Adrià Vidosa

PROGRAMACIÓN

ÍNDICE

1. ESTRUCTURA GENERAL

Función principal

```
1  #include<stdio.h>
2
3  int main(){
4
5
6
7      return 0;
8  }
```

//comentario de 1 línea

/*

comentario de + líneas

*/

2. TIPOS DE VARIABLES

```
3  #include<stdio.h> //Libreria
4
5  #define PI 3.1416 //Macro
6
7  int y = 5; //Variable Global
8
9  int main(){
10     int x = 10; //Variable Local
11
12     float suma = 0;
13
14     suma = PI + x;
15
16     printf("La suma es: %f",suma);
17
```

1.Libreria

2.Macro

3.Variable Global

4.Variable Local

3. TIPOS DE DATOS/IDENTIFICADORES (char, float, …)

```
int main(){
    char a = 'e'; //tamaño= 1byte  Rango: 0..255
    short b = -15; //tamaño= 2bytes Rango: -128...127
    int c = 1024; //tamaño= 2bytes Rango: -32768...32767
    unsigned int d = 128; //tamaño = 2bytes Rango: 0...65535
    long e = 123456; //tamaño = 4bytes
    float f = 15.678; //tamaño = 4 bytes
    double m = 123123.123123; //tamaño = 8bytes
```

4. ENTRADAS Y SALIDAS DE DATOS

```
7
8       printf("Hola, ingresa un numero: \n");/*Se pide al usuario que ingrese
9       un valor por teclado y el valor ingresado de guarda directamente en la variable
10      numero*/
11
12      scanf("%d", &numero);//se escanea el valor ingresado y se guarda en la variable numero
13
14      printf("El valor que ingresaste es: %d\n", numero);/*se muestra en pantalla
15      el valor que ingresamos por teclado*/
16
```

5. OPERADORES DE ASIGNACIÓN

```
4
5   int main(){
6       int a,b,c;
7       a = 10;
8       //a += 10; // a = a + 10
9       //a -= 5; // a = a - 5
10      //a *= 2; //a = a * 2
11      a /= 2; // a = a / 2
12
```

=	Asignación
+=	Suma y asignación
-=	Resta y asignación
*=	Multiplicación y asignación
/=	División y asignación
%=	Módulo y asignación

También hay: <<= >>= &= ^=

6. OPERADORES ARITMÉTICOS (+-/*)

Operador =	Asignación
Operador *	Multiplicación
Operador /	División
Operador %	Resto de división entera (mod)
Operador +	Suma
Operador -	Resta

7. OPERADORES DE COMPARACIÓN Y LÓGICOS

Operadores	Uso de los símbolos	Significado
== (igual que)	a == b	Se cumple si 'a' es igual que 'b'
!= (distinto a)	a != b	Se cumple si 'a' es distinto que 'b'
< (menor que)	a < b	Se cumple si 'a' es menor que 'b'
> (mayor que)	a > b	Se cumple si 'a' es mayor que 'b'
<= (menor o igual que)	a <= b	Se cumple si 'a' es menor o igual que 'b'
>= (mayor o igual que)	a >= b	Se cumple si 'a' es mayor o igual que 'b'

Comparación

&&	Operador and (y)
||	Operador or (o)
!	Operador not (no)

Lógico

8. OPERADOR DE INTERROGACIÓN "?"

```
7
8  #include<stdio.h>
9
10 int main(){
11     int numero;
12
13     printf("Digite un numero: ");
14     scanf("%i",&numero);
15
16     (numero%2==0) ? printf("El numero es par") : printf("El numero es impar");
17
18
19     return 0;
20 }
```

El uso principal de este operador es el de tomar decisiones y arrojar los 2 posibles resultados (verdadero y falso), todo en una simple línea de código.

9. SENTENCIA “if”

```
if (n1 % n2 == 0){
    printf("El numero %i es divisible entre %i",n1,n2);
}
```

10. SELECCIOÓN DOBLE “if else”

```
if (condición) {
    sentencias_si_verdadero;
} else {
    sentencias_si_falso;
}
```

11. SENTENCIA “switch”

```
    switch(dia) {
        case 1 :
            printf("Lun, Lunes");
            break;
        case 2 :
            printf("Mar, Martes");
            break;
        case 3 :
            printf("Mier, Miercoles");
            break;
        case 4 :
            printf("Jue, Jueves");
            break;
        case 5 :
            printf("Vie, Viernes");
            break;
        case 6 :
            printf("Sab, Sabado");
            break;
        case 7 :
            printf("Dom, Domingo");
            break;
        default :
            printf("No existe");
}
```

12. BUCLE “while”

```
while (/*condicion*/) {
    /* Código */
}
```

```
int i = 0;
while (i < 100) {
    printf("%d\n", i);
    i = i + 1;
}
```

13. BUCLE “do while”

```
do {
    /* CODIGO */
} while (/* Condición de ejecución del bucle */)
```

```
int aleatorio;
do {
    aleatorio = rand();
} while (aleatorio != 25);
```

14. BUCLE “for”

```
for (/* inicialización */; /* condición */; /* incremento */) {
    /* código a ejecutar */
}
```

```
int i;
for (i=0; i < 100; i = i + 1) {
    printf("%d\n", i);
}
```

`i = i + 1` puede escribirse en forma más reducida como `i++`