Estructuras de Control - Diagrama de Flujo

Introducción a la Programación - Introducción a la Computación - Fundamentos de la Informática

Ing. Electrónica - T.U.G. - T.U.E. - T.U.R. - T.U.W.- Prof. Tec. Elect. - T.U.T - T.U.M

Área de Servicios

Departamento de Informática - FCFMyN - Universidad Nacional de San Luis

Índice

1. Programación Estructurada

Una estructura se define como un *esquema* con cierta distribución y orden que permite representar una idea de forma simplificada.

La programación estructurada promueve diseñar y codificar de acuerdo a las siguientes reglas:

- Se codifica utilizando tres estructuras básicas de control básicas: secuencial, condicional y repetitiva.
- Diseño modular.
- Los módulos son diseñados de modo descendente (Top-Down).

El término programación estructurada hace referencia a un conjunto de técnicas que aumentan considerablemente la productividad del programa reduciendo significativamente el tiempo necesario para escribir, verificar, depurar y mantener los programas. Al utilizar un número limitado de estructuras de control se minimiza la complejidad de los programas y por consiguiente se reducen los errores.

La visión clásica de la programación estructurada se refiere al control de ejecución, aspecto fundamental a tener en cuenta al diseñar y codificar un programa en un lenguaje de alto nivel. La regla general establece que las instrucciones se ejecuten sucesivamente una tras otra, algunas de ellas se ejecutan o no dependiendo de que se cumpla alguna condición y otras instrucciones (los bucles) deben ejecutarse varias veces, ya sea en número fijo o hasta que se cumpla una condición determinada. Algunos lenguajes de programación más antiguos se apoyaban en una sola instrucción para modificar la secuencia de ejecución de las instrucciones: la instrucción *goto*, del inglés "go to"que significa *ir a*, la cual provocaba una transferencia incondicional del control a alguna sentencia en particular. Estas transferencias arbitrarias del flujo de la ejecución hacen los programas muy poco legibles y difíciles de comprender. A finales de los años sesenta, surgió una nueva forma de programar que reduce a la mínima expresión o elimina el uso de la instrucción goto y la sustituye por otras más comprensibles.

Esta metodología de programar se basa en un famoso teorema, desarrollado por Edsger Dijkstra, que demuestra que todo programa puede escribirse utilizando únicamente tres estructuras básicas de control: *la secuencia, el condicional y la repetición o iteración*.

1.1. Estructuras de Control básicas

Una estructura de control tiene un único punto de entrada y un único punto de salida y se compone de sentencias o de otras estructuras de control. Las estructuras básicas son:

- **Secuencial:** Es aquella que ejecuta las acciones sucesivamente, una a continuación de otra sin posibilidad de omitir ninguna, y naturalmente, sin bifurcaciones. Es la más sencilla de todas las estructuras, simplemente le indica al procesador que debe ejecutar de forma consecutiva una lista de acciones (que pueden ser, a su vez, otras estructuras de control). Para construir una secuencia de acciones basta con escribir una acción a continuación de la otra o, en algunos casos, se utiliza un operador al final de cada sentencia.
- **Condicional o Alternativa:** La estructura alternativa permite bifurcar el "flujo"del algoritmo en función de una expresión lógica. Se utilizan cuando en el desarrollo de la solución de un problema se debe tomar una decisión para establecer un proceso o un camino alternativo a seguir. Esta toma de decisión se basa en la evaluación de una o más condiciones que señalan alternativas o consecuencias, es decir, el camino (rama) a seguir.
- **Repetición o Iteración:** La estructura repetitiva o iterativa permite, como su nombre lo indica, repetir una acción (o grupo de acciones) un número prefijado de veces o depender de la evaluación de una expresión lógica.

2. Lenguajes Algorítmicos

En esta instancia el objeto de estudio se basa en los métodos de resolución de problemas, es decir, los algoritmos y no los programas, o sea, sus implementaciones concretas usando diferentes lenguajes de programación. El desarrollo

de algoritmos es un tema fundamental e importante en el diseño de programas, aplicaciones de cómputo y soluciones informáticas.

Los algoritmos pueden describirse, en mayor o menor detalle, utilizando diversos lenguajes: *Lenguajes Algorítmicos*. Consisten de una serie de símbolos y reglas que se utilizan para especificar un proceso en forma precisa. Considerando la forma en que describen un proceso, se pueden categorizar en:

- **Lenguaje Gráfico:** Es la representación gráfica de las operaciones que realiza un algoritmo (diagrama de flujo).
- **Lenguaje No Gráfico:**Representa en forma descriptiva las operaciones que debe realizar un algoritmo.

Son lenguajes algorítmicos: Lenguaje Natural, Diagrama de Flujo, Lenguaje de Programación de Algoritmos.

2.1. Diagrama de Flujo

Toda representación gráfica, de cualquier tipo que sea, se caracteriza por:

- Sencillez. Un método gráfico de diseño de algoritmo debe permitir la construcción de estos de manera fácil y sencilla.
- Claridad. Cuando un algoritmo es representado por un método gráfico debe estar lo suficientemente claro para un fácil reconocimiento de todos los elementos por parte de otra persona distinta de la que lo diseñó, .
- Normalización. Tanto los diseñadores de programas como los usuarios que necesitan la documentación de estos deben reconocer y utilizar las mismas normas de documentación.
- Flexibilidad. Todo método gráfico de representación debe permitir, sin grandes dificultades, posteriores modificaciones de algunas partes de un algoritmo y la inserción de alguna nueva.

Un diagrama de flujo es la representación gráfica de un algoritmo. Es la representación detallada en forma gráfica de cómo debe realizar los pasos, por ejemplo, una computadora para producir resultados. Esta representación gráfica se da cuando varios símbolos (que indican diferentes procesos), se relacionan entre si mediante líneas que indican el orden en que se deben ejecutar los procesos. Los símbolos utilizados han sido normalizados por el Instituto Norteamericano de Normalización (ANSI):

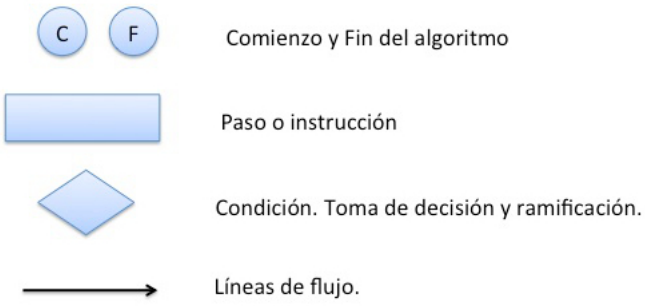

Figura 1: Simbología de Diagramas de Flujo

El diagrama de flujo goza de ventajas como el ser altamente intuitivo, fácil de leer, claro y preciso. Representa una herramienta muy potente de cara a comenzar a programar ya que su contenido gráfico lo hace menos árido que el algoritmo expresado en lenguaje natural (pseudocódigo). Las limitaciones principales de los diagramas de flujo derivan precisamente de su carácter de dibujo. No resultan tan fáciles de crear o de mantener como el texto del pseudocódigo. Un diagrama de flujo tiene un único punto de entrada y un único punto de salida y la simbología para representar a las diferentes estructuras de control es:

- Secuencia

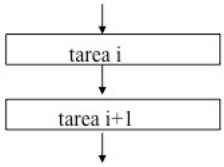

Figura 2: Representación de la Secuencia

- Selección

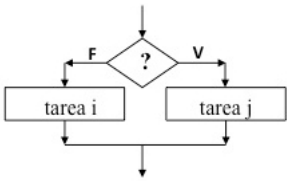

Figura 3: Representación de la Selección

- Iteración o Repetición

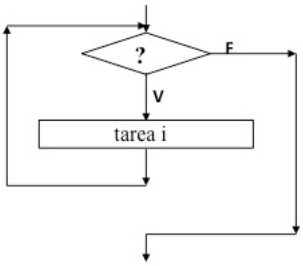

Figura 4: Representación de la Iteración