Universidad Nacional del Litoral
Facultad de Ingeniería y Ciencias Hídricas
Departamento de Informática

FUNDAMENTOS DE PROGRAMACIÓN

Asignatura correspondiente al plan de estudios de la carrera de Ingeniería Informática

UNIDAD 3

ESTRUCTURAS DE CONTROL

2019

UNIDAD 3

Estructuras de Control

Introducción

La formalización algorítmica que se comenzó a desarrollar en el tema anterior, ha permitido resolver algunos problemas sencillos empleando algoritmos computacionales.

El objetivo de este tema es desarrollar nuevas herramientas algorítmicas para resolver problemas más complejos, mediante el diseño de algoritmos estructurados. Esto sentará las bases para facilitar más adelante el desarrollo de programas en un lenguaje estructurado.

En nuestros primeros diseños de algoritmos sólo se utilizaron tres acciones primitivas fundamentales: lectura, asignación y escritura. Estos algoritmos fueron resueltos en base a una estructura secuencial de acciones: los pasos o acciones indicados se ejecutan uno tras otro, a medida que van apareciendo; es decir, secuencialmente.

Pero en el diseño de algoritmos, generalmente es necesario modificar el orden secuencial de ejecución del conjunto de acciones. El diseño estructurado de algoritmos brinda recursos para resolver este tipo de situaciones.

Teorema Fundamental de la Programación Estructurada

Böhm y Jacopini demostraron que: "*Todo problema computacional --sin importar su complejidad-- puede resolverse empleando solo tres estructuras básicas de control. Estas son: una de tipo **SECUENCIAL**, otra de tipo **CONDICIONAL** y una de tipo **REPETITIVO***".

Cada una de esas estructuras conforma un segmento algorítmico perfectamente identificable de acceso y salida únicos.

Un algoritmo estructurado está conformado por segmentos de código. Cada uno de estos segmentos tiene un único punto de entrada y un único punto de salida.

Los algoritmos así diseñados son más legibles, y permiten seguir más fácilmente su lógica, ayudando a la detección de errores, modificación y mantenimiento.

De acuerdo a lo mencionado, las acciones algorítmicas se encuadran en alguna de las estructuras básicas de control siguientes:

- Secuenciales

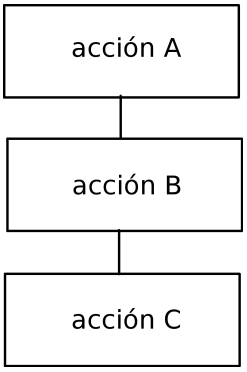
- Condicionales
- Repetitivas

Se analizarán -a continuación- cada una de ellas.

Estructura Básicas de Control

Estructura Secuencial

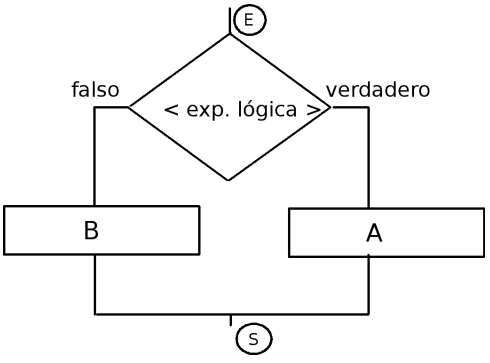
Las acciones correspondientes a esta estructura se van ejecutando en el orden en que aparecen, es decir secuencialmente. Esta estructura puede representarse esquemáticamente:



Recuérdese que los ejercicios correspondientes a las actividades del tema anterior, fueron resueltos empleando tres acciones primitivas fundamentales: Lectura, Asignación y Escritura. Su estructura general corresponde a las acciones de tipo secuencial.

Estructura Condicional Si-Entonces

Esta estructura implica una toma de decisión en el algoritmo, donde el ejecutante (la computadora) puede seguir un camino u otro, según el valor de verdad de una expresión lógica.
El esquema representativo de esta estructura es el siguiente:



Donde <exp. lógica> es cualquier proposición que arroje un resultado lógico: **Verdadero** o **Falso**.
La presencia de esta estructura en un algoritmo le indica al ejecutante que:

- Debe evaluar la expresión lógica planteada.
- Si es verdadera, debe ejecutar las acciones indicadas en A. Luego ir al fin de la estructura.
- Si es falsa, resolver el bloque B. Luego ir al fin de la estructura.

En el diagrama señalado, tanto el bloque A, como el B, pueden representar acciones primitivas elementales o alguna otra estructura de control.
Nótese que esta estructura condicional tiene un solo punto de entrada y un único punto de salida representados por un pequeño círculo con una E y una S respectivamente.
En pseudocódigo se expresará esta estructura de la manera siguiente:

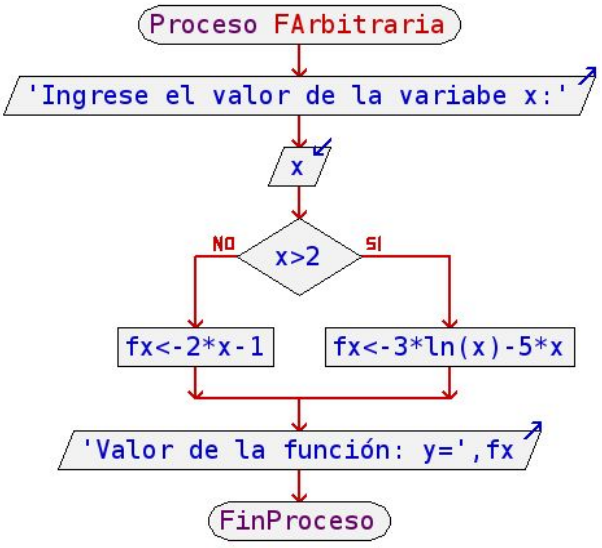
```
Si <exp. lógica> Entonces
     <Acción A>
Sino
     <Acción B>
FinSi
```

Donde las palabras claves **Si** y **FinSi** representan respectivamente la entrada y el fin de la estructura.
Aquí también, si la expresión lógica arroja **Verdadero**, el ejecutante realiza **A** y luego va a **FinSi**; si la expresión lógica arroja **Falso** realiza **B** y pasa a **FinSi**.

Problema Ejemplo: Determinar el valor de la función arbitraria y=f(x) que se indica a la derecha para un valor de la variable x que se lee como dato.

$$y = \begin{cases} 3\ln(x) - 5x & \text{si } x > 2 \\ 2x - 1 & \text{si } x <= 2 \end{cases}$$

Solución en Diagrama de Flujo:



Solución en Pseudocódigo:

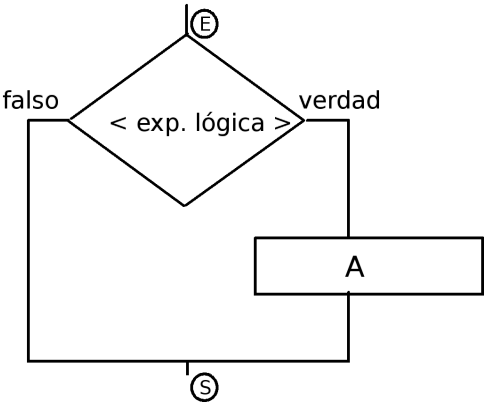
```
Algoritmo FArbitraria
     Escribir 'Ingrese el valor de la variabe x:'
     Leer x
     Si x > 2 Entonces
          fx ← 3*ln(x)-5*x
     Sino
          fx ← 2*x-1
     FinSi
     Escribir 'Valor de la función: y=', fx
FinAlgoritmo
```

Obsérvese que la visualización bidimensional que ofrece el diagrama permite una mejor legibilidad de la lógica del algoritmo respecto del pseudocódigo.

En una estructura condicional, la salida por Verdadero, siempre tendrá al menos una acción para ejecutar; pero es posible que no haya acciones en la salida por Falso de la expresión lógica. Esta variante de la estructura condicional se expresará en pseudocódigo de la manera siguiente:

```
Si <exp.lógica> entonces
     <Acción A>
FinSi
```

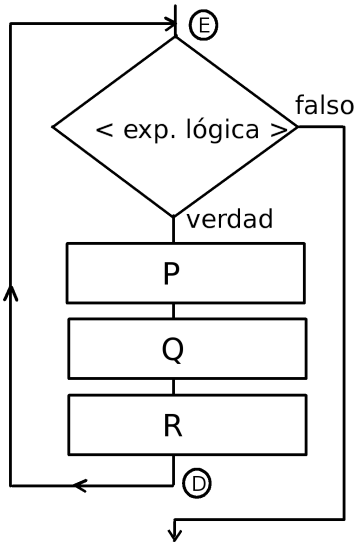
Y en un diagrama de flujo:



Es recomendable respetar la forma gráfica del esquema, tal como se propone. Esto es porque la forma de diamante o rombo se empleará para otras estructuras, y si se altera el esquema, puede resultar confuso el seguimiento de su lógica.

Estructura Repetitiva Mientras-Hacer

Esta estructura permite ejecutar una acción o grupo de acciones en forma reiterada, cierto número de veces, mientras se cumpla una condición. El esquema gráfico que se empleará en los diagramas de flujo será el siguiente:



El pequeño círculo con la **E** indica la entrada o ingreso a la estructura, donde debe evaluarse una expresión lógica. Si el valor obtenido es **Verdadero** el control de ejecución permanece dentro de la estructura y se ejecutan las acciones allí encerradas; en el ejemplo propuesto: **P**, **Q**, y **R** .
El círculo con la **D** señala el punto delimitador del **Mientras** e indica que se debe volver a evaluar la expresión lógica del comienzo.
El valor de verdad que se obtenga determinará si el control de ejecución permanece o sale del esquema.

La sintaxis correspondiente en pseudocódigo es:

```
Mientras <exp. lógica> Hacer
      <acción A>
      <acción B>
      <acción C>
FinMientras
```

El punto **D** del diagrama se corresponde con el delimitador **FinMientras**. En ese lugar se indica al ejecutante que debe volver a observar el valor de verdad de la proposición lógica. Si esta arroja **Verdadero** se vuelven a ejecutar las acciones A, **B**, **C**; si arroja **Falso**, el control escapa de la estructura para pasar a ejecutar la próxima acción después del **FinMientras**.

Como en la estructura condicional, aquí también se tiene un único punto de entrada y un único punto de salida.

Dos características particulares de la estructura repetitiva Mientras:

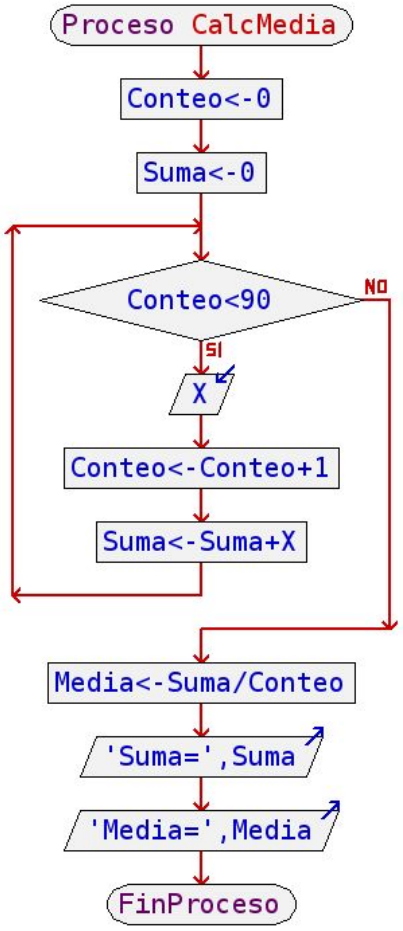
- Es posible que en un algoritmo las acciones encerradas en la estructura nunca lleguen a ejecutarse, si la expresión lógica arroja un falso en la primer evaluación.

- En el grupo de acciones que abarca la estructura es necesaria la presencia de cierta acción, que permita alguna vez, modificar el valor de verdad de la expresión lógica que controla el Mientras. De lo contrario se estaría en presencia de un bucle infinito.

Al disponer de un proceso repetitivo o iterativo en un algoritmo, se pueden plantear soluciones a problemas más complejos o que involucren una gran cantidad de datos. Se verá que los problemas y sus correspondientes algoritmos no difieren en mucho de lo que UD. venía haciendo hasta ahora; solo se incorpora a la secuencia de acciones una estructura que las itere (repita). De este modo se puede lograr la repetición sistemática de un proceso, lo que implica importantes ventajas desde el punto de vista computacional debido a la velocidad con que pueden operar las computadoras.

¿Qué ocurre ahora con la lectura de numerosos datos . Aprovechando el proceso iterativo se puede leer un dato o grupo de datos, realizar sobre ellos el proceso correspondiente, luego repetir la lectura para el segundo dato o grupo de datos, procesarlo y así sucesivamente. Para aclarar esto obsérvese el ejemplo siguiente.

> **Problema Ejemplo**: Leer una serie de datos numéricos correspondientes a las edades de 90 personas. Obtener e informar el valor de la suma y la media (promedio) de dicho conjunto de datos.
>
> **Solución empleando Diagrama de Flujo:**



Solución en pseudocódigo

```
Algoritmo CalcMedia
    Conteo ← 0
    Suma ← 0
    Mientras Conteo < 90 Hacer
        Leer X
        Conteo ← Conteo + 1
        Suma ← Suma + X
    FinMientras
    Media ←  Suma/Conteo
    Escribir 'Suma=', Suma
    Escribir 'Media=', Media
FinAlgoritmo
```

Nótese que la acción `Conteo ← Conteo + 1` permite controlar la estructura al incrementar en una unidad la variable Conteo. En cierto momento Conteo tomará el valor 90 y el control de ejecución escapará del Mientras para pasar a la acción que permite calcular la media.
Los datos se ingresan en la variable x de a uno por vez (90 veces).

Estructuras de Control Adicionales

De acuerdo al Teorema Fundamental de la Programación Estructurada enunciado por Baum y Jacopini serían suficientes las estructuras hasta aquí planteadas para resolver cualquier problema computacional. Pero en diversas situaciones --limitados por tan pocas herramientas-- es posible encontrar diseños de algoritmos algo intrincados, engorrosos, o faltos de claridad.
Por tal motivo, se estudiará el uso de estructuras adicionales, que no son sino variantes de las ya vistas y pueden ayudar a clarificar el diseño de ciertas soluciones.

Estructura Condicional de Selección Múltiple

En varias situaciones se presentará el caso de que la decisión a tomar para bifurcar el flujo o control de ejecución en un algoritmo no se basa en un proposición lógica única con dos posibles alternativas; sino, que los caminos posibles a seguir serán: 3, 4, ..,10 o más.
Tal situación puede resolverse con la Estructura Condicional Si-Entonces.
Se analizará en un ejemplo la resolución de un caso como el descrito.

Ejemplo
Un club deportivo posee N socios. Tiene 5 categorías de asociados: 1, 2, 3, 4 y 5; que corresponden respectivamente a vitalicios, mayores, juveniles cadetes e infantiles. A cada categoría le corresponde abonar una cuota mensual diferente, a excepción de las categorías cadetes e infantiles que pagan igual monto.
Además, los cadetes y juveniles --por este mes-- tienen un descuento del 25%, y el resto de las categorías un 10%. El club desea conocer el monto correspondiente a la recaudación mensual por el abono de cuotas de asociados, suponiendo que abona la totalidad de los mismos.

Datos generales del problema:
- **N**: número de asociados.
- **C1**, **C2**, **C3**, **C4**: monto de cada cuota.

Y por cada socio:
- **Cat**: categoría del socio.

Solución:

```
Algoritmo Club
      C ← 0 ;  Suma ← 0
      Leer N, C1, C2, C3, C4
      Mientras C<N Hacer
            Leer Cat
            C ← C+1
      Si Cat=1 Entonces
            Cuota ← C1*0.90
      Sino
            Si Cat=2 Entonces
                  Cuota ← C2*0.90
            Sino
                  Si Cat=3 Entonces
                        Cuota ← C3*0.75
                  Sino
                        Si Cat=4 Entonces
                              Cuota ← C4*0.75
                        Sino
                              Cuota ← C4*0.90
                        FinSi
                  FinSi
            FinSi
      FinSi
      Suma ← Suma + Cuota
      FinMientras
      Escribir 'Recaudación del mes:', Suma
FinAlgoritmo
```

Como se puede observar en la solución planteada para el ejemplo, el empleo de la estructura condicional Si-Entonces resuelve el caso; pero a pesar de la sencillez del problema, el seguimiento de la lógica es confuso.

Para tal situación se propone una estructura que contempla la posibilidad de establecer una selección entre varias posibilidades clarificado el algoritmo. Esta estructura se denomina Estructura Condicional de Selección Múltiple, y se trata de una generalización de la estructura condicional.
En ella, en lugar de evaluar una condición o expresión lógica, se coteja el valor de cierta variable, llamada variable de control de la estructura, con una lista de valores posibles. El valor determinará cual es la próxima acción a ejecutar.

```
Segun V Hacer
      1:
            <Acción A>
      2:
            <Acción B>
      3:
            <Acción C>
      n:
            <Acción Q>
FinSegun
```

Donde **V** es la variable de control, que debe ser de tipo numérico y además entera positiva sin incluir el cero. Al encontrar esta estructura, el ejecutante (la computadora) realiza lo siguiente:

- Observa el valor de la variable **V**.
- Buscar en la lista de valores propuestos el valor que coincida con el de **V**.
- Si encuentra dicho valor, ejecutar las acciones indicadas para esa opción. Luego, ir al fin de la estructura (**FinSegun**).
- Si el valor asignado a la variable, no coincide con ningún valor de la lista propuesta, salta al **FinSegun** sin ejecutar acciones.

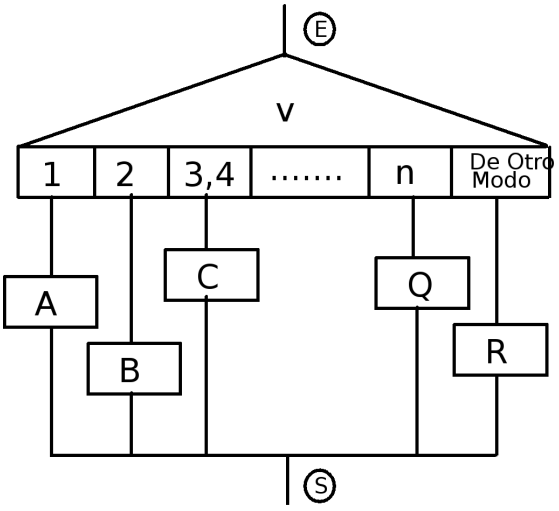
Opcionalmente, pueden plantearse acciones para el caso de que la variable de control V no coincida con ningún valor de la lista. La sintaxis alternativa es la siguiente:

```
Segun V Hacer
      1:
            <Acción A>
      2:
            <Acción B>
      3,4:
            <Acción C>
      n:
            <Acción Q>
      De Otro Modo:
            <Acción R>
FinSegun
```

Si para más de un valor, se debe realizar la misma acción (o grupo de acciones), tales valores pueden agruparse en la misma línea separados por coma. Es el caso de los valores **3** y **4** indicados más arriba al describir la sintaxis.
Para esos dos valores, debe realizarse la misma acción (o grupo de acciones) que se identificó con C.

Nota: A, B, C,...Q, y R pueden representar desde una acción elemental hasta un grupo de acciones o estructuras.

El esquema que se empleará en un diagrama de flujo para la estructura Según es el siguiente:



Al igual que la estructura condicional, se tiene aquí, un único punto de entrada y un único punto de salida que en el gráfico se indican con los pequeños círculos que encierran una **E** y una **S**.
Obsérvese en el mismo caso del club que se usó como ejemplo introductorio a este tema, cómo aplicar la estructura de selección múltiple.

Ejemplo:
Un club deportivo posee N socios. Tiene 5 categorías de asociados: 1, 2, 3, 4 y 5; que corresponden respectivamente a vitalicios, mayores, juveniles cadetes e infantiles. A cada categoría le corresponde abonar una cuota mensual diferente, a excepción de las categorías cadetes e infantiles que pagan igual monto.
Además, los cadetes y juveniles --por este mes-- tienen un descuento del 25%, y el resto de las categorías un 10%. El club desea conocer el monto correspondiente a la recaudación mensual por el abono de cuotas de asociados, suponiendo que abona la totalidad de los mismos.1

Solución:

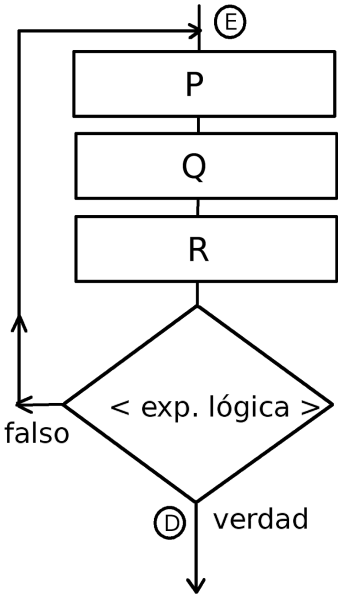
```
Algoritmo Club
    C ← 0 ;  Suma ← 0
    Leer  N, C1, C2, C3, C4
    Mientras C < N Hacer
        Leer Cat
        C ← C + 1
        Segun Cat Hacer
            1: Cuota ← C1 * 0.90
            2: Cuota ← C2 * 0.90
            3: Cuota ← C3 * 0.75
```

```
                4: Cuota ← C4 * 0.75
                5: Cuota ← C4 * 0.90
            FinSegun
            Suma ← Suma + Cuota
        FinMientras
        Escribir 'Recaudación del mes:', Suma
    FinAlgoritmo
```

Compare ambas soluciones del problema, con y sin la estructura de selección múltiple, y podrá observar la diferente legibilidad en favor del último ejemplo donde se ha empleado la estructura Según.

Estructura Iterativa Repetir-Hasta que

Esta herramienta algorítmica adicional, tiene cierta similitud con el Mientras-Hacer, ya que se debe evaluar cierta proposición lógica, cuyo resultado permite continuar o escapar del bucle iterativo. Su esquema gráfico se indica abajo:



En presencia de esta estructura, el ejecutante del algoritmo, ingresa a la estructura, resuelve las acciones indicadas en los bloques P, Q, y R; entonces evalúa la expresión lógica: si esta es falsa, vuelve a repetir las acciones. Así sucesivamente Hasta que la expresión lógica arroje un resultado verdadero.

Su sintaxis para el pseudocódigo es:

```
Repetir
     <acción P>
     <acción Q>
     <acción R>
Hasta Que <exp. lógica>
```

Al igual que el Mientras, la estructura **Repetir** posee algunas características particulares:

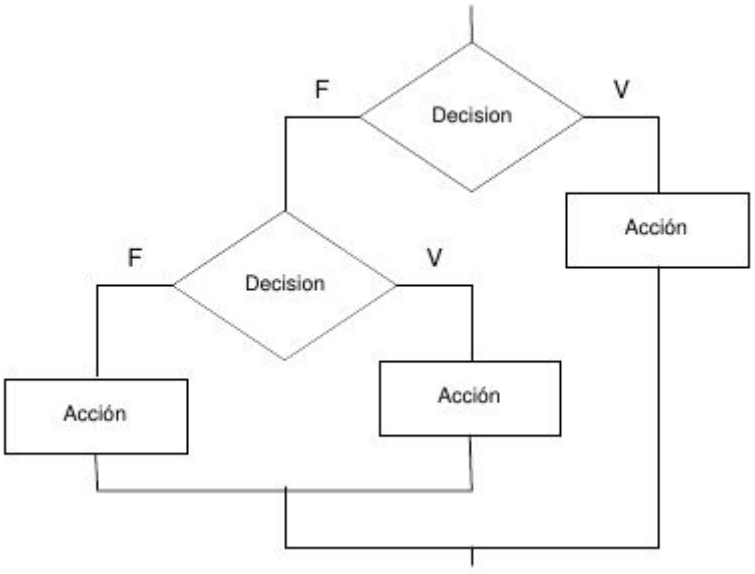
- Si la expresión lógica arroja un **Falso** en la primer evaluación, las acciones abarcadas por la estructura ya habrán sido ejecutadas. Por tanto --siempre-- las acciones de esta estructura serán realizadas al menos una vez.
- En el grupo de acciones que abarca la estructura es necesaria la presencia de cierta acción, que permita alguna vez, modificar el valor de verdad de la expresión lógica que controla el **Repetir**. De lo contrario se estaría en presencia de un bucle infinito.

Observación: en las dos estructuras iterativas o de repetición estudiadas, se puede notar que no es necesario conocer el número de iteraciones a realizar. Es común hallar problemas donde la proposición lógica que controla a ambas estructuras, sea dependiente de alguno de los datos de entrada del algoritmo.

Estructuras de control anidadas

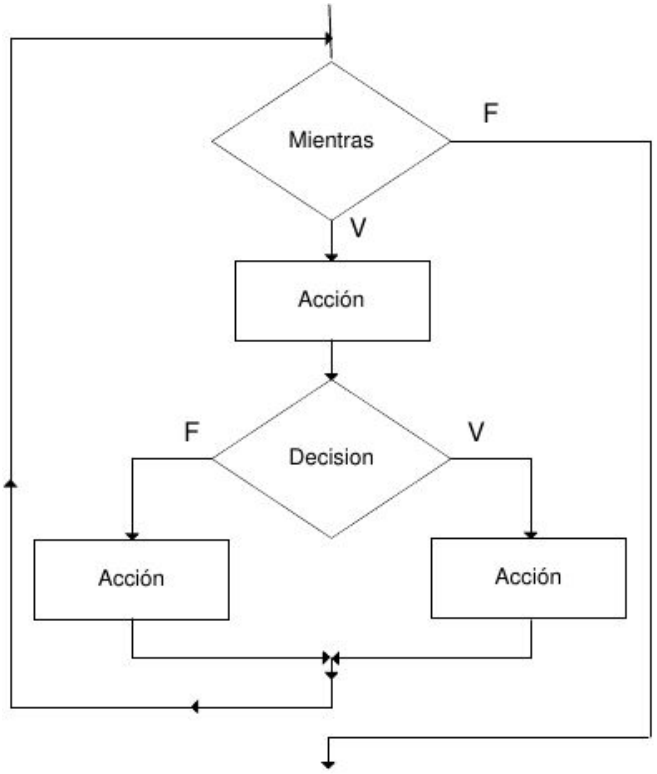
En el diseño de algoritmos, es usual el empleo de estructuras lógicas de control en situaciones más complejas, las cuales se resuelven combinando las estructuras básicas y adicionales que se desarrollaron en esta unidad. Obsérvense algunos casos:

Caso 1



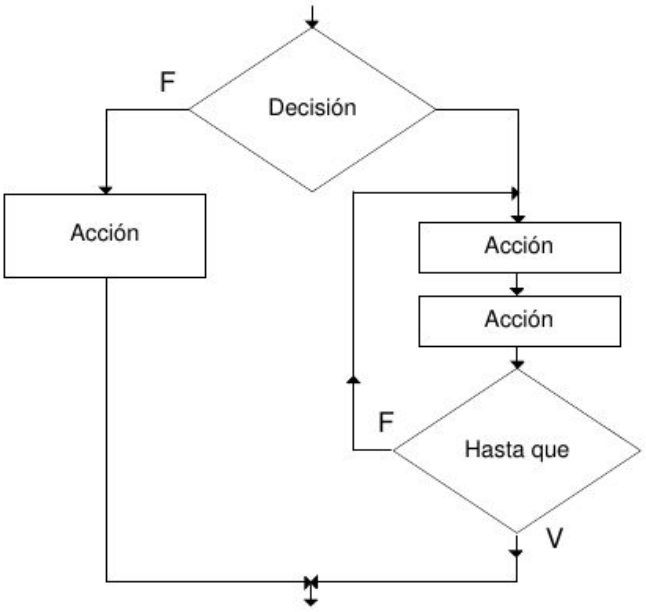
Aquí, se presenta una estructura condicional con una acción simple en la salida por VERDAD. Por FALSO, tiene otra estructura condicional anidada.
Ejemplo: observe la solución propuesta para el problema del Club que se usó para introducir el tema de la estructura *Segun*

Caso 2



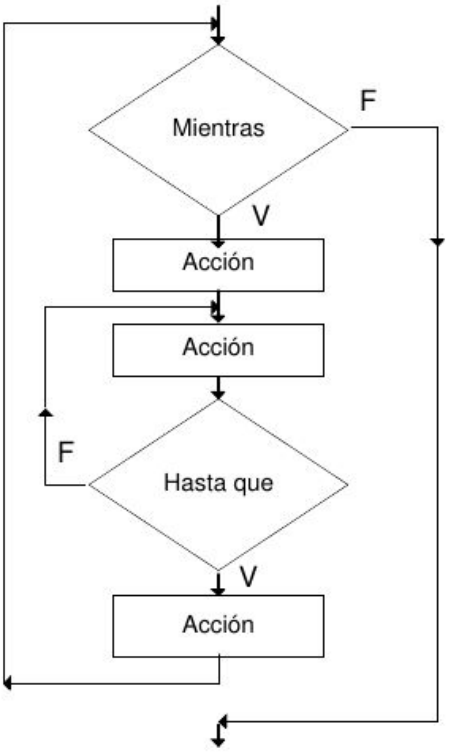
El diagrama plantea una estructura iterativa MIENTRAS, la cual encierra una estructura condicional.

Caso 3



En este caso, se tiene un *REPETIR* anidado dentro de la salida por *FALSO* de la estructura condicional.

Caso 4



Aquí una estructura **Mientras** encierra acciones de estructura secuencial y además una estructura iterativa **Repetir-Hasta que**.

IMPORTANTE: en todos los casos, observe que cada estructura encerrada o anidada en otra, tiene su punto de entrada y su punto de salida, dentro de la estructura anidante.

Síntesis

1. Las estructuras de control permiten resolver algorítmicamente problemas más complejos. Con ellas es posible tomar decisiones, iterar o repetir grupos de acciones.
2. Cada estructura de control tiene un único punto de entrada y un único punto de salida. Esto es propio del modelo o paradigma de la programación estructurada.
3. La estructura condicional Si-Entonces permite tomar un camino u otro en la secuencia de ejecución, en base al valor de verdad de una proposición lógica.
4. La estructura iterativa Mientras-Hacer permite evaluar una expresión lógica al inicio y si su valor de verdad arroja verdadero se ejecutan las acciones incluidas en la estructura; luego se vuelve a evaluar la expresión lógica y así sucesivamente. Las acciones incluidas en esta estructura se podrán ejecutar 0 o más veces.
5. La estructura de selección múltiple Según-Hacer permite ejecutar una o más acciones selectivamente en base al valor de una variable de control que es evaluada al inicio.
6. La estructura iterativa Repetir-Hasta Que actúa similarmente a la estructura Mientras-Hacer solo que la expresión lógica se evalúa al final; si esta arroja verdadero se abandona la estructura, y se continúa iterando en caso de arrojar falso.
7. A medida que lo problemas se hacen más complejos es usual tener que combinar estructuras encerrando o anidando unas dentro de otras. En estos casos recordar que cada estructura anidada debe estar completamente incluida dentro de la estructura exterior o anidante.