

# TEMA 3:

Fundamentos para la construcción de código a partir del algoritmo

# TEMA 3:

## 3.3 Estructuras de control condicional e iterativo

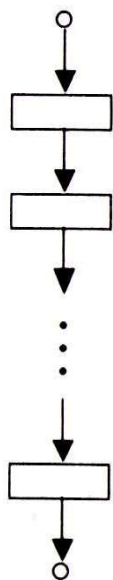
# Objetivo

- El alumno construirá programas utilizando el lenguaje de programación C a través de un análisis y modelado algorítmico previo.

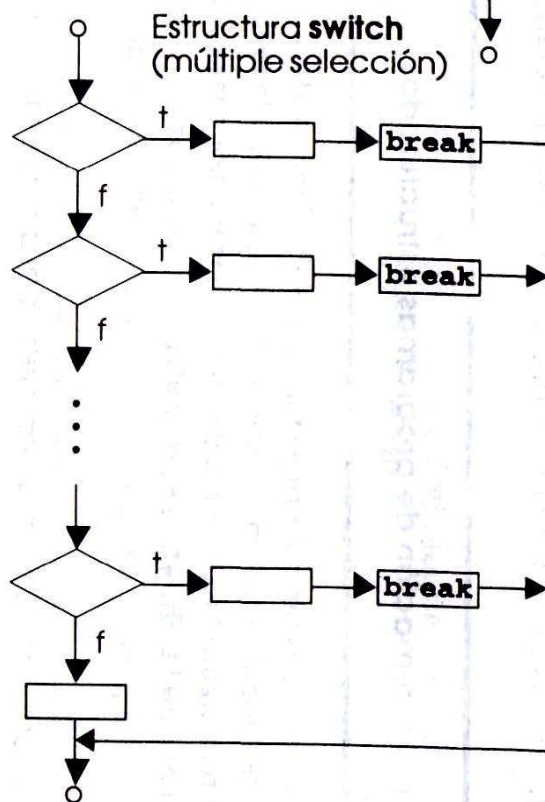
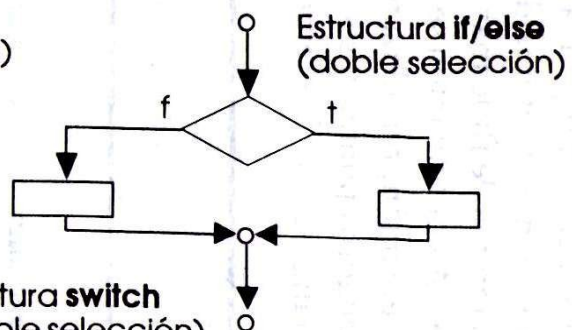
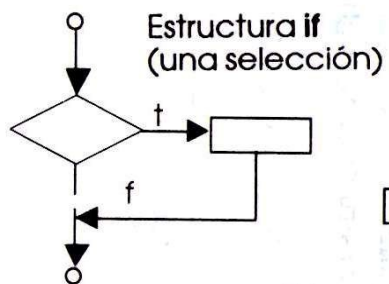
# Ciclos

- Nos permiten repetir una tarea determinado número de veces o mientras una condición que es validada se cumpla.
- Optimizan código al evitar escribirlo repetidamente.
- Ayudan a recorrer arreglos y matrices.
- Requieren una variable auxiliar contador.
- Dentro del paréntesis se colocan dos variables o una variable y un número separados por un operador de comparación.

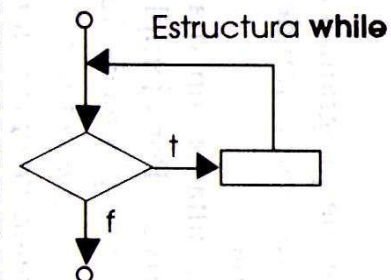
## Secuencia



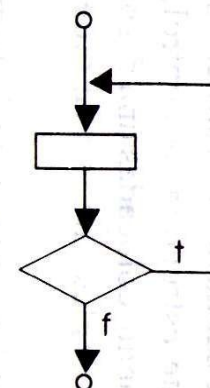
## Selección



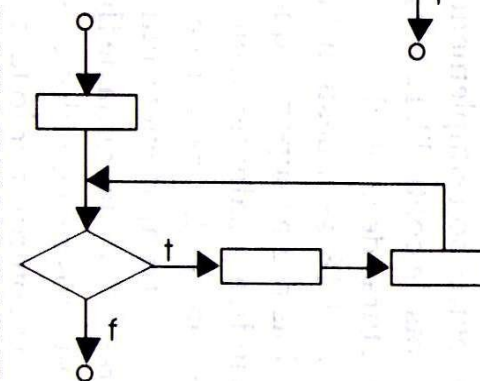
## Repetición



## Estructura do/while



## Estructura for



# Elementos de un ciclo

- Cuando deben hacer la plana "No debo bajar la tarea de internet" 100 veces realizan un ciclo.

**Empiezan a contar en 1**

→ 1

No debo bajar la tarea de internet

**Llevan la cuenta de 1 en 1**

→ 2

No debo bajar la tarea de internet

→ 3

No debo bajar la tarea de internet

**Terminan la plana al llegar a 100**



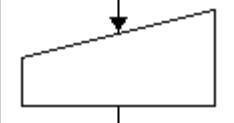
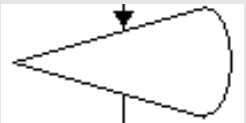


→ 100

No debo bajar la tarea de internet

# Elementos de un ciclo



# Pseudocódigo – DFD - C

ACCIÓN	DFD	LENGUAJE C
INICIO / ALGORITMO		<pre>int main(void) int main(int argc, char* argv[])</pre>
ASIGNAR / DECLARAR / DEFINIR		<pre>int x=0; float y=5.1;</pre>
LEER Y GUARDAR EN / LEER Y ALMACENAR EN		<pre>scanf("%d", &amp;x); scanf("%f", &amp;y); scanf("%d,%f", &amp;x, &amp;y);</pre>
IMPRIMIR / ESCRIBIR / MOSTRAR MENSAJE		<pre>printf("Hola Mundo"); printf("La suma es: %d", x);</pre>
CALCULAR / REALIZAR OPERACIÓN		<pre>z= x + y; suma= suma + conde;</pre>
FIN / FIN ALGORITMO		<pre>return 0;</pre>



# Ciclos Para y Mientras que

ACCIÓN	LENGUAJE C
<pre>DEFINIR Entero: conde=0  PARA conde=1 HASTA conde&lt;=10 INCREMENTO=1     acciones FIN PARA</pre>	<pre>int conde;  for (conde=1; conde&lt;=10; conde++) {     sentencias; }</pre>
<pre>DEFINIR Entero: conde=1  MIENTRAS QUE conde&lt;=10     acciones FIN MIENTRAS QUE</pre>	<pre>int conde=1;  while (conde&lt;=10) {     sentencias;     //conde++; }</pre>

# Ciclo Hacer-Mientras que

ACCIÓN	LENGUAJE C	DESCRIP.
<pre>DEFINIR Entero: conde=1  HACER     acciones FIN HACER  MIENTRAS QUE conde&lt;=10</pre>	<pre>int conde=1;  do {     sentencias;     //conde++; } while (conde&lt;=10) ;</pre>	<p>Siempre realiza por lo menos una vez el bloque de instrucciones agrupados en <b>do</b> se cumpla o no la condición que se evalúa en <b>while</b>. <b>Termina en “;”</b> a diferencia de while</p>

# Variable auxiliar Contador

- Variable cuyo valor se incrementa o decrementa en forma constante cada vez que ocurre una acción o repetición.
- Debe ser declarada de tipo de dato numérico, entera y de preferencia con un nombre relacionado, aunque generalmente se emplean las letras i, j, k, l, m, n.
- Controla la cantidad de veces que se repite un proceso o cálculo dentro de una estructura de repetición.
- Debe ser declarada como cualquier otra variable.
- Debe ser inicializada ya sea dentro de la misma instrucción de repetición si es un PARA o fuera del ciclo en los casos del MIENTRAS Y HACER-MIENTRAS.
- En su forma larga debe aparecer a ambos lados de la igualdad.

# Elementos de un ciclo

**Inicio  
del  
ciclo**

Es  
conde  $\leq 10$  ?

Yo soy el  
contador  
conde,  
conde  
vale  
ahora 1

**1er. Incremento de  
conde**

conde  
vale  
ahora 2

**2do. Incremento de  
conde**

conde  
vale  
ahora 3

Parte  
de:  
conde=1

Incremento en 1:  
conde = conde + 1;  
conde++;

**3er. Incremento de  
conde**

conde  
vale  
ahora 4

**4to. Incremento de  
conde**

conde  
vale  
ahora 5

Si conde  
vale 10  
hace última  
vez el  
contenido,  
si conde  
vale 11 el  
ciclo se  
rompe

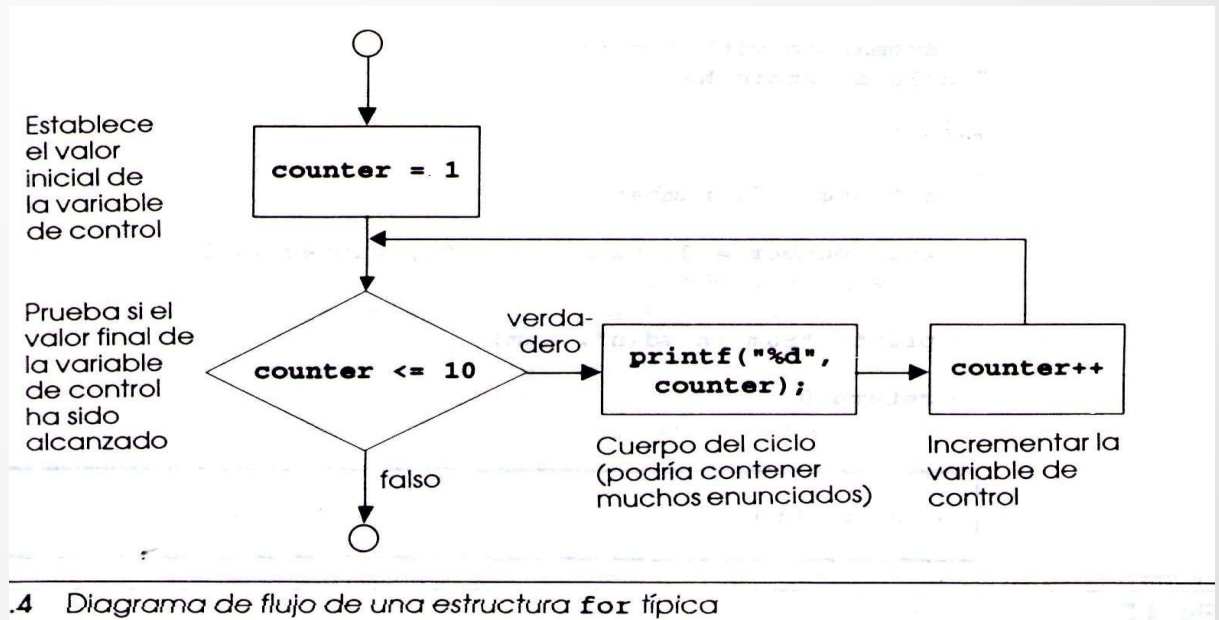
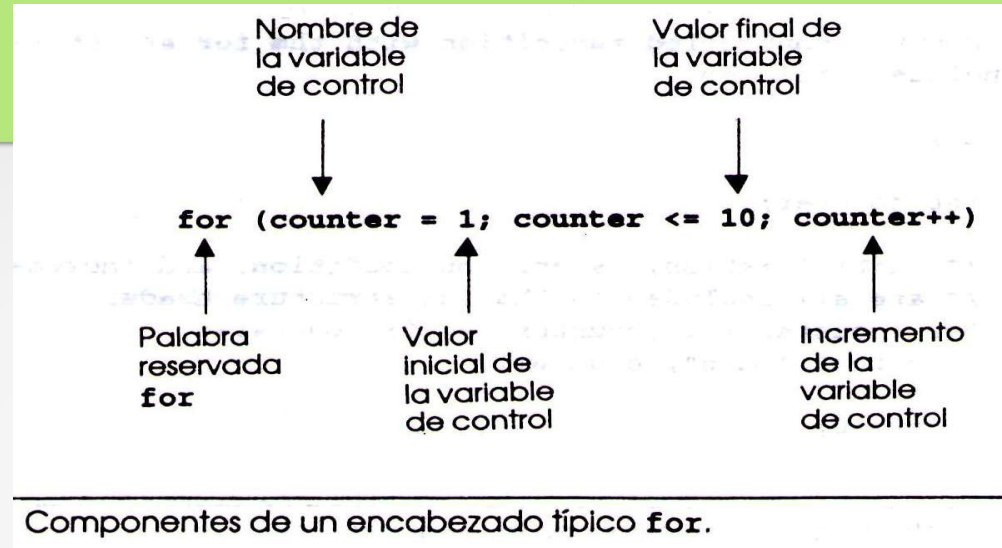


Hecho por Huicho :)

# Operadores y funciones en C

OPERACIÓN	FORMA 1	FORMA 2
Detener el ciclo en 10	<code>conde&lt;11</code>	<code>conde&lt;=10</code>
Incremento en 1	<code>conde++</code>	<code>conde= conde+1</code>
Incremento en 2	<code>conde+=2</code>	<code>conde= conde+2</code>
Decremento en 1	<code>conde--</code>	<code>conde= conde-1</code>
Decremento en 2	<code>conde-=2</code>	<code>conde= conde-2</code>
Acumulador/contador entre 2	<code>conde/=2</code>	<code>conde= conde/2</code>
Acumulador/contador por 2	<code>conde*=2</code>	<code>conde= conde*2</code>

# Ciclo for



# Ciclo Para de 1 al 10

INICIO

1. DEFINIR Entero: conde=0

2. IMPRIMIR 'Este algoritmo muestra del 1 al 10 usando un ciclo Para'

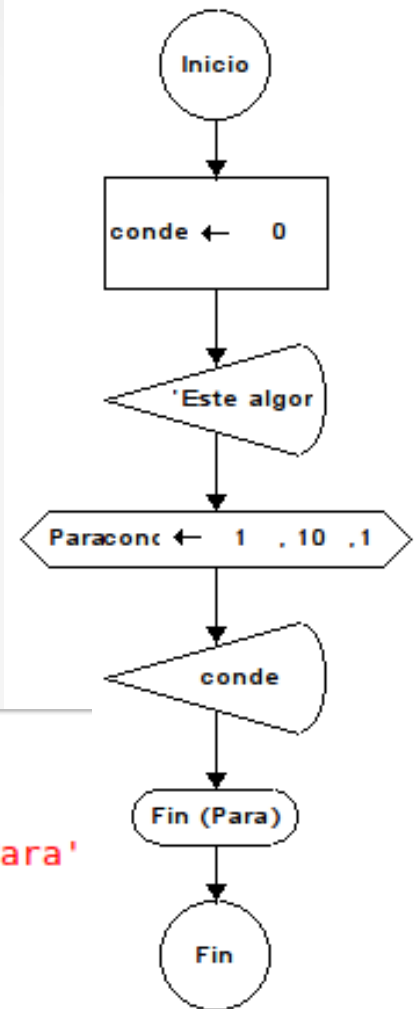
3. PARA conde=1 HASTA conde<=10 INCREMENTO=1

3.1 IMPRIMIR conde

4. FIN PARA

FIN

```
1  Algoritmo para1A10
2    Definir conde Como Entero
3    conde<-0
4    Escribir 'Este algoritmo muestra del 1 al 10 usando un ciclo Para'
5    Para conde<-1 Hasta 10 Con Paso 1 Hacer
6        Escribir conde
7    Fin Para
8  FinAlgoritmo
```

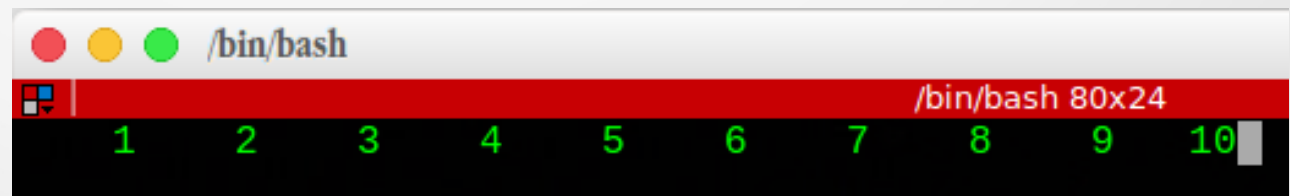




# Código en C y ejecución

- El contador se declara sin inicializar porque se hará en el **for**
- Es el más simple porque los 3 elementos están en la misma línea
- **%5hd** es el alias para imprimir un entero corto con un espacio de 5 dígitos

```
1  /*programa que muestra el valor del contador en ciclo for
2   * hecho por huicho*/
3
4   #include <stdio.h> //para printf y getchar
5
6   int main(int argc, char* argv[])
7   {
8       short conde; //contador entero pequeño
9
10      for(conde=1; conde<=10; conde++)
11      { //abre bloque a repetir
12          printf("%5hd", conde); //imprime lo que tiene contador
13      } //cierra bloque a repetir
14
15      getchar(); //mantiene en espera la ejecución
16      return 0;
17  }
```



```
/bin/bash
1 2 3 4 5 6 7 8 9 10
```



# Ciclo Mientras que de 1 al 10

INICIO

1. DEFINIR Entero: conde=1

2. IMPRIMIR 'Este algoritmo muestra del 1 al 10 usando un ciclo Mientras que'

3. MIENTRAS QUE conde<=10

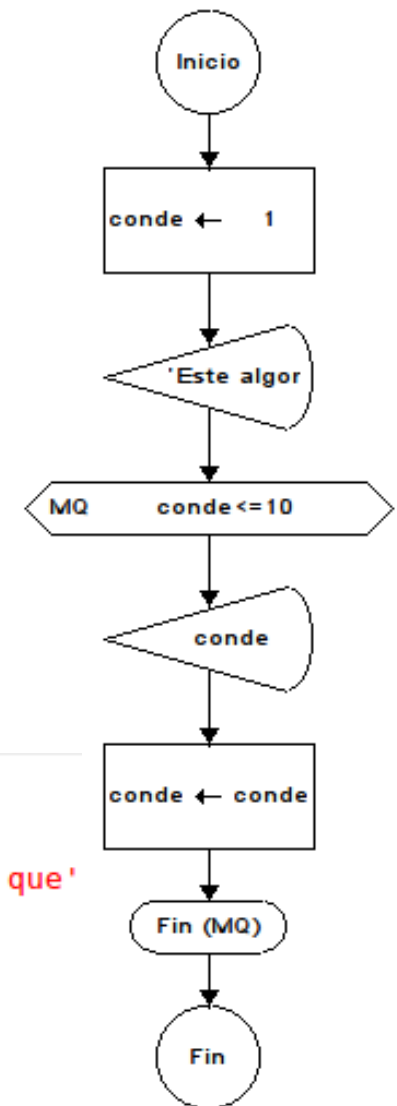
3.1 IMPRIMIR conde

3.2 CALCULAR conde= conde+1

4. FIN MIENTRAS QUE

FIN

```
1  Algoritmo mientras1A10
2      Definir conde Como Entero
3      conde<-1
4      Escribir 'Este algoritmo muestra del 1 al 10 usando un ciclo Mientras que'
5      Mientras conde<=10 Hacer
6          Escribir conde
7          conde<-conde+1
8      Fin Mientras
9  FinAlgoritmo
```



# Código en C y ejecución

- El contador conde se declara e inicializa porque **while** no lo hace
- El incremento se hace antes de cerrar el bloque del ciclo con forma larga o corta
- **%5hd** es el alias para imprimir un entero corto con un espacio de 5 dígitos

```
1  /*programa que muestra el valor del contador en ciclo while
2  * hecho por huicho*/
3
4  #include <stdio.h> //para printf y getchar
5
6  int main(int argc, char* argv[])
7  {
8      short conde=1; //contador entero pequeño inicializado
9
10     while(conde<=10) //valida condición y repite
11     {
12         printf("%5hd", conde); //imprime lo que tiene contador
13         conde= conde + 1; //incrementa contador en 1
14         //conde++; //incrementa contador en 1 con forma corta
15     }
16
17     getchar(); //mantiene en espera la ejecución
18     return 0;
19 }
```



```
/bin/bash
1 2 3 4 5 6 7 8 9 10
```

# Ciclo Hacer-Mientras que de 1 al 10

INICIO

1. DEFINIR Entero: conde=1

2. IMPRIMIR 'Este algoritmo muestra del 1 al 10 usando un ciclo Hacer-Mientras que'

3. HACER

3.1 IMPRIMIR conde

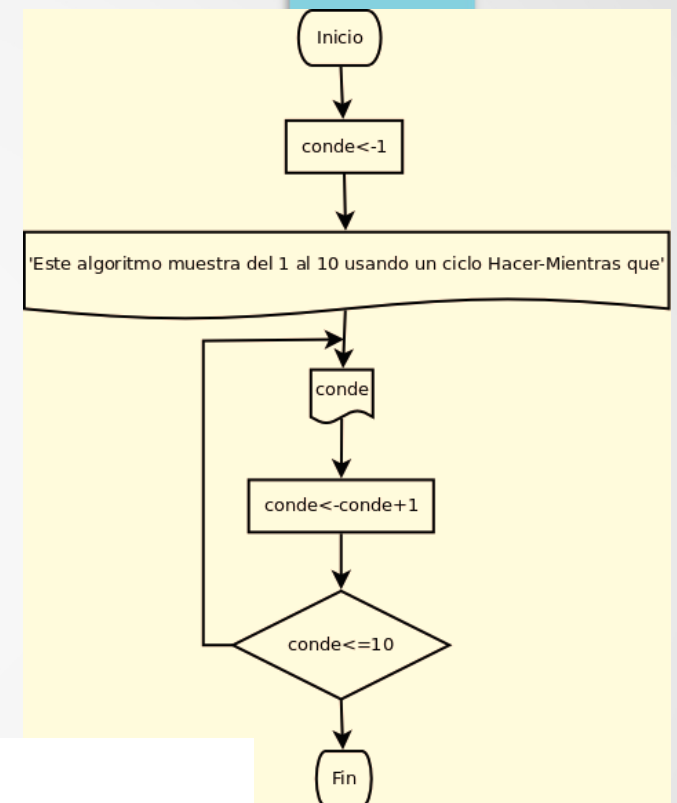
3.2 CALCULAR conde= conde+1

4. FIN HACER

5. MIENTRAS QUE conde<=10

FIN

```
1  Algoritmo repetirMientras1A10
2  Definir conde Como Entero
3  conde<-1
4  Escribir 'Este algoritmo muestra del 1 al 10 usando un ciclo Hacer-Mientras que'
5  Repetir
6  ..... Escribir conde
7  ..... conde<-conde+1
8  Mientras Que conde<=10 //se repite mientras la condición sea verdadera
9  FinAlgoritmo
```



# Código en C y ejecución

- El contador conde se declara e inicializa porque **do-while** no lo hace
- El incremento se hace antes de cerrar el bloque del ciclo con forma larga o corta
- **do-while** si lleva “;” después del paréntesis de la condición

```
1  /*programa que muestra el valor del contador en ciclo do-while
2  * hecho por huicho*/
3
4  #include <stdio.h> //para printf y getchar
5
6  int main(int argc, char* argv[])
7  {
8      short conde=1; //contador entero pequeño inicializado
9
10     do //bloque a repetir
11     {
12         printf("%5hd", conde); //imprime lo que tiene contador
13         conde= conde + 1; //incrementa contador en 1
14         //conde++; //incrementa contador en 1 con forma corta
15     }
16     while(conde<=10); //valida condición y repite
17
18     getchar(); //mantiene en espera la ejecución
19     return 0;
20 }
```



```
/bin/bash
/bin/bash 80x24
1 2 3 4 5 6 7 8 9 10
```

# Diferencia entre while y do-while

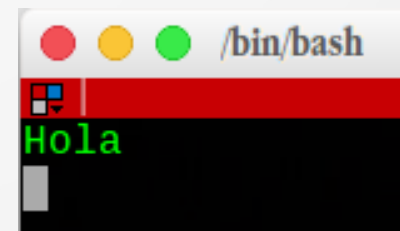
- Si inicializamos el contador en un número que provoque el incumplimiento de la condición a evaluar veremos que **por lo menos una vez el do-while ejecutará su bloque de código** a diferencia del ciclo while.
- Por ejemplo:
  - **Condición a evaluar** → `conde<=10`
  - **Valor inicial que no cumple** → `int conde=11;`

# Diferencia entre while y do-while

```
1  /*programa que muestra ciclo while
2  * hecho por huicho*/
3
4  #include <stdio.h> //para printf y getchar
5
6  int main(int argc, char* argv[])
7  {
8      short conde=11;
9
10     while(conde<=10)
11     {
12         printf("Hola\n");
13         conde= conde + 1;
14         //conde++; //forma corta
15     }
16
17     getchar();
18     return 0;
19 }
```



```
1  /*programa que muestra ciclo do-while
2  * hecho por huicho*/
3
4  #include <stdio.h> //para printf y getchar
5
6  int main(int argc, char* argv[])
7  {
8      short conde=11;
9      do
10     {
11         printf("Hola\n");
12         conde= conde + 1;
13         //conde++; //forma corta
14     }
15     while(conde<=10);
16
17     getchar();
18     return 0;
19 }
```



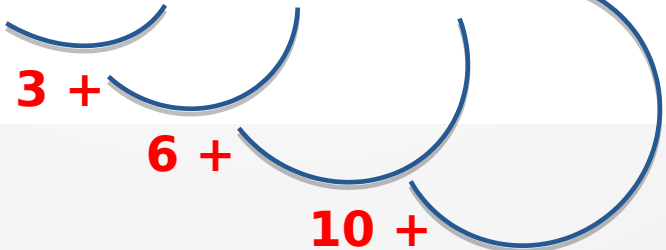
# Variable Acumulador

- Realiza la misma función que un contador con la diferencia de que el incremento o decremento no es constante.
- Es una variable que guarda el resultado de la suma de si misma con otra variable y eso lo suma de nuevo en la siguiente iteración por lo que siempre se almacena el resultado anterior.
- Debe ser declarada como cualquier otra variable y puede ser de cualquier tipo de dato numérico.
- Debe ser inicializada en 0 antes del ciclo cuando se opere en una suma o resta, o inicializada en 1 cuando se trate de una multiplicación.

# Acumulador en Suma

- Por ejemplo una suma definida como:

$$\sum_{i=m}^n x_i = x_m + x_{m+1} + x_{m+2} + \dots + x_n$$

$$\sum_{i=1}^5 i = 1 + 2 + 3 + 4 + 5 = 15$$




# Acumulador en Suma

- Para el ejemplo anterior:

```
int suma=0; //acumulador
int conde;  //contador
```

**conde=1**

```
for (conde=1; conde<=5; conde++)
{
    suma = suma + conde;
}  1  0  1
```

**conde=2**

```
for (conde=1; conde<=5; conde++)
{
    suma = suma + conde;
}  3  1  2
```

**conde=3**

```
for (conde=1; conde<=5; conde++)
{
    suma = suma + conde;
}  6  3  3
```

**conde=4**

```
for (conde=1; conde<=5; conde++)
{
    suma = suma + conde;
}  10  6  4
```

**conde=5**

```
for (conde=1; conde<=5; conde++)
{
    suma = suma + conde;
}  15  10  5
```

# Menú con selección múltiple

```
1  /*programa menú con switch - case
2  * hecho por huicho*/
3
4  #include <stdio.h> //para printf, scanf y getchar
5  #include <stdlib.h> //para exit
6
7  int main(int argc, char* argv[])
8  {
9      short opcion=0;
10
11     printf("Este programa implementa menú con selección múltiple \n\n", 163, 162, 163);
12     printf(" 1.Suma \n 2.Resta \n 3.Salir \n Ingresa tu opción: ",162);
13     scanf("%hd", &opcion);
14
15     switch(opcion) //evalúa variable opcion
16     { //abre bloque switch
17         case 1: //valida si opcion=1
18             printf("\n\n\t Opción suma", 162);
19             break; //evita que se ejecute el caso siguiente
20
21         case 2: //valida si opcion=2
22             printf("\n\n\t Opción resta", 162);
23             break; //evita que se ejecute el caso siguiente
24
25         case 3: //valida si opcion=3
26             printf("\n\n\t Opción salir", 162);
27             getchar(); //atrapa el enter del scanf
28             getchar(); //mantiene en espera la ejecución
29             exit(0); //termina la ejecución justo aquí
30             break; //evita que se ejecute el caso siguiente
31
32         default: //bloque a realizar si no fue ningún caso
33             printf("\n\n\t Opción no válida", 162, 160);
34             break; //evita que se ejecute el caso siguiente
35     } //cierra bloque switch
36
37     getchar(); //atrapa el enter del scanf
38     getchar(); //mantiene en espera la ejecución
39     return 0;
40 }
```



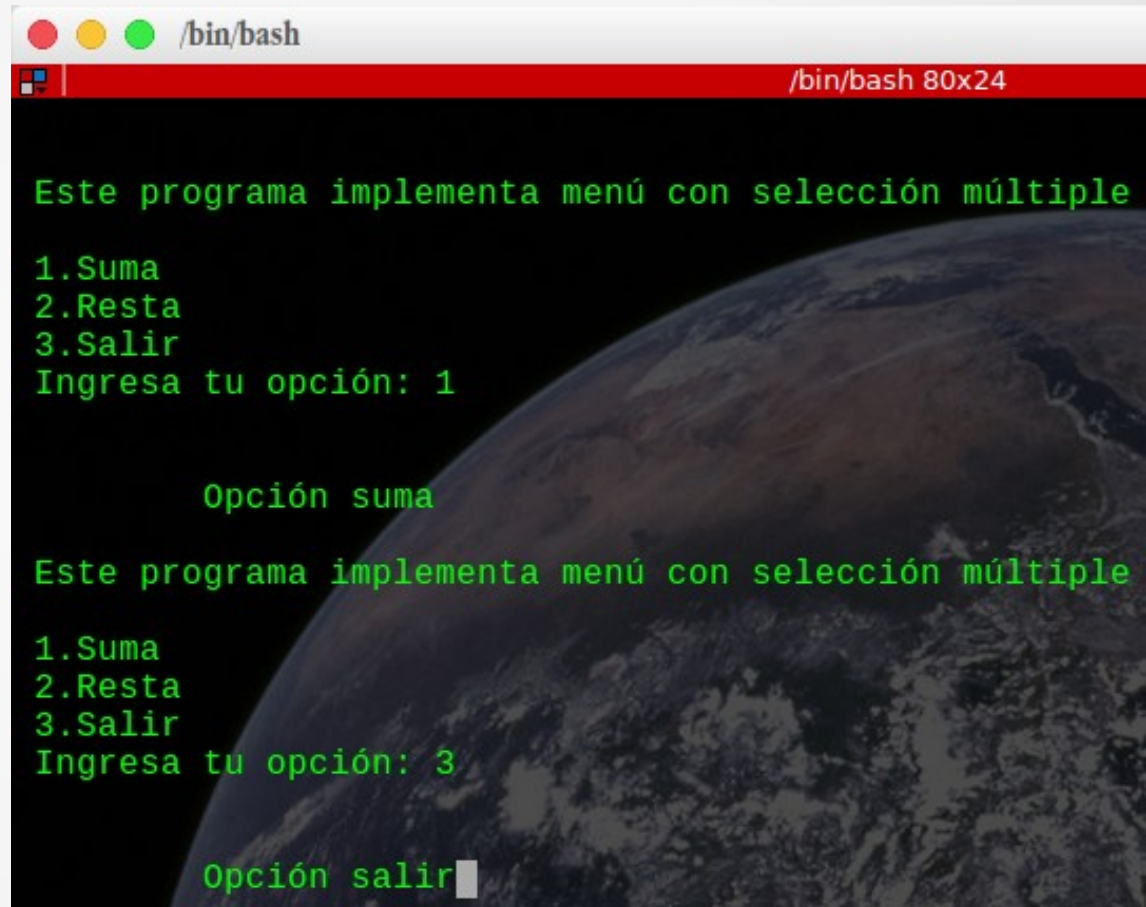
Hecho por Huicho :)

# Menú que se repite con do-while

```
1  /*programa menú con switch - case
2   * hecho por huicho*/
3
4  #include <stdio.h> //para printf, scanf y getchar
5  #include <stdlib.h> //para exit
6
7  int main(int argc, char* argv[])
8  {
9      short opcion=0;
10
11     //todo lo q quieran repetir va dentro del do
12     do
13     { //abre bloque do
14
15         printf("\n\n Este programa implementa men%c con selecci%cn m%cltiple \n\n", 163, 162, 163);
16         printf(" 1.Suma \n 2.Resta \n 3.Salir \n Ingresar tu opci%cn: ", 162);
17         scanf("%hd", &opcion);
18
19         switch(opcion) //evalúa variable opcion
20         { //abre bloque switch
21             case 1: //valida si opcion=1
22                 printf("\n\n\t Opci%cn suma", 162);
23                 break; //evita que se ejecute el caso siguiente
24
25             case 2: //valida si opcion=2
26                 printf("\n\n\t Opci%cn resta", 162);
27                 break; //evita que se ejecute el caso siguiente
28
29             case 3: //valida si opcion=3
30                 printf("\n\n\t Opci%cn salir", 162);
31                 getchar(); //atrapa el enter del scanf
32                 getchar(); //mantiene en espera la ejecución
33                 exit(0); //termina la ejecución justo aquí
34                 break; //evita que se ejecute el caso siguiente
35
36             default: //bloque a realizar si no fue ningún caso
37                 printf("\n\n\t Opci%cn no v%clida", 162, 160);
38                 break; //evita que se ejecute el caso siguiente
39         } //cierra bloque switch
40     } //cierre del bloque do
41     while(opcion!=3); //mientras el usuario presione diferente de 3 se repite
42
43     getchar(); //atrapa el enter del scanf
44     getchar(); //mantiene en espera la ejecución
45     return 0;
46 }
```

# Menú que se repite con do-while

- La impresión de las opciones del menú y el bloque `switch` se colocan dentro del bloque `do` porque queremos que se repita
- Mientras el usuario presione algo diferente a 3 se repetirá el contenido del bloque `do`
- No existe un incremento constante sino que depende de lo que suceda dentro del bloque
- `do-while` si lleva “,” después del paréntesis de la condición

A screenshot of a terminal window titled "/bin/bash" with a red title bar. The terminal displays a menu program. The first iteration shows the menu options: "1.Suma", "2.Resta", "3.Salir", followed by "Ingresa tu opción: 1". Then, it displays "Opción suma". The second iteration shows the same menu options, followed by "Ingresa tu opción: 3", and then "Opción salir". The background of the terminal is a dark image of the Earth from space.

```
/bin/bash
Este programa implementa menú con selección múltiple
1.Suma
2.Resta
3.Salir
Ingresa tu opción: 1

Opción suma

Este programa implementa menú con selección múltiple
1.Suma
2.Resta
3.Salir
Ingresa tu opción: 3

Opción salir
```

# Menú que se repite con do-while preguntando al usuario

- La condición a evaluar en el while deja de involucrar la opción del menú correspondiente a salir.
- Se pregunta al usuario si desea realizar otra operación del menú antes de cerrar la llave del bloque do.
- Opciones para repetir:
  - **(1 / 2)** → sencillo de evaluar y guardar en short
  - **(s / n)** → intermedio de evaluar por combinaciones y guardar en char
  - **(Si / No)** → complejo de evaluar por combinaciones, requiere guardar en cadena de caracteres e incluir biblioteca `string.h` para comparar



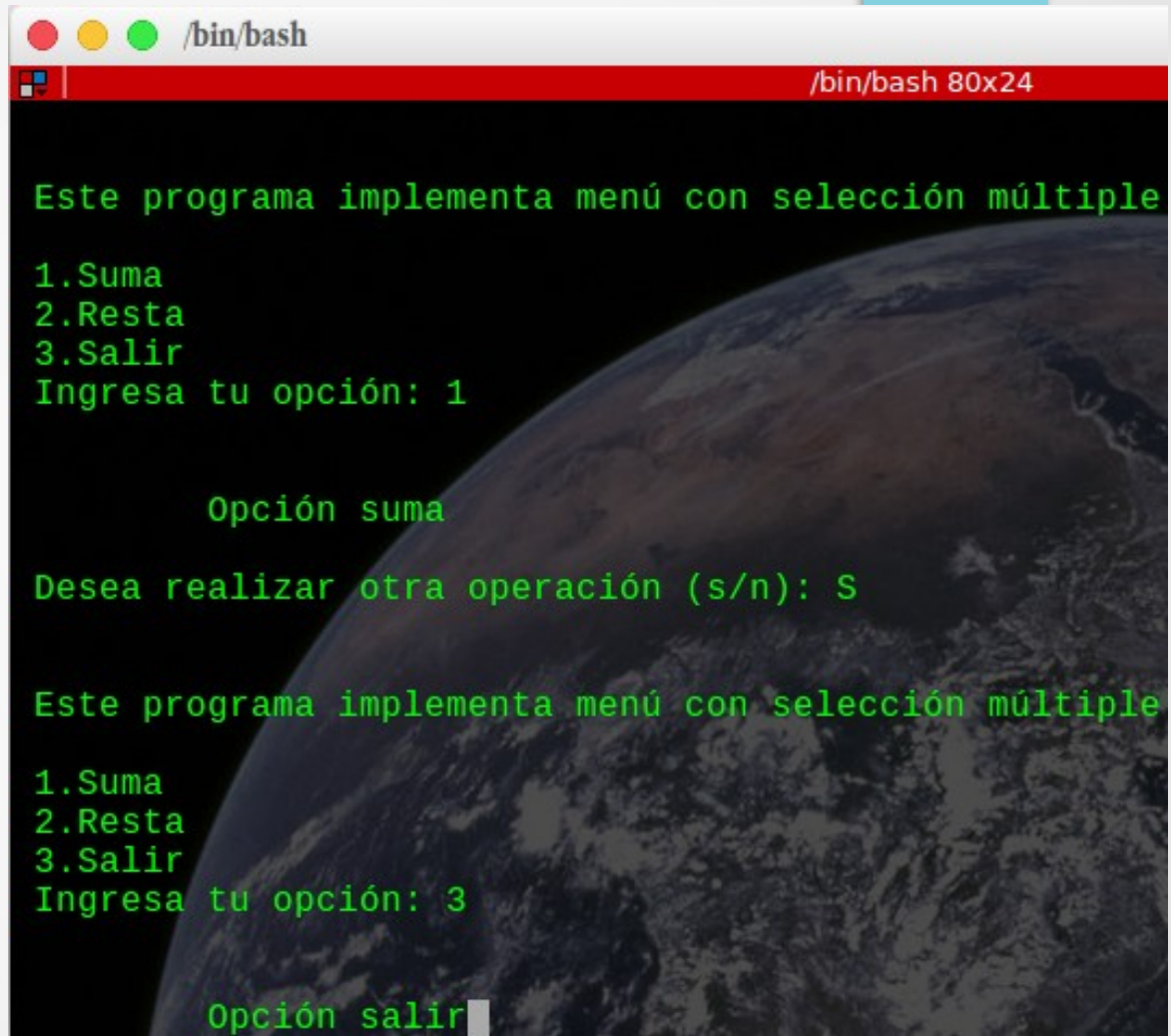
```

1  /*programa menú con switch - case que se repite preguntando a usuario
2  * hecho por huicho*/
3
4  #include <stdio.h> //para printf, scanf y getchar
5  #include <stdlib.h> //para exit
6
7  int main(int argc, char* argv[])
8  {
9      short opcion=0;
10     char repite='\0';
11
12     //todo lo q quieran repetir va dentro del do
13     do
14     { //abre bloque do
15
16         printf("\n\n Este programa implementa men%c con selecci%cn m%cltiple \n\n", 163, 162, 163);
17         printf(" 1.Suma \n 2.Resta \n 3.Salir \n Ingresa tu opci%cn: ", 162);
18         scanf("%hd", &opcion);
19
20         switch(opcion) //evalúa variable opcion
21         { //abre bloque switch
22             case 1: //valida si opcion=1
23                 printf("\n\n\t Opci%cn suma", 162);
24                 break; //evita que se ejecute el caso siguiente
25
26             case 2: //valida si opcion=2
27                 printf("\n\n\t Opci%cn resta", 162);
28                 break; //evita que se ejecute el caso siguiente
29
30             case 3: //valida si opcion=3
31                 printf("\n\n\t Opci%cn salir", 162);
32                 getchar(); //atrapa el enter del scanf
33                 getchar(); //mantiene en espera la ejecución
34                 exit(0); //termina la ejecución justo aquí
35                 break; //evita que se ejecute el caso siguiente
36
37             default: //bloque a realizar si no fue ningún caso
38                 printf("\n\n\t Opci%cn no v%clida", 162, 160);
39                 break; //evita que se ejecute el caso siguiente
40         } //cierra bloque switch
41         printf("\n\n Desea realizar otra operaci%cn (s/n): ", 162);
42         scanf(" %[^\\n]", &repite); //lee respuesta para repetir
43     } //cierre del bloque do
44     while(repite=='s' || repite=='S'); //mientras el usuario presione s se repite
45
46     getchar(); //atrapa el enter del scanf
47     getchar(); //mantiene en espera la ejecución
48     return 0;
49 }

```

# Menú que se repite con do-while preguntando al usuario

- Si la respuesta del usuario es “s” o “S” el menú se repetirá
- Si la respuesta es cualquier otro carácter el ciclo se romperá al no ser cierta la condición y ejecutará la instrucción posterior al while
- **do-while** si lleva “;” después del paréntesis de la condición

A screenshot of a terminal window titled "/bin/bash" with a red title bar. The terminal displays a menu program implemented with a do-while loop. The background of the terminal is a dark image of Earth from space. The text is in green. The first iteration shows the menu, the user selects option 1 (Suma), and is asked if they want to perform another operation, to which they respond 'S'. The second iteration shows the same menu, and the user selects option 3 (Salir), with the prompt "Opción salir" appearing below.

```
/bin/bash
Este programa implementa menú con selección múltiple
1.Suma
2.Resta
3.Salir
Ingresa tu opción: 1

Opción suma
Desea realizar otra operación (s/n): S

Este programa implementa menú con selección múltiple
1.Suma
2.Resta
3.Salir
Ingresa tu opción: 3

Opción salir
```