TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE NUEVO LAREDO

"Con la Ciencia por la Humanidad"

Introducción a la ISC y al Diseño Algorítmico

Curso propedéutico

Instructor:

Bruno López Takeyas

bruno.lt@nlaredo.tecnm.mx

7.- Control de flujo

- **7.1 Instrucciones algorítmicas básicas**
- **7.2 Representación de instrucciones en un diagrama de flujo**
- **7.3 Algoritmos secuenciales**
- **7.4 Estructuras selectivas**
- **7.5 Estructuras iterativas, repetitivas o cíclicas**
- **7.6 Ejemplo**

Después de haber analizado las diferentes tipos de expresiones que puede evaluar una computadora, ahora se revisa la forma de diseñar algoritmos. En este capítulo se atienden las instrucciones algorítmicas básicas y el diseño de algoritmos utilizando estructuras selectivas y cíclicas.

Instrucciones algorítmicas básicas

Existen tres instrucciones básicas consideradas para el diseño de algoritmos:

- *Entrada de datos.*
- *Salida de información.*
- *Asignación.*

Entrada de datos

Consiste en obtener un dato de un dispositivo de entrada (como el teclado) y almacenarlo en una variable. En general, la acción de ingresar un dato a una **variable** se expresa en el pseudocódigo mediante la palabra **LEER** (Fig. 8.1).

```
LEER variable
P. ejem.
LEER Estatura
LEER ClaveEmpleado, NombreEmpleado
```

Fig. 8.1. La instrucción de entrada de datos representada en pseudocódigo.

Símbolos de entrada de datos

Existen varios símbolos para representar la instrucción de la entrada de datos en un diagrama de flujo (Fig. 8.2).

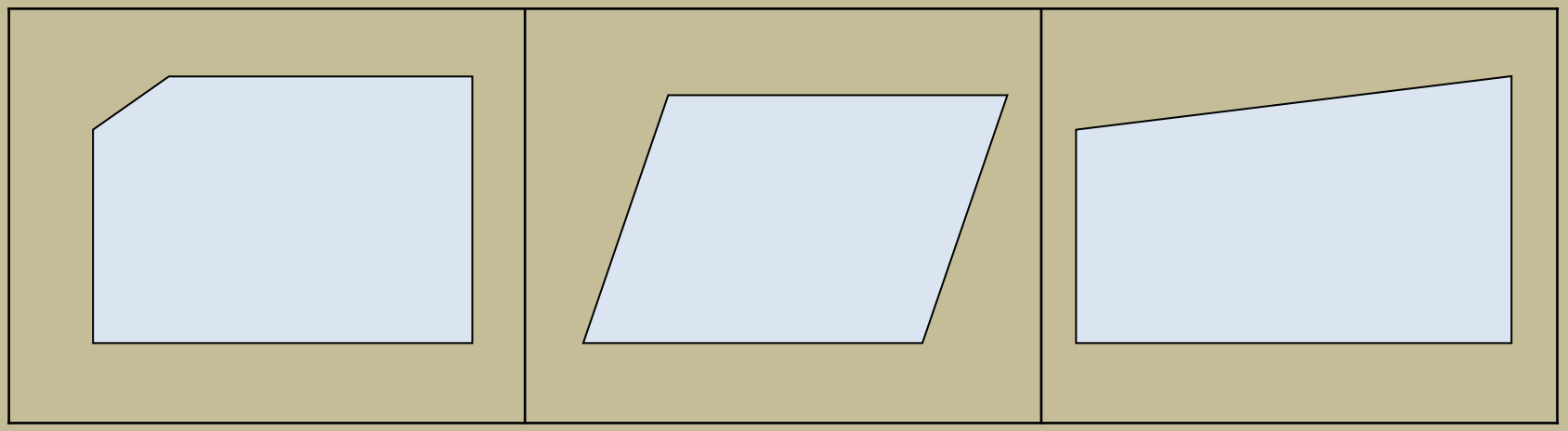


Fig. 8.2. Símbolos de un diagrama de flujo para la instrucción LEER.

Ejemplo de entrada de datos

La Fig. 8.3. muestra la representación de la entrada de datos cuando se desea capturar el radio de una circunferencia en un algoritmo.

Pseudocódigo	***Diagrama de flujo***
`LEER Radio`	

Fig. 8.3. Ejemplo de representación de la entrada de datos.

Salida de datos o información

Consiste en mostrar el valor de una variable, constante o expresión en un dispositivo de salida (como la pantalla). En general, la acción de mostrar un dato se expresa en el pseudocódigo mediante la palabra **IMPRIMIR** (Fig. 8.4).

```
IMPRIMIR variable
P. ejem.
IMPRIMIR Precio
IMPRIMIR “Instituto Tecnológico de Nuevo Laredo”
IMPRIMIR “Precio = “;x
```

Fig. 8.4. La instrucción de salida representada en pseudocódigo.

Símbolos de salida de datos o información

La Fig. 8.5 muestra el símbolo del diagrama de flujo que representa la instrucción de salida de datos o información. Cuando se desea mostrar un mensaje en la pantalla, basta con ponerlo entre comillas. Si no se colocan las comillas, entonces imprime el valor de la variable indicada.

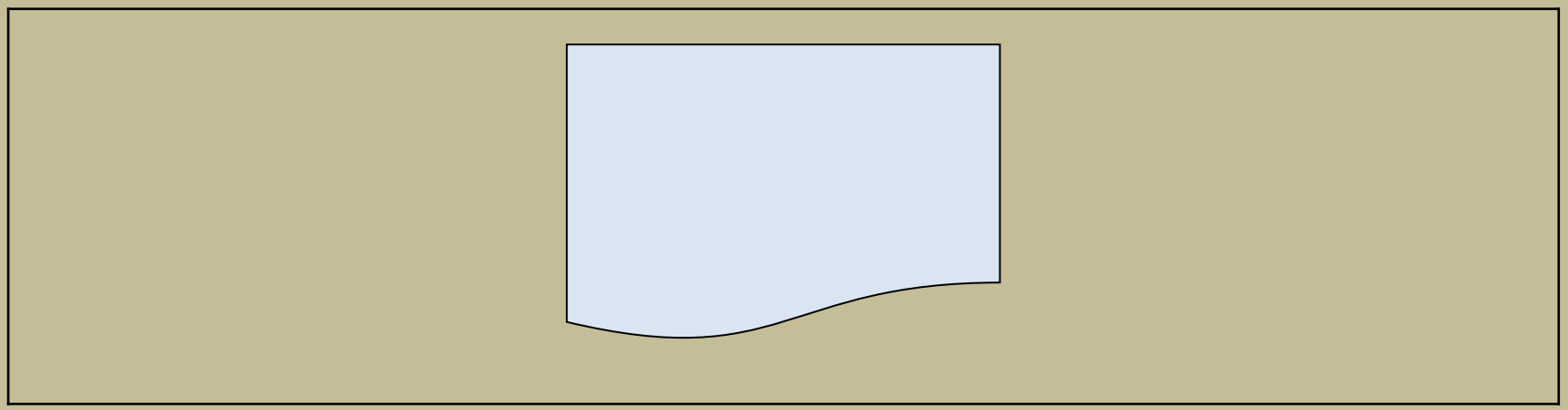


Fig. 8.5. Símbolo de la salida de datos o información en un diagrama de flujo

Ejemplo de salida de datos o información

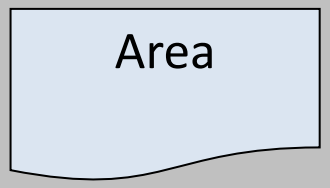
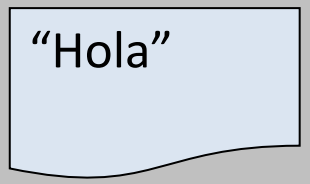
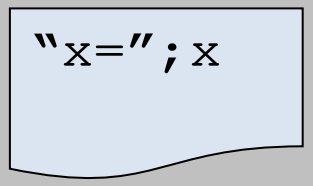
Pseudocódigo	*Diagrama de flujo*
`IMPRIMIR Area`	
`IMPRIMIR "Hola"`	
`IMPRIMIR "x=";x`	

Fig. 8.6. Ejemplos de representación de la salida de datos o información.

Asignación

Consiste en almacenar en una **variable** el valor de una **expresión**. La expresión puede ser una simple variable o una combinación de variables, literales y operadores. La asignación se expresa en el pseudocódigo como lo muestra la Fig. 8.7.

```
variable = expresión
P. ejem.
Edad = 26
x = z * 3
Bandera = x>3 OR y<=8
```

Fig. 8.7. La instrucción de asignación.

Asignación

En la representación de una asignación, la variable receptora del valor siempre debe colocarse al lado izquierdo del símbolo de asignación (=), de lo contrario se produce un error. En un diagrama de flujo, la asignación se representa mediante un rectángulo (Fig. 8.8).

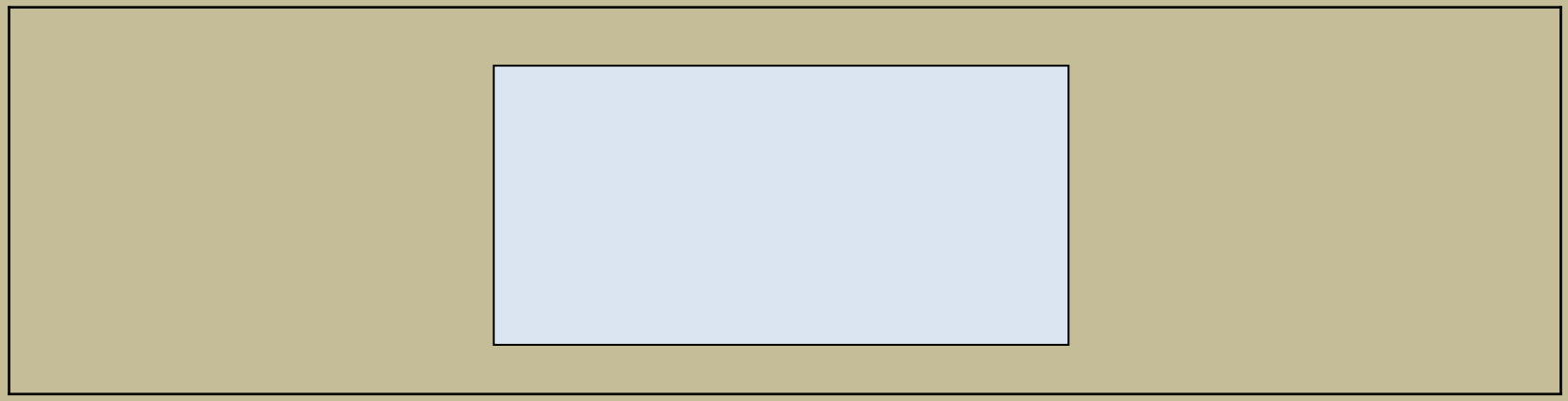


Fig. 8.8. Símbolo de la asignación en un diagrama de flujo.

Ejemplo de asignación

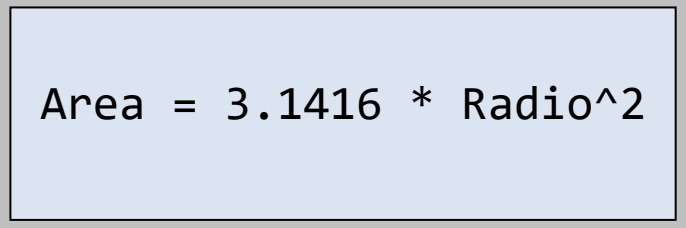
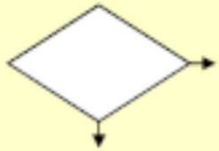
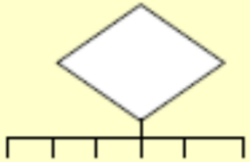
Pseudocódigo	*Diagrama de flujo*
`Area = 3.1416 * Radio^2`	

Fig. 8.9. Ejemplo de representación de la asignación.

Representación de instrucciones en un diagrama de flujo

Un diagrama de flujo es una de las técnicas de representación de algoritmos más usada que utiliza los símbolos estándar y que tiene los pasos del algoritmo escritos en dichos símbolos unidos por flechas, denominadas líneas de flujo que indican la secuencia. La Fig. 8.10 muestra los símbolos principales en el diseño de diagramas de flujo.

Símbolo	Función
	Terminal (representa el comienzo y final de un programa. Puede representar también una interrupción necesaria en un programa.
	Teclado (se utiliza para capturar datos desde el teclado)
	Proceso (cualquier tipo de operación que pueda originar cambio de valor, formato o posición de la información almacenada en memoria, operaciones aritméticas, de transferencia, etc.)
	Decisión (indica operaciones lógicas o de comparación de datos y en función del resultado de la misma determina cuál de los distintos caminos alternativos del programa se debe seguir; normalmente tiene dos salidas: V o F).
	Decisión múltiple (en función del resultado de la comparación se seguirá uno de los diferentes caminos de acuerdo con dicho resultado).
	Conector (sirve para enlazar dos partes cualesquiera de un diagrama a través de un conector en la salida y otro conector en la entrada. Se refiere a la conexión en la misma página del diagrama).
	Indicador de dirección o línea de flujo (indica el sentido de ejecución de las operaciones).
	Línea conectora (sirve de unión entre dos símbolos)
	Conector (conexión entre dos puntos situados en páginas diferentes)

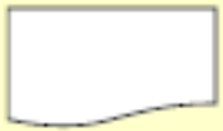
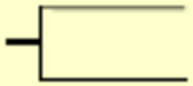
	Impresora (se utiliza cuando se desea desplegar datos)
	Comentarios (se utiliza para añadir comentarios clasificadores a otros símbolos del diagrama de flujo. Se pueden dibujar a cualquier lado del símbolo).
	Llamada a subrutina o aun proceso determinado (una subrutina es un módulo independiente del programa principal que recibe una entrada procedente de dicho programa, realiza una tarea determinada y regresa, al terminar, al programa principal).
	Entrada/Salida (cualquier tipo de introducción de datos en la memoria desde los periféricos. También se utilizan para la manipulación de datos en los archivos).
	Apertura y Cierre de archivos
	Ciclos (utilizado en estructuras repetitivas llamados ciclos; p. ejem. ciclos `while`, `do-while`)
	Ciclos (utilizado en procesos repetitivos de tipo `for`).

Fig. 8.10. Principales símbolos de los diagramas de flujo.

Algoritmos secuenciales

Se conoce como algoritmos secuenciales a aquellos que siguen de manera lineal la secuencia de pasos durante su ejecución, es decir, carecen de bifurcaciones o decisiones que pudieran alterar la ruta de ejecución.

Para representar cada instrucción en un pseudocódigo, simplemente se enumera consecutivamente cada una de las instrucciones, mientras que en un diagrama de flujo, la secuencia de pasos se representa mediante flechas que conectan los símbolos.

La Fig. 8.11 muestra un ejemplo de un algoritmo secuencial que recibe como entrada el valor del radio de una circunferencia, calcula su área e imprime el resultado.

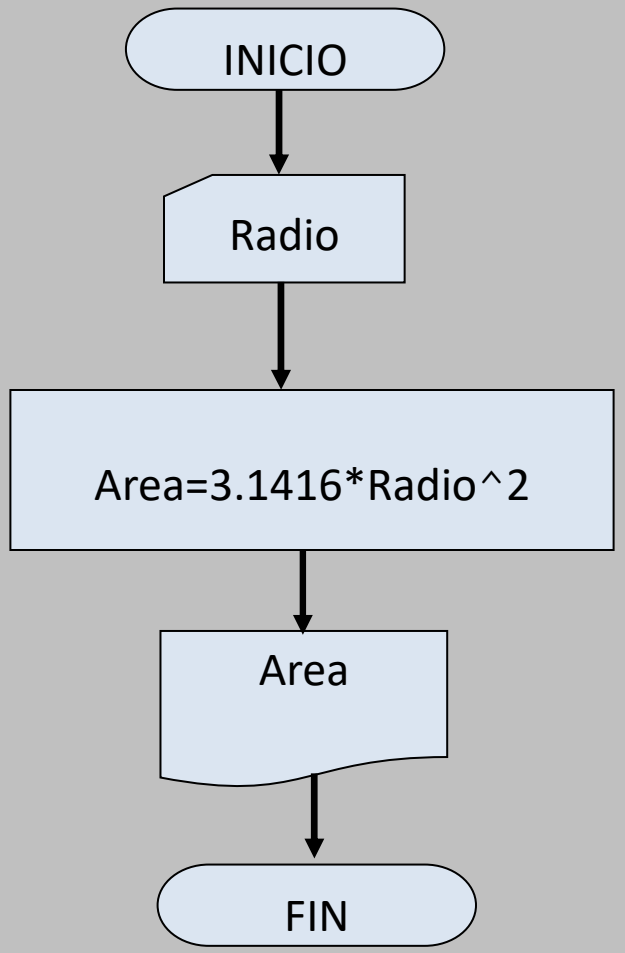
Pseudocódigo	Diagrama de flujo
1.- INICIO 2.- LEER Radio 3.- Area = 3.1416 * Radio^2 4.- Imprimir Area 5.- FIN	

Fig. 8.11. Ejemplo de un algoritmo secuencial.

Estructuras selectivas

Para que una computadora “tome decisiones” requiere de algoritmos que contengan expresiones lógicas y estructuras selectivas. Las estructuras selectivas evalúan una expresión lógica y dependiendo del resultado booleano, toman una decisión para cambiar la ruta de ejecución de un algoritmo. Para implementar una estructura selectiva se requiere de una expresión lógica y el conjunto de instrucciones a ejecutar cuando el resultado de dicha expresión sea verdadero o las instrucciones a ejecutar si es falso. Existen tres tipos de estructuras selectivas dependiendo de las posibles trayectorias a seguir por el algoritmo:

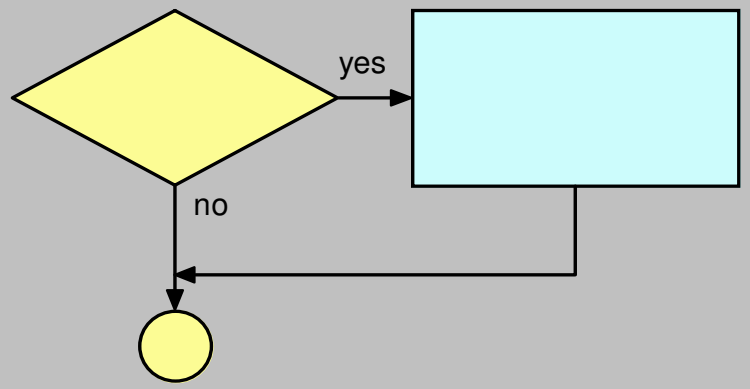
- *Simple (Si-Entonces)*
- *Doble (Si-Entonces-Sino)*
- *Múltiple (SI-Caso)*

Estructura selectiva simple (Si-entonces)

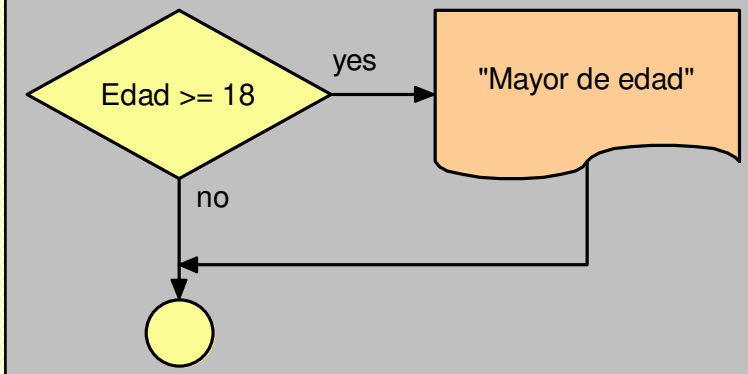
Este tipo de estructuras selectivas solamente tienen una posible trayectoria que se ejecuta si el resultado de la expresión lógica es verdadero; es decir, si el resultado de la expresión lógica es falso, entonces no ejecuta ninguna instrucción.

Para representar esta estructura selectiva mediante pseudocódigo se usa la instrucción SI-ENTONCES (*IF* en inglés) y el diagrama de flujo utiliza el símbolo de un rombo (Fig. 8.12).

Estructura selectiva simple (Si-entonces)

Pseudocódigo	*Diagrama de flujo*
SI *expresión_lógica* ENTONCES …… {FIN DE LA CONDICIÓN}	

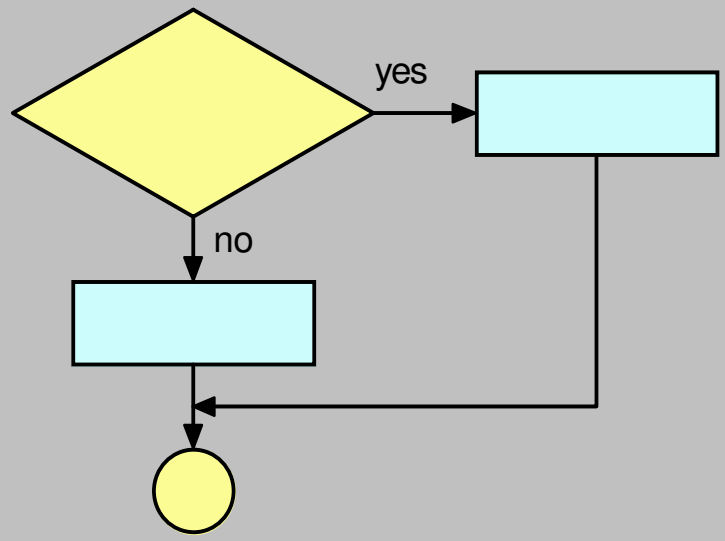
Ejemplo de estructura selectiva simple

Pseudocódigo	*Diagrama de flujo*
SI *Edad>=18* ENTONCES IMPRIMIR "Mayor de edad" {FIN DE LA CONDICIÓN}	

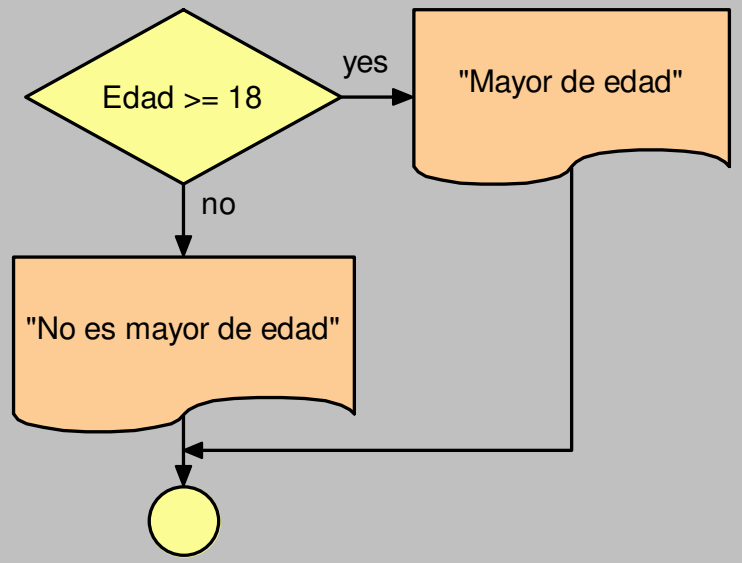
Estructura selectiva doble (Si-Entonces-Sino)

Este tipo de estructuras selectivas tienen dos posibles trayectorias que se ejecutan dependiendo del resultado de la expresión lógica evaluada (verdadero o falso). Para representar esta estructura selectiva mediante pseudocódigo se usa la instrucción SI-ENTONCES-SINO (*IF-ELSE* en inglés) y el diagrama de flujo también utiliza el símbolo de un rombo, sólo que ahora tiene las dos posibles alternativas (Fig. 8.14).

Estructura selectiva doble (Si-Entonces-Sino)

Pseudocódigo	*Diagrama de flujo*
SI *expresión_lógica* ENTONCES ..... SINO ..... {FIN DE LA CONDICIÓN}	

Ejemplo de estructura selectiva doble

Pseudocódigo	*Diagrama de flujo*
SI *Edad>=18* ENTONCES IMPRIMIR "Mayor de edad" SINO IMPRIMIR "No es mayor de edad" {FIN DE LA CONDICIÓN}	

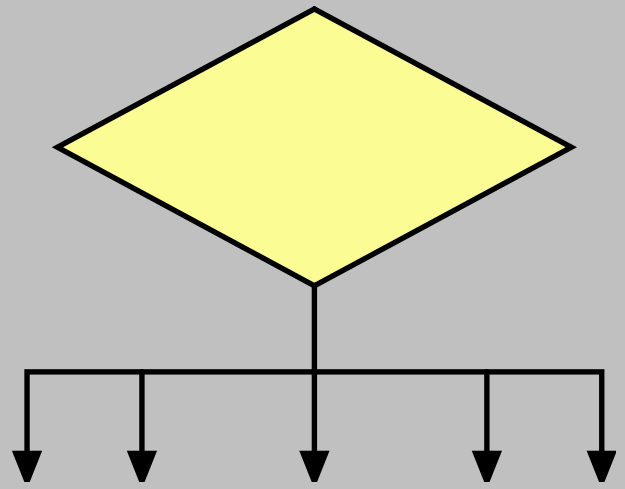
Estructura selectiva múltiple

Este tipo de estructura selectiva es útil cuando se desea comparar el valor de una variable con un conjunto de casos y determinar si es igual que alguno de ellos. Dependiendo del resultado de la comparación, el algoritmo seguirá uno de los diferentes caminos.

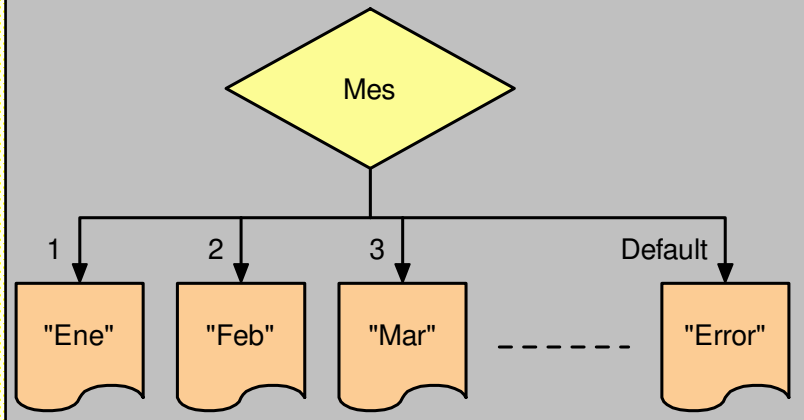
Para representar esta estructura selectiva mediante pseudocódigo se usa la instrucción SI-CASO (*SWITCH-CASE* en inglés) y el diagrama de flujo también utiliza el símbolo de un rombo, sólo que ahora tiene todas las posibles alternativas (Fig. 8.16).

Estructura selectiva múltiple

Pseudocódigo	*Diagrama de flujo*

SI *expresión*
  CASO c1: …
  CASO c2: …
  CASO c3: …
  DEFAULT: …
{FIN DE LA CONDICIÓN}
	

Ejemplo de estructura selectiva múltiple

Pseudocódigo	*Diagrama de flujo*
SI *Mes* CASO 1:IMPRIMIR "Ene" CASO 2:IMPRIMIR "Feb" CASO 3:IMPRIMIR "Mar" DEFAULT: IMPRIMIR "Error" {FIN DE LA CONDICIÓN}	

Estructuras iterativas, repetitivas o cíclicas

En numerosas ocasiones se requiere que un algoritmo ejecute en varias ocasiones la misma instrucción, conjunto de instrucciones o un mismo proceso, en este caso, sería poco elegante escribir varias veces el proceso a ejecutar; en lugar de esto, se utilizan estructuras iterativas, repetitivas o cíclicas (ciclos), que repiten una secuencia de instrucciones un determinado número de veces (iteraciones). La cantidad de iteraciones puede ser fija (determinada por el analista) o variable (en función de algún dato dentro del algoritmo).

Componentes de las estructuras iterativas, repetitivas o cíclicas

Las estructuras cíclicas se componen de 4 partes:

Inicialización: Para implementar una estructura cíclica se debe contar el número de iteraciones a través de una variable de control del ciclo, la cual debe ser inicializada con el valor adecuado antes de empezar a iterar.

Condición de salida: Es muy importante definir cuándo se terminan las iteraciones a través de una condición de salida, la cual se implementa utilizando una expresión lógica que contiene la variable de control del ciclo. Si no se define correctamente la condición de salida, entonces se corre el riesgo de que la estructura cíclica actúe indefinidamente provocando estancamiento del algoritmo (comúnmente se dice que se "cicla" el algoritmo).

Paso: Se conoce con este nombre al conteo de las iteraciones. Regularmente, cada vez que se ejecuta una iteración, se incrementa la variable de control del ciclo (la cual actúa como un contador); sin embargo, algunos algoritmos utilizan el paso del ciclo con acumuladores.

Cuerpo: Es el conjunto de instrucciones o procesos que se ejecutarán repetidamente dentro de la estructura cíclica.

Tipos de estructuras iterativas, repetitivas o cíclicas

De manera general, existen 3 tipos de estructuras cíclicas:

- *REPETIR-DESDE-HASTA (FOR)*
- *MIENTRAS-HACER (WHILE)*
- *HACER-MIENTRAS (DO-WHILE)*

Ciclo Repetir-Desde-Hasta (*for*)

Esta estructura cíclica tiene una característica particular: se conoce el inicio y el fin de las iteraciones y las controla a través de un contador. Esta estructura repetirá el cuerpo del ciclo tantas veces como se le indique. A partir de un valor inicial de la variable de control se va incrementando en cada iteración hasta llegar a un valor determinado, momento en que se sale de la estructura cíclica (Fig. 8.18).

Ciclo Repetir-Desde-Hasta (*for*)

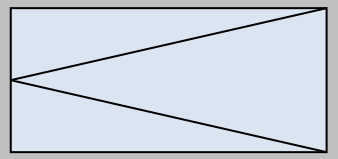
Pseudocódigo	Diagrama de flujo
REPETIR CON *VariableControl* DESDE *ValorInicial* HASTA *ValorFinal* PASO *ValorPaso* …… {FIN DEL CICLO}	

Fig. 8.18. Representación de una estructura cíclica Repetir-Desde-Hasta.

Ejemplo de Ciclo Repetir-Desde-Hasta (*for*)

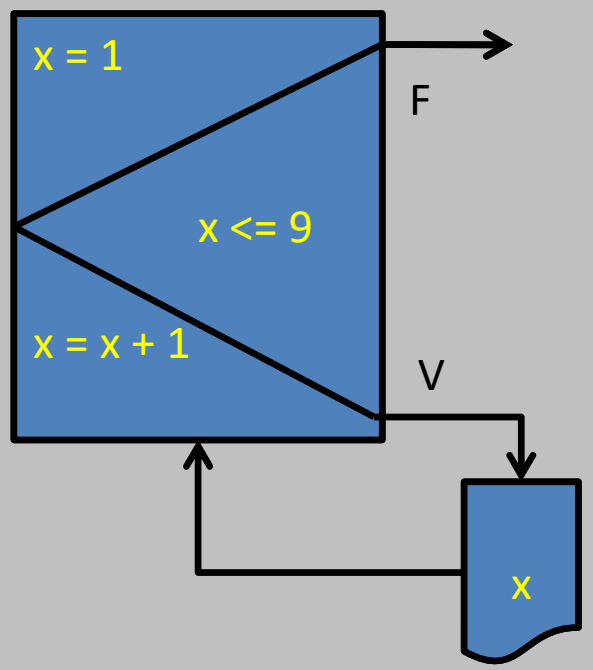
Pseudocódigo	*Diagrama de flujo*
1.- INICIO 2.- REPETIR CON x DESDE 1 HASTA 9 PASO 1 2.1.- IMPRIMIR x 3.- {FIN DEL CICLO DEL PASO 2} 4.- FIN	

Fig. 8.19. Ejemplo de un algoritmo secuencial.

Ciclo Mientras-Hacer (*while*)

La principal característica de esta estructura cíclica es que se itera mientras se cumple una condición, es decir, repetirá el cuerpo del ciclo mientras la expresión lógica de la condición de salida sea verdadera (Fig. 8.20).

Pseudocódigo	*Diagrama de flujo*
MIENTRAS *ExpresiónLógica* HACER …… {FIN DEL CICLO MIENTRAS}	

Fig. 8.20. Representación de una estructura cíclica Mientras-Hacer.

Ejemplo de ciclo Mientras-Hacer (*while*)

Pseudocódigo	*Diagrama de flujo*

```
1.- INICIO
2.- i = 1
3.- MIENTRAS i<=10 HACER
    3.1.- IMPRIMIR i
    3.2.- i = i + 1
4.- {FIN DEL CICLO DEL PASO 3}
5.- FIN
```

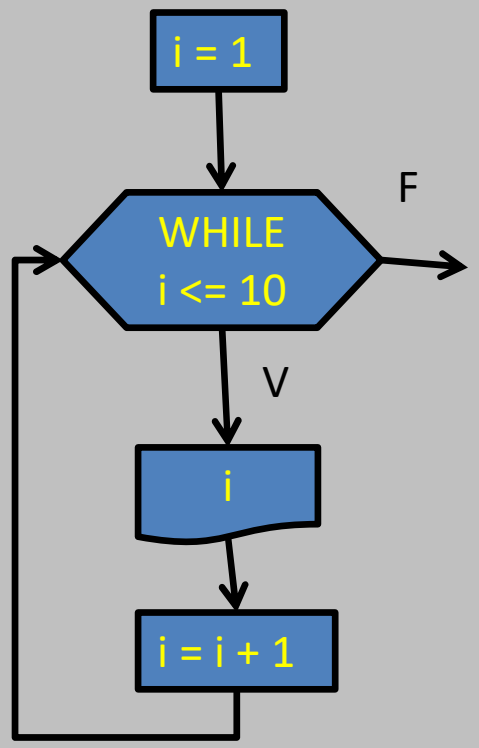


Fig. 8.21. Ejemplo de un algoritmo secuencial.

Consideraciones del ciclo Mientras-Hacer (*while*)

Cuando se implementa una estructura cíclica Mientras-Hacer se debe considerar lo siguiente:

- ***La inicialización de la variable de control debe colocarse fuera del cuerpo del ciclo (antes de la condición), ya que de lo contrario, la estructura se comporta como un ciclo indefinido (se cicla el algoritmo).***
- ***A diferencia de un ciclo Repetir-Desde-Hasta, en una estructura cíclica Mientras-Hacer el incremento de la variable de control debe formar parte del cuerpo del ciclo.***
- ***Al ejecutarse, primero se evalúa la condición y dependiendo del resultado booleano, entonces se ejecuta el cuerpo del ciclo. Debido a esto, el cuerpo del ciclo se puede ejecutar 0 o más veces, es decir, si la primera vez que se evalúa la condición de salida resulta falsa, entonces no se ejecuta el cuerpo del ciclo y directamente se dirige a la salida del mismo.***

Ciclo Hacer-Mientras (*do-while*)

La principal característica de esta estructura cíclica es que también se itera mientras se cumple una condición, es decir, repetirá el cuerpo del ciclo mientras la expresión lógica de la condición de salida sea verdadera; sin embargo, la evaluación de la expresión lógica se hace después de haber ejecutado el cuerpo del ciclo (Fig. 8.22).

Pseudocódigo	***Diagrama de flujo***

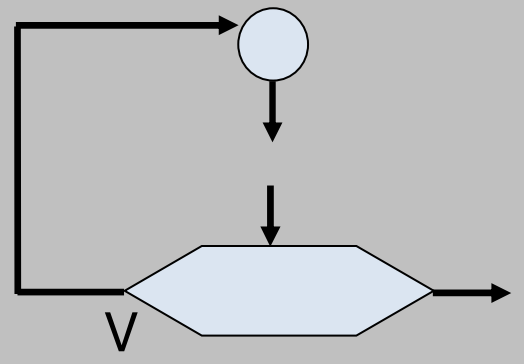
`HACER`
`.....`
`MIENTRAS` *`ExpresiónLógica`*
	

Fig. 8.22. Representación de una estructura cíclica Hacer-Mientras.

Ejemplo de ciclo Hacer-Mientras (*do-while*)

Pseudocódigo	*Diagrama de flujo*

```
1.- INICIO
2.- i = 0
3.- HACER
      3.1.- IMPRIMIR i
      3.2.- i = i + 1
4.- MIENTRAS i<=10
5.- FIN
```

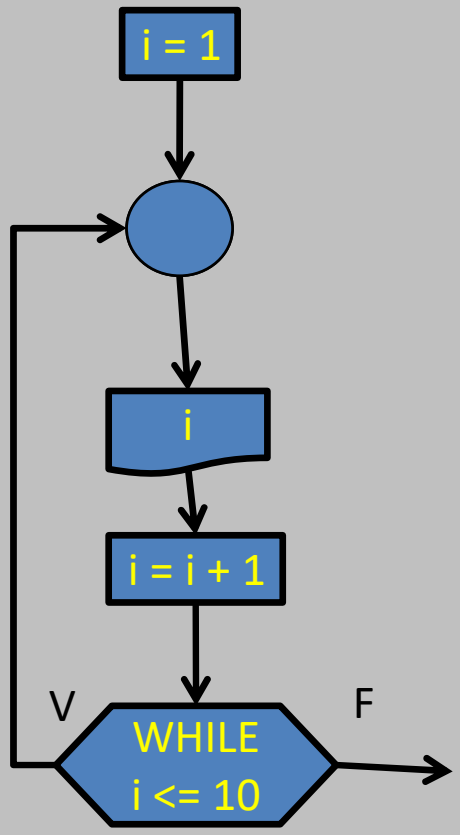


Fig. 8.23. Ejemplo de una estructura cíclica Hacer-Mientras.

Consideraciones del ciclo Hacer-Mientras (*do-while*)

Cuando se implementa una estructura cíclica Hacer-Mientras se debe considerar lo siguiente:

- ***La inicialización de la variable de control debe colocarse fuera del cuerpo del ciclo, ya que de lo contrario, la estructura se comporta como un ciclo indefinido (se cicla el algoritmo).***
- ***A diferencia de un ciclo Repetir-Desde-Hasta y de manera semejante a un ciclo Mientras-Hacer, en una estructura cíclica Hacer-Mientras el incremento de la variable de control debe formar parte del cuerpo del ciclo.***
- ***Al recorrer esta estructura cíclica, primero se ejecuta el cuerpo del ciclo y después se evalúa la condición. Debido a esto, el cuerpo del ciclo se puede ejecutar 1 o más veces, es decir, si la primera vez que se evalúa la condición de salida resulta falsa, entonces el cuerpo del ciclo ha sido ejecutado al menos una vez.***

Semejanzas y diferencias entre los ciclos

Semejanzas

Los tres tipos de ciclos realizan las mismas acciones, sin embargo, puesto que tienen estructura lógica diferente, entonces lo realizan de manera diferente; pero todos coinciden en tener los cuatro componentes de los ciclos (inicialización, condición de salida, paso y cuerpo).

Diferencias

La estructura lógica de las estructuras cíclicas es diferente, lo que origina diferencias en la secuencia de ejecución del ciclo. Las principales diferencias son:

El ciclo Repetir-Desde-Hasta tiene la inicialización y paso de la variable de control dentro de la definición de la estructura cíclica, es decir, no forman parte del cuerpo del ciclo; mientras que los ciclos Mientras-Hacer y Hacer-Mientras, deben colocar la inicialización de la variable de control antes de iniciar el ciclo y el paso debe ser colocado dentro del cuerpo, formando parte de las iteraciones del ciclo.

Las estructuras cíclicas Mientras-Hacer y Hacer-Mientras son muy parecidas, solo difieren en la ubicación de la condición de salida (una la hace al principio del ciclo y la otra al final). Se puede decir que si gráficamente se gira con 180° a una de ellas, se obtiene la otra.

¿Cómo saber qué ciclo utilizar?

Si un algoritmo requiere implementar una estructura cíclica, se recomienda tener en cuenta las siguientes consideraciones de uso:

Repetir-Desde-Hasta.- Se recomienda este ciclo cuando se conoce exactamente el inicio y el final de las iteraciones o bien la cantidad de veces que debe iterar el algoritmo. Es la estructura cíclica más sencilla de diseñar e implementar. Este ciclo no se recomienda cuando las instrucciones del cuerpo del ciclo pudiesen afectar el resultado de la condición de salida, para ello, se recomienda ya sea el ciclo Mientras-Hacer o el ciclo Hacer-Mientras.

Mientras-Hacer.- Se sugiere el uso de esta estructura cíclica cuando un algoritmo requiere evaluar la condición de salida antes de ejecutar el cuerpo del ciclo; es decir, cuando se desea que el cuerpo del ciclo se ejecute cero o más veces.

Hacer-Mientras.- Se recomienda el uso de esta estructura cíclica cuando un algoritmo necesita que el cuerpo del ciclo se ejecute al menos una vez.

Prácticas

- Descargue del sitio web:

`https://nlaredo.tecnm.mx/takeyas/LibroISC`

- *Práctica 7.1.- Instrucciones algorítmicas selectivas*
- *Práctica 7.2.- Ciclos*

Tarea

Resuelva en el sitio web

https://nlaredo.tecnm.mx/takeyas/LibroISC

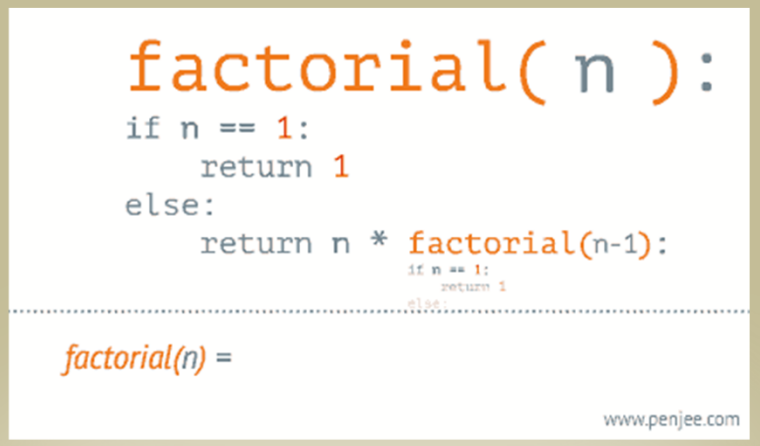
- ❑ **Cuestionario 7.1**

Ejercicio resuelto

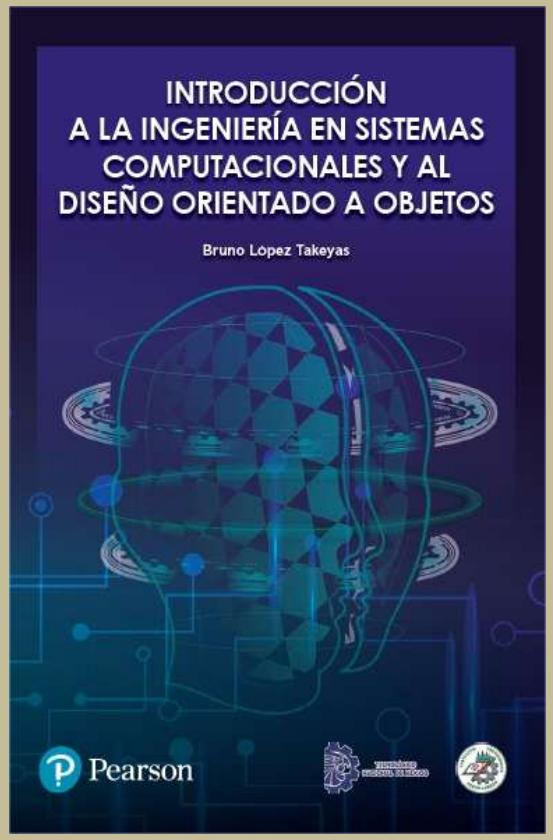
Consulte en el sitio web

https://nlaredo.tecnm.mx/takeyas/LibroISC

- **Ejercicio resuelto 7.6.- Análisis y diseño para calcular el factorial de un número**



Fuentes de información:

- López Takeyas, Bruno. (2019) “Introducción a la Ingeniería en Sistemas Computacionales y al diseño orientado a objetos”. Editorial Pearson.

- https://nlaredo.tecnm.mx/takeyas/LibroISC/