Estructuras básicas de control.

- Estructura secuencial -

Normalmente un programa, o una parte del mismo, consiste en una secuencia de instrucciones que se tienen que ejecutar una detrás de otra para realizar una operación. Esto se denomina una estructura secuencial y se compone de un grupo de acciones que se han de realizar todas y en el orden en que están escritas, sin posibilidad de omitir ninguna de ellas.

Por ejemplo, calcular la suma y el producto de dos números:
El problema es muy simple, lo primero que se tiene que hacer es leer dos números A y B. Después, sumarlos, luego multiplicarlos y por último, mostrar los resultados en la pantalla de la computadora. Estas acciones se deben ejecutar en este orden y secuencialmente.

Y el programa en pseudocódigo sería:

Código:

```
Inicio
      Leer A y B
      SUMA = A  B
      PRODUCTO = A * B
      Mostrar SUMA, PRODUCTO
Fin
```

Se ve que la estrucutra secuencial expresa perfectamente la solución al problema.

- Estructura condicional -

Cuando se esta construyendo un programa, es normal tener que seleccionar un conjunto de instrucciones entre varias posibles, dependiendo de que se cumpla o no una determinada condición.

Esto se denomina estructura condicional que presenta las variantes:

- Estructura condicional simple
En este tipo, si la condición se cumple, se ejecutan un conjunto de acciones, pero si no se cumple no se realiza ninguna accion.

El pseudocódigo de esta estructura es el siguiente:

Código:

```
Si condición
  Entonces
      Acción-1
      Acción-2
      ..........
      Acción-n
Fin_si
```

Por ejemplo, el problema a resolver consiste en leer por teclado un número, que se denominará NUM, y si es mayor que 20 visualizarlo en la pantalla.

Código:

```
Inicio
    Leer un número (NUM)
    Si NUM > 20
        Entonces
            Mostrar NUM
    Fin_si
Fin
```

En el ejemplo se señala que si, y sólo si, el valor NUM es mayor que 20 se debe mostrar este valor; pero si no se cumple la condición no se hace nada.

- Estructura condicional doble

En este tipo se ejecutará un bloque de instrucciones u otro, dependiendo de que la condición sea cierta o falsa.

El presudocódigo de esta variante es:

Código:

```
Si condición
  Entonces
      Acción-1
      Acción-2
      .....
      Acción-n
  Si no
      Acción-A
      Acción-B
      .....
      Acción-Z
Fin_si
```

Por ejemplo: seleccionar y visualizar el mayor de dos números leídos.

Código:

```
Inicio
  Leer A y B
  Si A > B
      Entonces
        Mostrar "El Mayor es:" A
      Si no
        Mostrar "El Mayor es:" B
  Fin_si
Fin
```

El problema es sencillo: primero se leen los dos números A y B. Si A es mayor que B entonces se visualiza en la pantalla el texto "El Mayor es:" y a continuación el valor de A. Y en caso contrario, es decir, si A no es mayor que B, se visualiza el texto "El Mayor es:" seguido del valor de B. El programa así diseñado es ilustrativo, pero no es totalmente correcto debido a que ambos números pueden ser iguales, y no contempla esta opción, que se comentará posteriormente.

- Estructura condicional múltiple

En este caso, dependiendo del valor que tome la variable númerica que controla la condición, se ejecutará una de las *n* acciones posibles definidas en la estructura.

El pseudocódigo de esta variante es el siguiente:

Código:

```
Según condición
  = 1 Acción-1
  = 2 Acción-2
  .....
  = n Acción-n
Fin_según
```

Por ejemplo: leer desde el teclado un valor numérico, denominado NUM-CARTA, asociado a una carta de una baraga (del 1 al 12), y mostrar en pantalla el nombre de la carta asociado a ese número.

Código:

```
Inicio
  Leer NUM-CARTA
```

```
  Según NUM-CARTA
      = 1 Mostrar "As"
      = 2 Mostrar "Dos"
      ..........
      = 12 Mostrar "Rey"
  Fin_según
Fin
```

En primer lugar, se lee un número de la carta y, a continuación, dependiendo del valor de la variable NUM-CARTA se mostrará el nombre de la carta. Las estructuras condicionales múltiples derivan de las estructuras condicionales dobles de tipo anidado, es decir, cuando la parte "si no" de una estructura condicional doble se transforma en otra estructura condicional.

Así, el ejemplo de seleccionar y visualizar el mayor de dos números dados debería quedar programado, teniendo en cuenta que pueden ser iguales, como sigue:

Código:

```
Inicio
  Leer A y B
  Si A > B
      Entonces
        Mostrar "El Mayor es:" A
  Si no
  Si A < B
      Entonces
        Mostrar "El Mayor es:" B
  Si no
        Mostrar "A y B son iguales"
  Fin_si
Fin
```

El programa lee dos números A y B, si A es mayor que B muestra A como el mayor, si no pregunta si A es menor que B y en caso afirmativo muestra B como el mayor. Pero si no se cumplen ninguna de las dos condiciones quiere decir que ambos números son iguales.

Transformado en una "estructura condicional múltiple" sería de la forma:

Código:

```
Inicio
  Leer A y B
  Según Comparación (A,B)
      = 1 Mostrar "El Mayor es:" A
      =-1 Mostrar "El Mayor es:" B
```

```
        = 0 Mostrar "A y B son iguales"
    Fin_según
  Fin
```

En este caso, se compara A con B mediante la función Comparación (A,B) que devuelve el valor 1,-1,0 según sea el resultado de comparar A con B. Si el resultado de la comparación es igual a 1 quiere decir que A > B, si es -1 indica que A < B y si es 0, señala que A y B son iguales.

- Estructura repetitiva -

En un programa es normal tener que ejecutar repetidamente un conjunto de instrucciones, dependiendo de que sea cierta o no una condición. La condición se conoce como "condición de salida" y la instrucción se denomina "repetitiva" o "de bucle".

Existen tres versiones:
- Estructura tipo Mientras.
- Estructura tipo Hasta.
- Estructura tipo Para.

- Estructura tipo Mientras

En este tipo, el bloque de instrucciones (acciones) se repetirá mientras que la condición sea cierta. La condición se evalua al comienzo de la estructura. Esto implica que el bloque de instrucciones puede no ejecutarse ninguna vez si la condición de salida es inicialmente falsa.

El pseudocódigo de esta estructura es de la forma:

Código:

```
Mientras Condición
  Acción-1
  Acción-2
  ..........
  Acción-n
Fin_mientras
```

Por ejemplo: una empresa tienen grabados los datos personales de sus empleados. se desea imprimir el nombre de aquellos empleados mayores de 55 años.

Código:

```
Inicio
  Leer Empleado
  Mientras Haya-Empleado
      Si Empleado-Edad > 55
        Entonces
            Mostrar Empleado
      Fin_si
```

```
      Leer Empleado
  Fin_mientras
Fin
```

En primer lugar, se leen los datos de un empleado, a continuación se evalúa la condición de salida (Haya-Empleado), preguntando si existen empleados. Si la condición es cierta, porque realmente se han leído los datos de un empleado, entonces se comprueba, con una estructura condicional simple, si la edad del empleado es mayor de 55 años, si lo es se muestra el nombre del empleado. Después se vuelve a leer otro empleado y se retorna a la evaluación de la condición salida del bucle, cuya finalización se realiza cuando no existan más empleado. Observese que si en la primer lectura no hay empleados, el bucle Mientras no se realizará nunca, ya que se comprueba la condición al principio de la estructura Mientras y, sólo si es cierta, se entra en ello.

- Estructura tipo Hasta

En este tipo, el bloque de acciones se repetirá hasta que la condición sea cierta. Dicha condición se evalúa al final de la estructura. Esto implica que el bloque de instrucciones se ejecutará al menos una vez, aunque la condición de salida ya sea cierta al entrar en dicha estructura.
La diferencia fundamental entre ambas estructuras repetitivas es que, en el primer tipo (tipo Mientras), las acciones del bucle no se realizan nunca si la condición de salida del mismo es inicialmente falsa.
Por el contrario, las acciones del bucle, en el segundo tipo (tipo Hasta), se realizarán al menos una vez, ya que la condición se evalúa después de haber sido realizadas dichas acciones.

El pseudocódigo para esta estructura es:

Código:

```
Repetir
  Acción-1
  Acción-2
  ......
  Acción-n
Hasta Condición de Fin_repetir
```

Por ejemplo: "Visualizar la tabla de multiplicar del número 4"

Código:

```
Inicio
  CONT = 0
  Repetir
      CONT = CONT + 1
      PROD = CONT * 4
      Mostrar PROD
```

```
  Hasta CONT = 10
Fin
```

Para resolver este problema se necesitan dos variables: CONT, que es un contador que almacena cuatos números se han visualizado (es la varibale que contiene la condición de salida del bucle); y PROD, que almacena el valor del número a mostrar y que corresponde a los números de la tabla.

- Estructura tipo Para

Si el número de repeticiones del bucle (iteraciones) es fijo o se conoce de antemano, se puede utilizar una estructura tipo "Para", en lugar de una estructura tipo "Mientras". La estructura "Para" indica que las acciones del bucle se realizan un número específico de veces y que la estructura controla automáticamente el número de repeticiones.
Para dicho control hay que definir dentro de la estructura el nombre de una variable, su valor inicial, su valor final y un incremento fijo.
Ello quiere decir que inicialmente el bloque de acciones se ejecuta con el valor inicial de la variable, incrementándose este valor en cada iteración con el valor del incremento y finalizan las iteraciones cuando el valor de la variable sobrepasa su valor final. La evaluación de la condición de salida se realiza al comienzo de cada iteración.

El pesudocódigo de este tipo de estructura es el siguiente:

Código:

```
Para VAR desde V1 hasta V2 incremento V3
  Acción-1
  Acción-2
  .....
  Acción-n
Fin_para
```

Por ejemplo: "Construir un programa en pseudocódigo estructurado que calcule la suma de los números comprendidos entre 1 y 100, ambos inclusive"

Código:

```
Inicio
  SUMA = 0
  Para NUM desde 1 hasta 100
      SUMA = SUMA + NUM
  Fin_para
  Mostrar SUMA
Fin
```

En el ejemplo se utiliza la variable SUMA para almacenar la suma de los 100 primeros números y la variable NUM para controlar el bucle. Éste empieza en 1 y llega hasta 100 con incremento 1, y cuando NUM sobrepasa el valor 100, indica que ya se han sumado todos los números, con lo que el bucle tipo "Para" termina y en SUMA se tendrá el valor buscado.
La diferencia fundamental entre los tipos "Mientras" y "Para" radica en que en la estructura "Mientras" hay que realizar, mediante instrucciones, la inicialización de la variable que controla el bucle y su incremento; mientras que en la estructura "Para" esto se hace automáticamente.

Así el ejemplo anterior desarrollado con una estructura del tipo Mientras daría lugar al siguiente código:

Código:

```
Inicio
  SUMA = 0
  NUM = 1
  Mientras NUM <= 100
      SUMA = SUMA + NUM
      NUM = NUM + 1
  Fin_mientras
  Mostrar SUMA
Fin
```

Observesé que en el código anterior hay que incorporar la instrucción de inicialización de NUM (NUM=1), y de incremento (NUM=NUM 1), cosa que no hizo falta en la estructura "Para". Pero recuérdese que la estructura "Para" sólo es aplicable cuando se conoce a priori el número de repeticiones del bucle, es decir el valor V3 de la variable VAR