Diagramas de Flujo

Son una representación gráfica de la secuencia lógica de las operaciones o acciones que debe realizar un ordenador, así como la secuencia o el flujo de datos, de una manera gráfica, para la resolución de un problema.

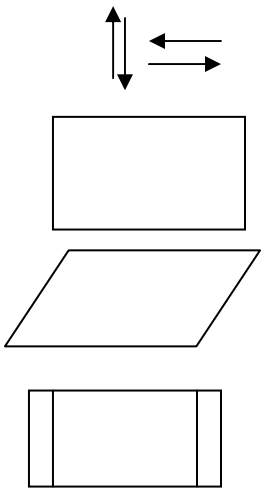
Todo diagrama de flujo (también conocido como ordinograma en este nivel) debe cumplir las siguientes características:

- Debe ser independiente del lenguaje de programación que se utilizará.
- Debe ser normalizado para facilitar el intercambio de documentación
- Debe ser intuitivo para facilitar su entendimiento.
- Debe ser flexible para facilitar futuras modificaciones.

El diseño de todo ordinograma debe reflejar:

1. Un principio o inicio que marca el comienzo de ejecución del programa y que viene determinado por la palabra INICIO
2. La secuencia de operaciones, lo mas detallada posible y siguiendo el orden en que se deberán de ejecutar (de arriba-abajo y de izquierda-derecha)
3. Un final que marca la finalización de la ejecución del programa y que viene determinado por la palabra FIN.
4. Todos los símbolos utilizados deben ser los normalizados y deben estar conectados por líneas de conexión o líneas de flujo de datos.
5. Las líneas de conexión no pueden cruzarse.
6. A un símbolo de proceso pueden llegarle varias flechas, pero de él solo puede salir una.
7. A un símbolo de decisión pueden llegarle varias flechas, y de él deben salir dos (decisión lógica) o más (decisión múltiple) flechas.
8. Al símbolo de INICIO no le llega ninguna flecha y solo parte una flecha.
9. Al símbolo de FIN le pueden llegar multiples flechas de conexión, pero de él no parte ninguna flecha.

Símbolos de operación o procesos

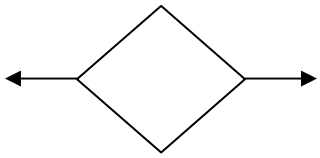


Flechas indicadoras de la dirección del flujo de datos

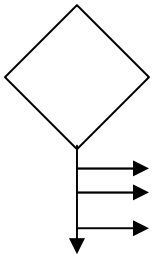
Proceso u operación en general

Operación de Entrada/Salida en general

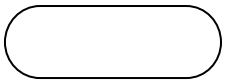
Subprograma o función, es decir, un módulo independiente que realizar una tarea y devuelve el control al programa



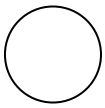
Decisión de dos salidas



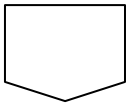
Decisión de N salidas



Símbolo de INICIO y de FIN



Conector de lineas de flujo en la misma página

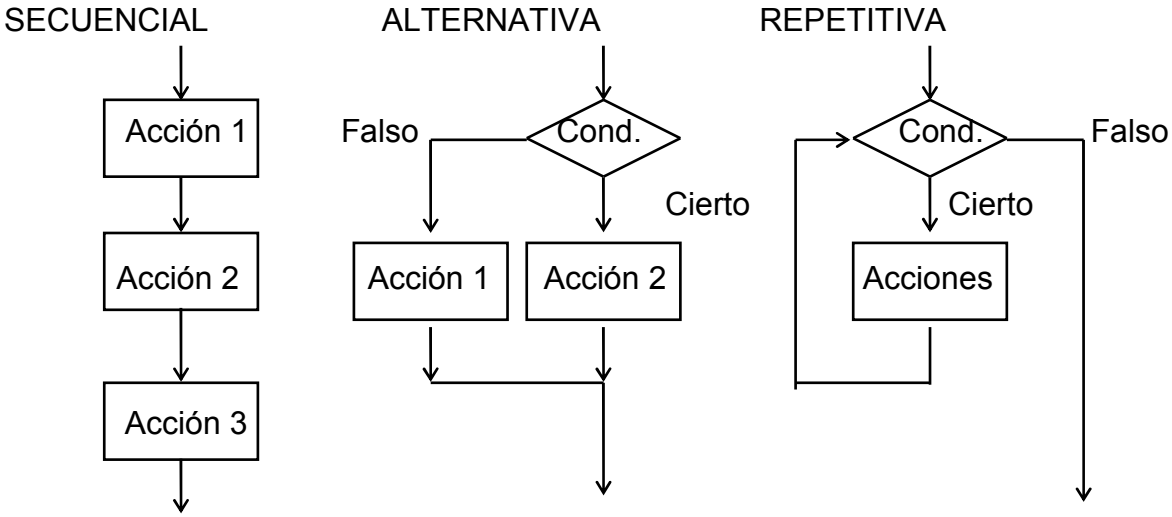


Conector de lineas de flujo en distinta página

Aunque los diagramas de flujo permitan todo tipo de bifurcaciones, sin ningún tipo de condición, la programación estructurada se basa en la descripción del los algoritmos mediante estructuras de control de flujo de tres tipos: secuenciales, alternativas, y repetitivas. Las estructuras de control de programación estructurada se caracterizan por tener una sola entrada y una única salida.

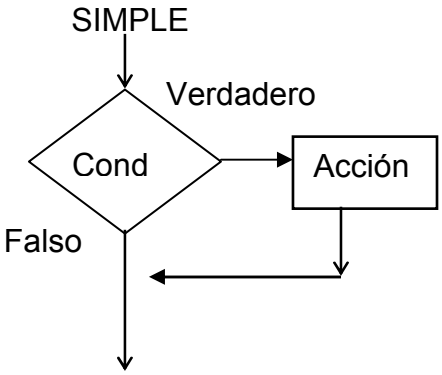
La representación de los bloques básicos de las distintas estructuras de control de la programación estructurada se muestran a continuación.

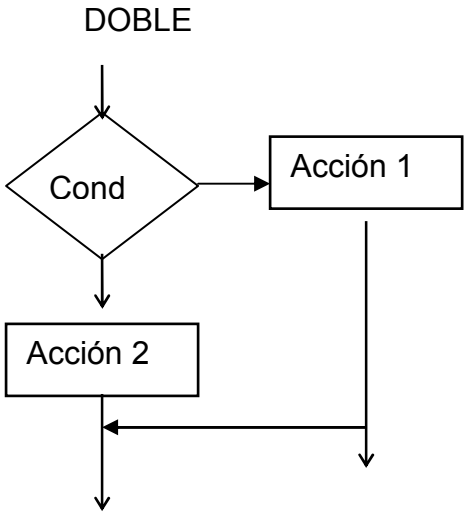
ESTRUCTURAS DE CONTROL

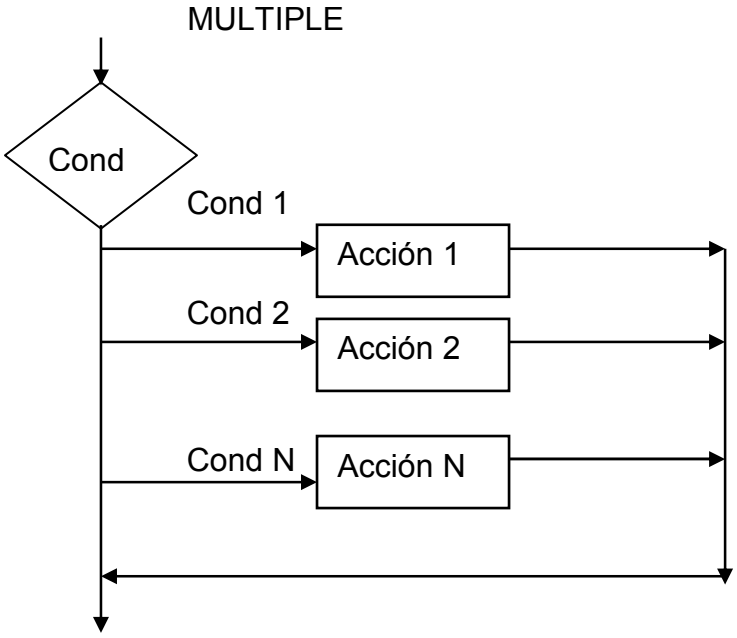


La estructura de control secuencial ejecuta las acciones sucesivamente unas a continuación de otras, sin posibilidad de omitir ninguna acción y sin poder hacer bifurcaciones. Tiene por tanto una sola entrada y una única salida.

Una estructura de control alternativa bifurca el flujo de un algoritmo según se cumplan una o varias condiciones. La estructura alternativa simple ejecuta una acción si la condición es cierta, en caso contrario se la salta. La estructura alternativa doble permite la elección entre dos acciones según la condición sea cierta o falsa. La estructura alternativa múltiple permite la elección entre varias acciones según los valores de una variable selector.

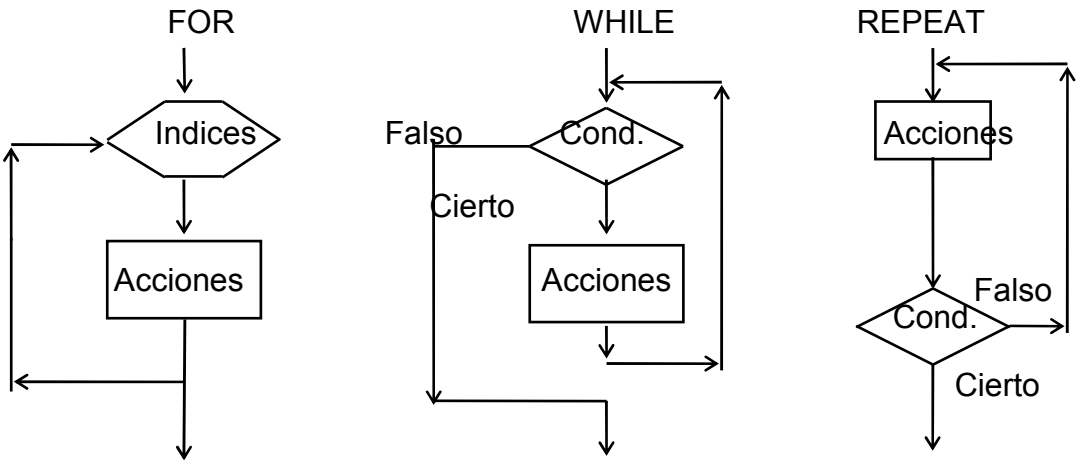






Una estructura de control repetitiva permite ejecutar las acciones un número de veces que puede estar definido a priori, o indefinido hasta que se cumpla una determinada condición. Se denomina bucle o lazo al conjunto de las acciones repetidas. Las estructuras de control repetitivas pueden ser de tres tipos: estructura FOR, estructura WHILE, y estructura REPEAT.

En la estructura FOR el número de repeticiones se conoce antes de realizar el bucle, por medio de sus índices. La estructura WHILE repite las acciones mientras la condición de control del bucle sea cierta, esta condición está colocada al principio del bucle. La estructura REPEAT repite las acciones mientras la condición de control del bucle sea falsa, esta condición está colocada al final del bucle.



Esta demostrado el teorema de la programación estructurada, que dice: todo algoritmo puede ser descrito utilizando solamente tres tipos de estructuras de control: secuencial, alternativa y repetitiva. No se necesitan bifurcaciones incondicionales para la descripción de los algoritmos.

Inconvenientes de los diagramas de flujo:

- Los diagramas de flujo de algoritmos complejos y detallados son muy laboriosos de realizar.
- Las acciones a seguir después de un símbolo de decisión, pueden ser difíciles de encontrar debido a la complejidad de los caminos.
- No existen normas fijas en la elaboración de los diagramas de flujo, que permitan introducir todos los detalles que el usuario desee.
- La facilidad de hacer bifurcaciones puede crear malos hábitos de programación, es decir se pueden construir programas poco estructurados.
- Los diagramas de flujo muestran la lógica de un algoritmo pero oscurecen su estructura.
- Normalmente se utilizan los diagramas de flujo para describir pequeños algoritmos, cuando se está comenzando a programar, o para mostrar esquemas muy generales de una aplicación informática.