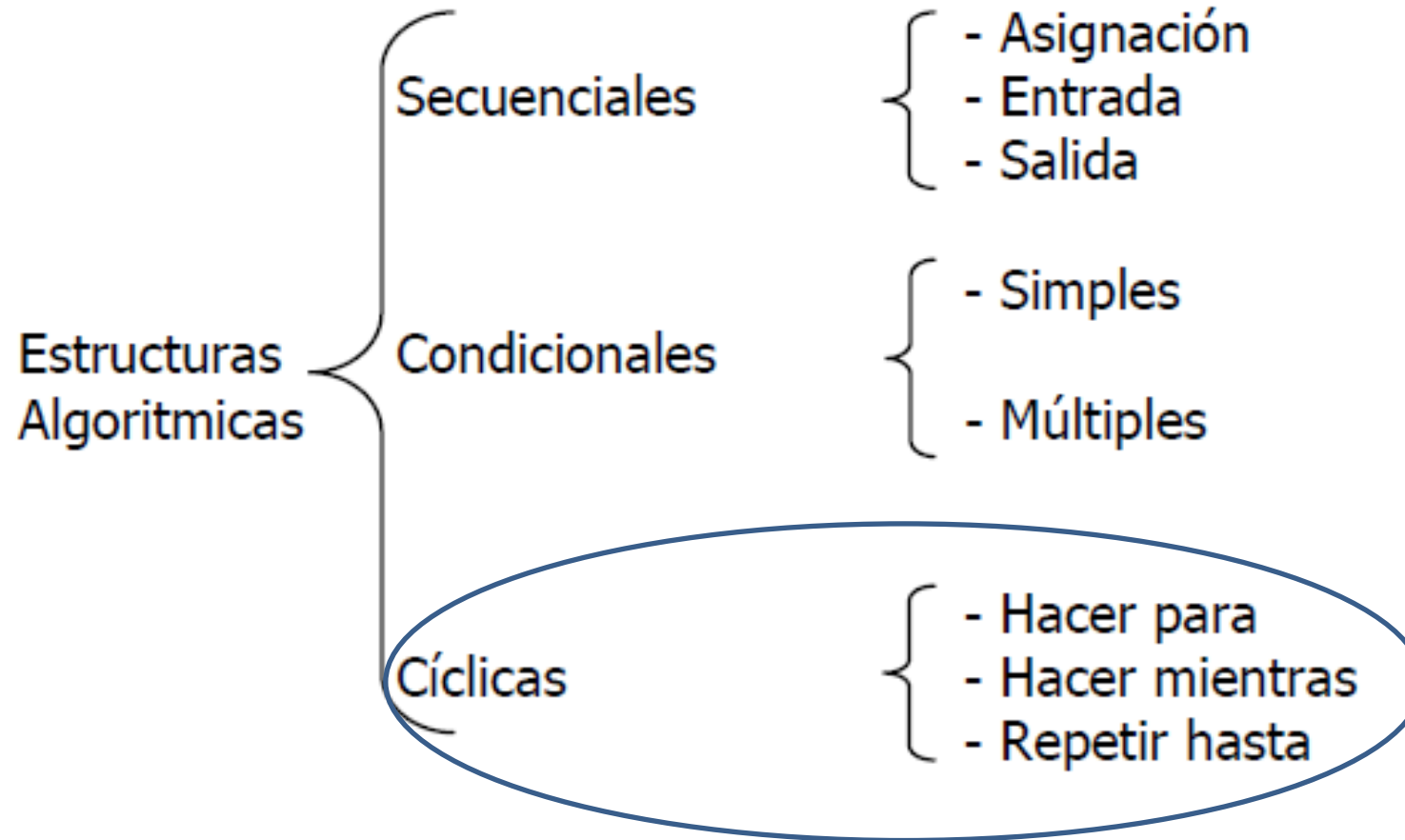




Estructuras Algorítmicas



Estructuras de Control Básicas: Repetitivas

Supongamos que tenemos que resolver el problema de sumar una determinada cantidad de números. ¿Cómo resolveríamos el problema? Consideraríamos una variable SUMA que se inicializaría en cero y en la misma se irían sumando cada uno de los números ingresados.

```
...  
Suma <- 0  
Leer numero  
Suma <- suma + numero  
Leer numero  
Suma <- suma + numero  
Leer numero  
Suma <- suma + numero  
...
```

NO SE RESUELVE ASÍ !!!

Y así, la cantidad de números que se deseen sumar.

Es decir que el programa repite las sentencias de *LEER Y SUMAR* para cada número leído, tales acciones repetidas constituyen un *BUCLE* o *CICLO*

Estructuras de Control Básicas: Repetitivas

La estructura repetitiva se utiliza cuando se quiere que un conjunto de instrucciones se ejecuten un cierto número finito de veces.

Existen dos tipos de estructuras repetitivas:

- Aquella en donde se tiene perfectamente establecido el número de veces que un grupo de acciones se van a ejecutar .
- Aquella en la que el número de repeticiones es desconocido y se hará hasta que se cumpla o no cierta condición.

Repetitivas con cantidad conocida de ciclos

- Para

Cuando conocemos de antemano el número de veces en que se desea ejecutar una acción o grupo de acciones.

Esta estructura ejecuta las acciones del cuerpo del bucle un número especificado de veces.

Controla en modo automático el valor de una variable que cuenta la cantidad de ciclos ejecutados.

Para <VC>=<VI> **hasta** <VF> **Con Paso** **i** **hacer**

Instrucción

Fin_Para

VC: variable que controla el ciclo

VI: valor inicial

VF: valor final

i : incremento (**si es 1 puede no estar**)



Estructuras de Control Básicas: Repetitivas

- Para - Ejemplo

Realizar un algoritmo para hallar la potencia enésima de un número real (con n entero positivo) sin usar el operador de la potenciación

Análisis:

Consideremos que el número base de la potencia lo almaceno en la variable num y el exponente en n

Nos quedaría calcular,

$$\text{num}^n = \underbrace{\text{num} * \text{num} * \text{num} * \dots * \text{num}}_{n \text{ veces}}$$

n veces

Pseudocódigo:

Inicio

 Escribir 'Ingrese el valor de la base'

 Leer num

 Escribir 'Ingrese el valor del exponente'

 leer n

 pot = 1

 para i=1 Hasta n Hacer

 pot = pot * num

 Fin_Para

 Escribir 'La potencia es: ', pot

Fin



Repetitivas con cantidad desconocida de ciclos

Son aquellas en la que el número de ciclos o iteraciones no se conoce por anticipado y el cuerpo del bucle se realiza dependiendo de una determinada condición. Por esta razón se los denomina BUCLES CONDICIONALES.

Estas estructuras son básicamente dos:

Mientras

Repetir – Hasta que

Estas dos se diferencian en que la verificación de la condición para repetir el ciclo se hace al inicio con mientras y al final con repetir.



Estructuras de Control Básicas: Repetitivas – Mientras

El cuerpo del bucle se realiza **mientras** se cumple una determinada condición.

Mientras <condición> **Hacer**
 Instrucciones
Fin_Mientras

La condición (expresión lógica) se evalúa antes de cada ejecución del bucle.

Si es verdadera, se ejecuta el bucle, y si es falsa, el control pasa a la sentencia siguiente del bucle.

Si la condición resulta falsa cuando se ejecuta el bucle por primera vez, el cuerpo del bucle no se ejecutará nunca.



Estructuras de Control Básicas: Repetitivas – Repetir-hasta que

Se ejecuta hasta que la condición se hace verdadera. Después de cada ejecución del cuerpo del bucle la condición (expresión lógica) se evalúa, si es falsa se repite la ejecución del bucle, y si es verdadera, el control pasa a la sentencia siguiente del bucle.

Repetir

instrucciones

Hasta Que <condición>



Estructuras de Control Básicas: Repetitivas

Para Tener en cuenta:

Mientras o Hasta que, la condición sea verdadera el bucle se ejecutará.

Esto significa que el bucle se ejecutará a menos que “algo” en el interior del bucle modifique la condición haciendo que su valor cambie.

Si la expresión nunca cambia el valor, entonces el bucle no termina nunca y se denomina *bucle infinito*.

Es decir que la condición lógica debe ser modificada por una sentencia en el cuerpo del bucle, en caso contrario el bucle es infinito

La condición lógica **debe tener un valor** la primera vez que se evalúa



Ejemplo: repetir hasta que

Suele usarse para controlar el ingreso de datos:

Ingresar un numero positivo

.....

Repetir

 Escribir 'Ingrese un numero positivo'

 leer numero

Hasta Que numero > 0

....



Ejemplo – División sin usar el operador

Escribir un algoritmo para obtener el resultado de la división entre dos números enteros positivos sin usar el operador división ni el operador MOD. El algoritmo debe informar, como resultado, el cociente y el resto de la división.

Análisis:

Si el dividendo lo almacenamos en N1 y el divisor en N2

Lo que tenemos que hacer es

N1 N2
 COCIENTE

RESTO Pero sin usar división ni MOD



Si a N1 le restamos N2 (siempre que podamos)

La cantidad de veces que pude restar es el COCIENTE

Y lo que me sobro es el RESTO



Ejemplo – División sin usar el operador

Veamos un ejemplo con números

$N1=13$ y $N2=3$

13

-3

10

-3

7

-3

4

-3

1

Pude restar 4 veces
Entonces $COCIENTE=4$

$RESTO=1$

Pero si $N1=5$ y $N2=7$

$COCIENTE=0$

$RESTO=5$

Por el segundo ejemplo vemos que puede ser que nunca se tenga que realizar el ciclo, entonces inicializamos las variables $COCIENTE$ en 0 y $RESTO$ con el mismo valor de $N1$

Para luego si hacer una repetición de restas mientras se pueda!!!



Ejemplo – División sin usar el operador

Pseudocódigo

inicio

Repetir

 Escribir 'Ingrese Dividendo'

 Leer N1

hasta que $n1 > 0$

Repetir

 Escribir 'Ingrese Divisor'

 Leer N2

hasta que $N2 > 0$

cociente = 0

resto = N1

Mientras Resto $\geq$ N2 Hacer

 Resto = resto - N2

 cociente = cociente + 1

Fin_Mientras

Escribir 'El cociente de la división es: ', cociente, ' y el resto es: ', resto

Fin