerás. Es decir, la resolución de un problema complejo se realiza dividiendo el problema en subproblemas y a continuación dividiendo estos subproblemas en otros de nivel más bajo, hasta que pueda ser *implementada* una solución en la computadora. Este método se conoce técnicamente como diseño descendente (***topdown***) o **modular**. El proceso de romper el problema en cada etapa y expresar cada paso en forma más detallada se denomina *refinamiento sucesivo*.

Cada subprograma es resuelto mediante un **módulo** (*subprograma*) que tiene un solo punto de entrada y un solo punto de salida.

Cualquier programa bien diseñado consta de un *programa principal* (el módulo de nivel más alto) que llama a subprogramas (módulos de nivel más bajo) que a su vez pueden llamar a otros subprogramas. Los programas estructurados de esta forma se dice que tienen un *diseño modular* y el método de romper el programa en módulos más pequeños se llama *programación modular*.

La utilización de la técnica de diseño ***Top-Down*** tiene los siguientes objetivos básicos:

- *Simplificación del problema y de los subprogramas de cada descomposición.*
- *Las diferentes partes del problema pueden ser programadas de modo independiente e incluso por diferentes personas.*
- *El programa final queda estructurado en forma de bloque o módulos lo que hace más sencilla su lectura y mantenimiento.*

La **Programación Estructurada** se basa en el diseño modular (Programación Modular), donde cada módulo es diseñado en forma descendente (TopDown) y cada módulo se diseña basándose en la combinación de tres estructuras básicas.

La Programación Estructurada nos dice que:

Todo Algoritmo puede ser construido en forma correcta utilizando únicamente tres estructuras básicas. Secuencial, Decisión e Iteración.

El proceso que convierte los resultados del análisis del problema en un diseño modular con refinamientos sucesivos que permitan una posterior traducción a un lenguaje se denomina **diseño del algoritmo**.

Características de los algoritmos

Las características fundamentales que debe cumplir todo algoritmo son:

- Un algoritmo debe ser *preciso, cada instrucción utilizada debe ser clara y sin ambigüedades.*
- *Un algoritmo debe producir al menos un resultado.*
- *Un algoritmo debe estar compuesto por una cantidad finita de instrucciones.* Si se sigue un algoritmo, se debe terminar en algún momento; o sea, debe tener un número finito de pasos.
- Un algoritmo bebe estar compuesto por instrucciones que sean *suficientemente básicas para que pueda ser resuelta por cualquier computadora.*

La definición de un algoritmo debe describir tres partes: *Entrada*, *Proceso* y *Salida.*

Lenguajes Algorítmicos

Los lenguajes algorítmicos son herramientas que nos permiten diseñar y escribir los algoritmos. *Son* una serie de símbolos y reglas que se utilizan para describir de manera explícita un proceso.

Tipos de Lenguajes Algorítmicos:

- **Gráficos:** *Es la representación gráfica de las operaciones que realiza un algoritmo. (**Diagrama de Flujo)***
- **No gráficos:** *Representa en forma descriptiva las operaciones que debe realizar un algoritmo. (**Pseudocódigo**)*

Diagrama de flujo

Un diagrama de flujo es la representación gráfica de un algoritmo. También se puede decir que es la representación detallada en forma gráfica de cómo deben realizarse los pasos en la computadora para producir resultados.

Esta representación gráfica se da cuando varios símbolos (que indican diferentes procesos en la computadora), se relacionan entre si mediante líneas que indican el orden en que se deben ejecutar los procesos.

PSEUDOCÓDIGO

Mezcla de lenguaje de programación y español (o ingles o cualquier otro idioma) que se emplea, dentro de la programación estructurada, para realizar el diseño de un programa. En esencial, el pseudocódigo se puede definir como un lenguaje de especificaciones de algoritmos.

Es la representación narrativa de los pasos que debe seguir un algoritmo para dar solución a un problema determinado.

El pseudocódigo utiliza palabras que indican el proceso a realizar.

ESTRUCTURAS ALGORÍTMICAS

Las estructuras de operación de programas son un grupo de formas de trabajo, que permiten, mediante la manipulación de variables, realizar ciertos procesos específicos que nos lleven a la solución de problemas. Estas estructuras se clasifican de acuerdo con su complejidad en:

Secuencial

- Ingresos.
- Egresos.
- Procesos (asignación y operadores aritméticos).

Condicional

A las estructuras secuenciales se suman las decisiones:

- Simples.
- Dobles.
- Múltiples.

BIBLIOGRAFÍA

- Osvaldo Cairó Battistutti (2005). Metodología de la Programación. Editorial Alfaomega.
- Pedro Vicente Rosero Montaño (2006). Introducción a la Programación de Computadores.
- Luis Joyanes Aguilar (2008). Fundamentos de Programación. Cuarta edición. Editorial McGRAW-HILL.
- Francisco Javier Pinales Delgado, César Eduardo Velázquez Amador. Problemario de algoritmos resueltos con diagrama de flujos y pseudocódigo. Universidad Autónoma de Aguascalientes.