MATERIA: **FUNDAMENTO DE COMPUTADORES DIGITALES**

Estructuras de Repetición

UNLZ
Universidad Nacional
de Lomas de Zamora

Facultad de
INGENIERÍA
UNLZ

Estructuras de Repetición

Las **estructuras de repetición** son las llamadas **estructuras** cíclicas, iterativas o de bucles. Permiten ejecutar un conjunto de instrucciones de manera repetida (o cíclica) mientras que la expresión lógica a evaluar se cumpla (sea verdadera). En lenguaje C existen tres **estructuras de repetición**: while, do-while y for.

while

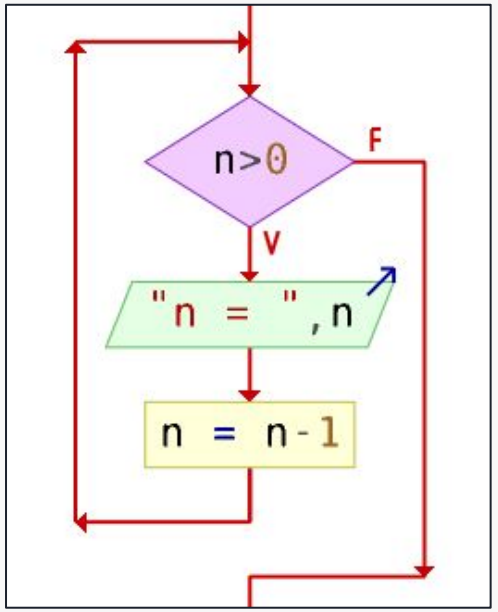
Esta estructura repite una instrucción o grupo de instrucciones mientras una expresión lógica sea cierta.

Cuando no se conoce el número de repeticiones por anticipado, la sentencia while lo resuelve con el empleo de una determinada condición.

Lo primero que hace esta sentencia es evaluar si se ejecuta el bucle, es decir que es posible que nunca se realice acción alguna.

Mientras la condición se cumpla, el bucle sigue iterando, por eso es importante no caer en ciclos de repetición infinitos.

while



Itera cero o más veces, eventualmente infinitas

Sintaxis en C:

```
while (condicion) {
    Sentencias
}
```

Ejemplo sintaxis en C

```
while (n > 0) {
        printf("n = %d\n", n);
        n--;
}
```

Ejemplo: calcular la suma de números ingresados por teclado hasta que se ingrese un cero.

```c

#include <stdio.h>
#include <stdlib.h>

int main()
{
    int S = 0;
    int N;
    printf("Ingrese un numero (0 para salir): ");
    scanf("%d",&N);
    while ( N != 0)
    {
        S = S + N;
        printf("Ingrese un numero (0 para salir): ");
        scanf("%d",&N);
    }
    printf("La sumatoria es  %d",S);
    return 0;
}
```

Ejemplo:
hallar el promedio de números ingresados por teclado hasta que aparezca uno que no sea par y mayor a 20.

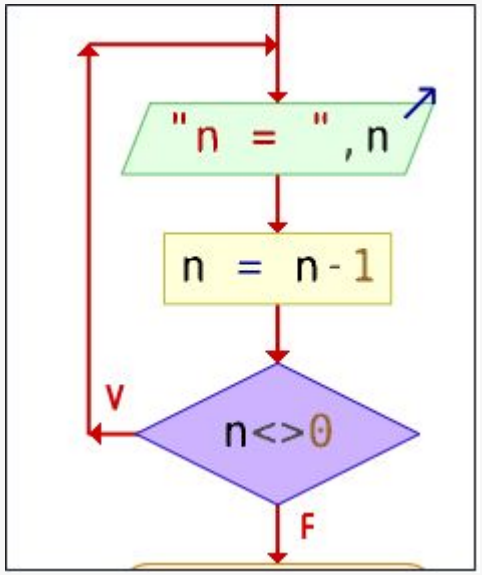
```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int S = 0;
    int C = 0;
    int N;
    float P;
    printf("Ingrese un numero: (salir con num par mayor a 20)");
    scanf("%d",&N);
    while ( !((N%2==1) && (N>20) ) )
    {
        S = S + N;
        C++;
        printf("Ingrese un numero: ");
        scanf("%d",&N);
    }
    if(C == 0)
        printf("No hubo números de ingreso");
    else
    {
        P = S/C;
        printf ("El promedio es %f ",P);
    }

    return 0;
}
```

do while

Esta estructura repite una instrucción o grupo de instrucciones mientras una expresión lógica sea cierta.

Un aspecto muy importante de la presente estructura de control es que la expresión lógica no se evalúa hasta el final de la estructura con lo cual el bucle se ejecuta al menos una vez, contraria a la estructura «while» que podía no ejecutarse nunca. Es decir que después de cada iteración el bucle evalúa la condición, si es verdadera sigue repitiendo y si la condición es falsa termina.

do while



Itera una o más veces, eventualmente infinitas

Sintaxis en C:

```
do {
        sentencias
} while (condicion);
```

Ejemplo sintaxis en C

```
do {
        printf("n = %d\n", n);
        n--;
} while (n);

/* se detiene cuando n valga cero */
/* equivalente a poner while (n<>0)*/
```

Ejemplo: hacer un algoritmo que muestre del 0 al 20 (solo los números pares).

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int N = 0;
    do
    {
        printf("%d \n",N);
        N = N + 2;
    }
    while ( N < 21 );
}
```

Ejemplo:
en un bosque se necesita saber el promedio de diámetro de cada tronco de ciprés y el promedio de su altura. El proceso termina cuando el usuario responde con una 'N', mientras tanto, debe responder con 'S'.

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int A,D;
    char opcion;
    int C = 0;
    float SA = 0;
    float SD = 0;
    do
    {
        printf("Ingrese su altura: ");
        scanf("%d",&A);
        printf("Ingrese su diametro: ");
        scanf("%d",&D);
        SA = SA + A;
        SD = SD + D;
        C++;
        printf("¿Desea seguir ingresando datos? S/N \n");
        scanf(" %c",&opcion);
    }
    while(opcion=='S');
    float PA = SA/C;
    float PD = SD/C;
    printf ("El promedio de altura de los cipreses es %f \n",PA);
    printf ("El promedio de diámetro de los cipreses es %f \n",PD);
    return 0;
}
```

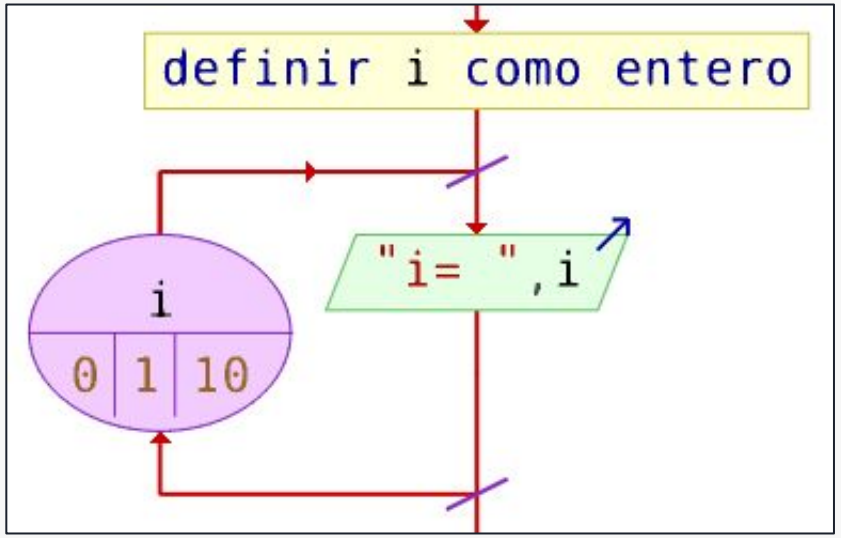
Bucle For

Forma general

```
for (sentencias-antes; condición; sentencias-finales) {
        Sentencias;
}
```

Es equivalente a

```
sentencias-antes
while (condición) {
        Sentencias
        sentencias-finales
}
```




```
*** Ejecución Iniciada. ***
i= 0
i= 1
i= 2
i= 3
i= 4
i= 5
i= 6
i= 7
i= 8
i= 9
i= 10
*** Ejecución Finalizada. ***
```

Ejemplo C

```c
for(i = 0; i < 10; i++)
    printf("i = %d \n", i);
```

```c
//------>> equivalente
i = 0;
while (i < 10)
{
    printf("i = %d \n", i);
    i++;
}
```

```c
for(n = 100,j = 2; j < n; j += 2, n -= j)
{
    printf("j = %d      n = %d      \n", j, n);
}
```

```c
//------>> equivalente
n = 100;
j = 2;
while (j < n)
{
    printf("j = %d      n = %d      \n", j, n);
    n -= j;
    j += 2;
}
```

Ejemplo: ingresar diez números y hallar su promedio.

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    float S = 0; //usamos float para evitar el redondeo en el promedio
    int i;
    float P;
    int A;
    for (i=0; i<10; i++)
    {
        printf("Ingrese un valor entero: ");
        scanf("%d",&A);
        S = S + A;
    }
    P = S/10;
    printf ("El promedio es %f ",P);
    return 0;
}
```

Es importante saber que los resultados finales deben ubicarse fuera del ciclo de repetición, por el contrario, estaríamos mostrando en el ejemplo diez veces “El promedio es “…

Ejemplo:
hallar el promedio de altura de los alumnos del curso A y el promedio de los alumnos del curso B y mostrar sus resultados. El curso A tiene 26 alumnos mientras que el B tiene 30

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    float SA = 0;
    float SB = 0;
    int altura;
    float PA,PB;
    printf("Curso A: ");
    for (int i=0; i<26; i++)//a partir del estandar C99 se puede declarar I acá
    {
        printf("Ingrese la altura del alumno: ");
        scanf ("%d",&altura);
        SA = SA + altura;
    }
    printf("Curso B: ");
    for (int i=0; i<30; i++)
    {
        printf("Ingrese la altura del alumno: ");
        scanf ("%d",&altura);
        SB = SB + altura;
    }
    PA = SA/26;
    PB = SB/30;
    printf("El promedio de altura del curso A es %f",PA);
    printf("El promedio de altura del curso B es %f",PB);
}
```

Ejemplo: mostrar de modo decreciente los números del N al 1. El usuario debe ingresar dicho número.

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int N;
    printf("Ingrese la cantidad de repeticiones: ");
    scanf("%d",&N);
    for (int i=N; i>0; i--)
    {
        printf("%d ",i);
    }
    return 0;
}
```

Bucle for anidado

Los bucles pueden estar anidados, es decir uno dentro de otro. Se debe tener cuidado con que el bucle interior esté completamente dentro del bucle exterior. Si se cruzan, se produce un error.

Ejemplo: realizar un algoritmo que produzca por pantalla lo siguiente:

1	→	100	110	120	130	140	150
2	→	100	110	120	130	140	150
3	→	100	110	120	130	140	150
4	→	100	110	120	130	140	150
5	→	100	110	120	130	140	150
6	→	100	110	120	130	140	150
7	→	100	110	120	130	140	150
8	→	100	110	120	130	140	150
9	→	100	110	120	130	140	150
10	→	100	110	120	130	140	150

Solución

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    for (int i=1; i <= 10; i++)
    {
        printf ("%d ----> ",i);
        for (int j=100; j<=150; j=j+10)
        {
            printf("%d  ",j);
        }
        printf("\n"); //imprime un salto de linea
    }
    return 0;
}
```

break, continue y goto

En el lenguaje C, el break, continue y goto son instrucciones de control de flujo que permiten modificar el comportamiento del programa en ciertas situaciones.

- **break** sirve para salir del bloque **switch**, **for** ,**while** o **do while** que lo encierra y seguir con la siguiente instrucción al bloque en que se encuentra.

- **continue** se usa en ciclos **for** ,**while** y **do while**. Su efecto es saltear el resto del ciclo y volver a chequear la condición del ciclo para o bien comenzar una nueva iteración o bien darlo por terminado.

- **goto** esta instrucción permite saltar a una etiqueta específica dentro del código. El uso de goto se considera una mala práctica de programación, ya que puede hacer que el código sea difícil de entender y de mantener.

Ejemplos break y continue

```c
#include <stdio.h>

int main() {
   int i;

   // Ejemplo de break
   printf("Ejemplo de break\n");
   for (i = 1; i <= 10; i++) {
      if (i == 5) {
         break;
      }
      printf("%d ", i);
   }
   printf("\n\n");

   // Ejemplo de continue
   printf("Ejemplo de continue\n");
   for (i = 1; i <= 10; i++) {
      if (i == 5) {
         continue;
      }
      printf("%d ", i);
   }
   printf("\n");

   return 0;
}
```

Ejemplo de **break**
1 2 3 4

Ejemplo de **continue**
1 2 3 4 6 7 8 9 10

Ejemplo goto

```c
#include <stdio.h>

int main() {
   int i = 1;

   // Ejemplo de goto
   printf("Ejemplo de goto\n");
   inicio:
   printf("%d\n", i);
   i++;
   if (i <= 5) {
      goto inicio;
   }

   return 0;
}
```

```
Ejemplo de goto
1
2
3
4
5
```