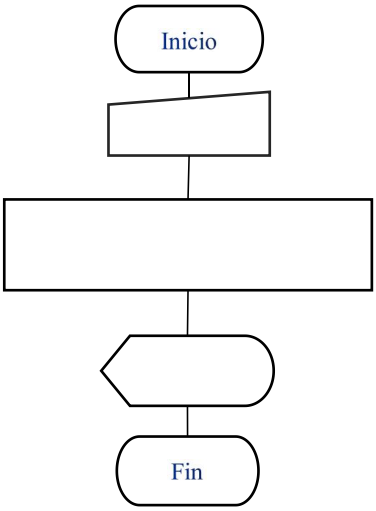
Programación Estructurada

En general, en un programa (escrito dentro de un solo módulo), las instrucciones se ejecutan secuencialmente, es decir, una después de otra según el orden en que se han colocado. Esto ya lo vimos en los programas que hemos escrito hasta ahora.

Así que la programación que hemos realizado hasta ahora es secuencial. Permite escribir programas como una secuencia de instrucciones que se ejecutan una tras otra, invariantemente, desde el principio hasta el final. No importa cuantas veces se ponga a correr el programa, siempre se ejecutará de la misma manera. Los resultados diferentes en cada ocasión, solo dependerán de los diferentes datos de entrada que se proporcionen cada vez.

A ello se debe que los flujogramas (esquemas gráficos) de diseño de todos los programas anteriores, escritos en un solo módulo, han tenido la misma estructura:



Pero no todos los procesos que se dan en el mundo real -fábricas, empresas, oficinas, restaurantes, instituciones educativas, en el hogar, etc.- son secuenciales. Muchos requieren de realizar sub procesos alternativos de acuerdo a unas condiciones previas, lo que implica la necesidad de tomar decisiones en la realización de tareas. Otros requieren de la realización repetida de sub procesos, hasta alcanzar un objetivo.

Así que muchas veces, de acuerdo a cierto criterio, sería conveniente que la siguiente instrucción a ejecutar no fuera la que viene a continuación, sino que la ejecución debiera transferirse a otro punto del programa. Para ello existen las instrucciones de control de flujo, que permiten programar estructuradamente.

Tradicionalmente estas instrucciones son:

a) Las instrucciones para la toma de decisiones.

b) Las instrucciones para la realización de iteraciones (repeticiones).

El primer paradigma de programación de computadoras inventado vino a resolver lo engorroso que era escribir programas que resolvieran tareas, los cuales se escribían en lenguaje ensamblador o en lenguaje de máquina. Se inventaron, entonces, los

lenguajes de alto nivel, que permitieron programar bajo un paradigma que se dio en llamar paradigam de programación estructurada.

Este paradigma permite fundamentalmente tres cosas:

a) Incorporación de instrucciones de control de flujo en nuestros programas.

b) Modularizar aquellos programas que deben resolver problemas que son particularmente grandes y/o complejos, según se explicó en el documento anterior.

c) Resolver las tareas utilizando comandos que son equivalentes a palabras retomadas del lenguaje humano. Esta es una característica fundamental de un lenguaje de alto nivel.

De hecho, el primer lenguaje de programación de computadoras de alto nivel y que permitía programar bajo el paradigma de la programación estructurada fue el FORTRAN. Este lenguaje nació entre finales de la década de los 50 y principios de la década de los 60, del siglo pasado. Su nombre significa:

Fortran: Formula translation language (lenguaje traductor de fórmulas)

Fue creado para la solución de problemas del área de las matemáticas e ingeniería. Fue bastante aceptado y ampliamente difundido, a tal punto que se le hicieron varias revisiones y ampliaciones a través de la historia. A pesar de ser tan antiguo, puede decirse que sigue siendo totalmente valido e importante su uso. La ciencia de los Métodos Numéricos fue, durante muchas décadas, enseñada utilizando este lenguajes y solo fue sustituido por la "innovación" que suponía el uso de otros lenguajes más recientes y novedosos, que daban unas prestaciones que van más allá del área de los métodos numéricos propiamente dicha y que benefician el desempeño de los programadores. Ejemplo de esto es **Octave**, que no solo es un lenguaje de programación, sino un entorno que permite: intercambiar la vista entre una consola que simula una calculadora y un área de edición de programas, revisión del estado de las variables, visualización del historial de comandos, manipulación relativamente sencilla de gráficos, incorporación del concepto de matemática simbólica (y por tanto de inteligencia artificial) en la solución de varias tareas, ejecución de tareas complejas con la ejecución de un solo comando u operador, etc.

La programación estructurada también puede verse como un conjunto de reglas que desarrollan en el programador los hábitos para lograr un buen estilo. Es bastante flexible para permitir una considerable creatividad y expresión personal; pero sus reglas imponen suficientes restricciones para hacer que los programas resultantes sean muy superiores a sus versiones no estructuradas. El producto terminado es mucho más elegante y relativamente más fácil de entender que un programa no estructurado.

La idea clave de la programación estructurada es que cualquier algoritmo computacional requiere de tan sólo tres tipos de instrucciones de control de flujo fundamentales (algunos también les llaman estructuras de control de flujo):

a) Secuencias, conocidas también como estructuras o instrucciones secuenciales.

b) Selección o decisión, también conocidas como estructuras o instrucciones condicionales.

c) Instrucciones de repetición, también conocidas como iteraciones, bucles, lazos o ciclos.

Limitándonos a dichas estructuras, el programa será relativamente claro y fácil de seguir.

La programación estructurada busca que los programas sean:

- Probables: debe poder probarse que el programa funcionará bien.
- Mantenible: es decir que se le pueda dar mantenimiento, que se pueda adaptar a diferentes realidades y que los cambios que se requieran se hagan con facilidad.
- Confiables: debe tenerse la certeza de que bajo las mismas condiciones el programa hará siempre lo mismo.

Teorema de la programación estructurada

Este teorema señala que la combinación de las tres estructuras básicas: secuencia, selección e iteración, son suficientes para expresar cualquier función computable.