Tema 2: Fundamentos de C

Oscar Perpiñán Lamigueiro - David Álvarez

1. Primeros pasos
2. Datos en C
   - Introducción
   - Almacenamiento de la Información
   - Números enteros
   - Números reales
   - Caracteres
3. Lectura de datos en C desde el teclado
4. Operadores
   - Tipos
   - Operaciones con variables
   - Conversión de tipos de datos

Hello World!

```c
#include <stdio.h>

int main()
{
  printf("Hello World!\n");

  return 0;
}
```

Hello World! (2)

```c
#include <stdio.h>

int main()
{
  printf("Hello World!\n");
  printf("¡Hola Mundo!\n");
  printf("Bonjour le Monde!\n");
  printf("Hallo Welt!\n");

  return 0;
}
```

Comentarios

```c
/** Este simple programa sirve para
   mostrar un mensaje en pantalla */
#include <stdio.h>

// Todo programa necesita una función main.
// Su contenido está delimitado entre llaves
int main()
{
  //La función printf muestra el mensaje en pantalla
  // Atención: el mensaje debe ir entre comillas
  printf("Hello World!\n");
  // La función main devuelve un entero con return.
  return 0;
} // Aquí acaba main y por tanto el programa
```

1 Primeros pasos

2 Datos en C
- Introducción
- Almacenamiento de la Información
- Números enteros
- Números reales
- Caracteres

3 Lectura de datos en C desde el teclado

4 Operadores
- Tipos
- Operaciones con variables
- Conversión de tipos de datos

1 Primeros pasos

2 Datos en C
- Introducción
- Almacenamiento de la Información
- Números enteros
- Números reales
- Caracteres

3 Lectura de datos en C desde el teclado

4 Operadores
- Tipos
- Operaciones con variables
- Conversión de tipos de datos

Constantes y Variables

Constantes datos cuyo valor no se puede modificar durante la ejecución del programa

Variables datos cuyo valor se puede modificar mediante el operador *asignación* (=)

Constantes y Variables

```
int main()
{
  // declara una variable con el identificador v1
  int v1;
  // declara una constante simbólica
  // con el identificador c1
  const int c1 = 4;
  // declara una variable v2,
  // y le asigna el valor 2 (una constante literal)
  int v2 = 2;
  // asigna el valor de la
  // constante c1 a la variable v1
  v1 = c1;
  // ídem con v2 (cambia su valor previo)
  v2 = c1;
  // error: c1 es una constante
  c1 = 3;
  return 0;
}
```

Nombres de constantes y variables

- Primer carácter: letra o carácter de subrayado (_) (**nunca un número**).
- Una o más letras (A-Z, a-z, *ñ excluida*), dígitos (0-9) o caracteres de subrayado.
- Tienen que ser distintos de las palabras clave.
- Las mayúsculas y las minúsculas son diferentes para el compilador.
- Es aconsejable que los nombres sean representativos

Palabras clave o reservadas

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

1 Primeros pasos

2 Datos en C
- Introducción
- Almacenamiento de la Información
- Números enteros
- Números reales
- Caracteres

3 Lectura de datos en C desde el teclado

4 Operadores
- Tipos
- Operaciones con variables
- Conversión de tipos de datos

bit

- Los ordenadores utilizan el sistema de numeración binario (dos dígitos, 0 y 1) para almacenar información.
- Un **dígito binario** (0 ó 1) se denomina *bit* (*binary digit*).
- Con **N bits** pueden representarse **2^N símbolos o 2^N números**
  - Ejemplo: con N = 8 bits se pueden representar los números positivos desde el 0 al 255 (2^8 - 1).

Representación de la información: binario y decimal

Ejemplo en decimal: 3452

10^3	10^2	10^1	10^0
1000	100	10	1
3	4	5	2

$$3452 = 3 \cdot 1000 + 4 \cdot 100 + 5 \cdot 10 + 2 \cdot 1$$

Ejemplo en binario: 10001111

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	**8**	**4**	**2**	**1**
1	0	0	0	**1**	**1**	**1**	**1**

$$128 + 8 + 4 + 2 + 1 = 143$$

No sólo números

- Cualquier información puede representarse con un conjunto de bits.
- ASCII (American Standard Code for Information Interchange): Estándar de 7 bits (128 caracteres), 95 caracteres imprimibles (del 32 al 126).

01001000	01101111	01101100	01100001
72	111	108	97
H	o	l	a

Unidades de almacenamiento

- Byte: 8 bits ($2^8 = 256$)
- Kilobyte (KB): 1024 bytes ($2^{10} = 1024$)
- Megabyte (MB): 1024 KB (2^{20} bytes = 2^{10} KB)
- Gigabyte (GB): 1024 MB (2^{30} bytes = 2^{10} MB = 2^{20} KB)
- ...

Tipos de datos

`int` números enteros

```
100 -41 0 12345
```

`float` y `double` números reales

```
3.0 101.2345 -0.0001 2.25e-3
```

`char` caracteres

```
's' '4' ';'
```

`_Bool` *booleanos*, 0 y 1

1 Primeros pasos

2 Datos en C
- Introducción
- Almacenamiento de la Información
- Números enteros
- Números reales
- Caracteres

3 Lectura de datos en C desde el teclado

4 Operadores
- Tipos
- Operaciones con variables
- Conversión de tipos de datos

Definición, asignación y printf con int

```c
#include <stdio.h>

int main()
{
  // int designa variable de números enteros
  int dia;
  // Asignamos un valor a la variable dia
  dia = 3;
  // Hacemos la asignación junto con la definición
  int mes = 2;
  // Usamos %i para números enteros
  printf("Hoy es día %i, del mes %i y año %i.\n", dia, mes, 2023);
  // Y también %d
  printf("Hoy es día %d, del mes %d y año %d.\n", dia, mes, 2023);

  return 0;
}
```

Rango de variables `int` con signo

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

int main() {
  printf("Un int ocupa %d bytes",
        sizeof(int));
  printf(" y abarca desde %d hasta %d.\n",
        INT_MIN, INT_MAX);
  printf("Un long int ocupa %d bytes",
        sizeof(long int));
  printf(" y abarca desde %ld hasta %ld.\n",
         LONG_MIN, LONG_MAX);
  printf("Un long long int ocupa %d bytes",
        sizeof(long long int));
  printf(" y abarca desde %lld hasta %lld.\n",
        LLONG_MIN, LLONG_MAX);
  return 0;
}
```

Rango de variables `int` sin signo

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

int main() {
  printf("Un unsigned int ocupa %d bytes",
        sizeof(unsigned int));
  printf(" y abarca desde 0 hasta %u.\n",
        UINT_MAX);
  printf("Un unsigned long int ocupa %d bytes",
        sizeof(unsigned long int));
  printf(" y abarca desde 0 hasta %lu.\n",
        ULONG_MAX);
  printf("Un unsigned long long int ocupa %d bytes",
        sizeof(unsigned long long int));
  printf(" y abarca desde 0 hasta %llu.\n",
        ULLONG_MAX);
 return 0;
}
```

1 Primeros pasos

2 Datos en C
- Introducción
- Almacenamiento de la Información
- Números enteros
- Números reales
- Caracteres

3 Lectura de datos en C desde el teclado

4 Operadores
- Tipos
- Operaciones con variables
- Conversión de tipos de datos

Variables y uso de printf con reales

```c
#include <stdio.h>

int main()
{
  // Usamos %f para números reales
  printf("Esto es un número real %f\n", 102.305);
  // Indicamos número de decimales explicitamente
  printf("escrito con dos decimales %.2f\n", 102.305);

  double num = 103.56e10;
  printf("Esto es un número real %f\n", num);
  printf("... en notación científica %e\n", num);
  printf("... y de forma automática %g\n", num);

  return 0;
}
```

Rango de números reales

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

int main() {

  printf("Un float ocupa %d bytes",
       sizeof(float));
  printf(" y abarca desde %e hasta %e.\n",
       FLT_MIN, FLT_MAX);

  printf("Un double ocupa %d bytes",
       sizeof(double));
  printf(" y abarca desde %e hasta %e.\n",
       DBL_MIN, DBL_MAX);

  return 0;
}
```

1 Primeros pasos

2 Datos en C
- Introducción
- Almacenamiento de la Información
- Números enteros
- Números reales
- Caracteres

3 Lectura de datos en C desde el teclado

4 Operadores
- Tipos
- Operaciones con variables
- Conversión de tipos de datos

Definición y asignación de caracteres

```c
#include <stdio.h>

int main()
{
  // Usamos char para asignar caracteres
  char letra = 'z';
  printf("La última letra es la %c\n", letra);
  printf("La primera letra es la %c\n", 'a');

  // Y char con un número?
  // Tabla ASCII
  char letra = 122;
  printf("La última letra es la %c\n", letra);

  return 0;
}
```

Uso de printf

```c
#include <stdio.h>

int main()
{
  // Usamos %c para caracteres
  // Atención: para delimitar caracteres usamos '
  printf("La última letra del alfabeto es la %c\n",
        'z');
  //Usamos %i para enteros
  printf("Su valor en la tabla ASCII es %i\n",
        'z');
  //Y si usamos %c para un número?
  printf("El número %i es la letra %c\n",
        122, 122);
  return 0;
}
```

1. Primeros pasos
2. Datos en C
   - Introducción
   - Almacenamiento de la Información
   - Números enteros
   - Números reales
   - Caracteres
3. Lectura de datos en C desde el teclado
4. Operadores
   - Tipos
   - Operaciones con variables
   - Conversión de tipos de datos

Lectura de números enteros con scanf

```c
#include <stdio.h>

int main()
{
  int num;

  printf("Escribe un número\n");

  //Atención: con scanf el nombre de la
  //variable debe ir precedido de &
  scanf("%i", &num);

  printf("Has escrito el número %i\n", num);

  return 0;
}
```

Errores comunes con `scanf`

- Escribir dentro de la cadena de control mensajes y secuencias de escape (p.ej. `\n`).
- Olvidar poner el operador **&** delante de los argumentos cuando son variables de los tipos básicos (`int`, `float`, `double`, `char`)
- Poner un especificador de formato no compatible con el tipo del argumento.

Lectura de números reales con scanf

Identificador de Formato

`float` `%f`

```
#include <stdio.h>

int main()
{
  float peso, altura;

  printf("Indica tu peso (kg) y altura (m)\n");
  scanf("%f %f", &peso, &altura);

  printf("Pesas %f kg, y mides %f m.\n",
     peso, altura);
  return 0;
}
```

Lectura de caracteres con scanf

```c
#include <stdio.h>

int main()
{
  char letra;
  printf("Escribe una letra\n");
  scanf("%c", &letra);
  printf("Has escrito letra %c\n", letra);
  return 0;
}
```

1 Primeros pasos

2 Datos en C
- Introducción
- Almacenamiento de la Información
- Números enteros
- Números reales
- Caracteres

3 Lectura de datos en C desde el teclado

4 Operadores
- Tipos
- Operaciones con variables
- Conversión de tipos de datos

1 Primeros pasos

2 Datos en C
- Introducción
- Almacenamiento de la Información
- Números enteros
- Números reales
- Caracteres

3 Lectura de datos en C desde el teclado

4 Operadores
- Tipos
- Operaciones con variables
- Conversión de tipos de datos

Aritméticos

```
x + y
x - y
x / y
x * y
x % y //módulo o resto de división de enteros
```

Relacionales

```
x == y
x != y
x > y
x >= y
x < y
x <= y
```

Lógicos

- AND, OR, NOT

```
x && y //AND
x || y //OR
!x //NOT, operador unario
```

- Operador condicional ? (ternario)

```
// expresión boleana ? valor si cierto : valor si falso
x > y ? "cierto" : "falso"
x == y ? "true" : "false"
```

Asignación

```
x = y
x += y // x = x + y
x -= y // x = x - y
x *= y // x = x * y
x /= y // x = x / y
x %= y // x = x % y
```

```
// ERROR: en el lado izquierdo no puede ir una expresión
x + y = 1
```

Incrementales

```
y = ++x // x = x + 1; y = x (preincremento)
y = x++ // y = x; x = x + 1 (postincremento)

y = --x // x = x - 1; y = x (predecremento)
y = x-- // y = x; x = x - 1 (postdecremento)
```

sizeof

Proporciona el tamaño de su operando en bytes.

```c
#include <stdio.h>

int main()
{
  int i1;
  float f1;
  double d1;
  char c1;

  printf("Un entero ocupa %d bytes\n", sizeof i1);
  printf("Un float ocupa %d bytes\n", sizeof f1);
  printf("Un double ocupa %d bytes\n", sizeof d1);
  printf("Un caracter ocupa %d bytes\n", sizeof c1);

  return 0;
}
```

Bits

```
x & y // Bits AND
x | y // Bits OR
x ^ y // Bits XOR
x ~ y // Bits NOT (complemento)
x << 1 // Desplazamiento de bits
x >> 1
```

1. Primeros pasos
2. Datos en C
   - Introducción
   - Almacenamiento de la Información
   - Números enteros
   - Números reales
   - Caracteres
3. Lectura de datos en C desde el teclado
4. Operadores
   - Tipos
   - Operaciones con variables
   - Conversión de tipos de datos

Aritméticos con enteros

```c
#include <stdio.h>

int main()
{
  int x, y, sum;
  x = 10;
  y = 15;
  sum = x + y;

  printf("La suma de %i con %i es %i\n",
         x, y, sum);
  return 0;
}
```

Aritméticos con caracteres

```
#include <stdio.h>

int main()
{
  char letra, Letra;
  letra = 'z';
  Letra = letra - 32;

  printf("La letra %c en mayúscula es %c\n",
        letra, Letra);
  return 0;
}
```

Aritméticos con números reales

```c
#include <stdio.h>

int main()
{
  float peso, altura, imc;

  printf("Indica tu peso (kg) y altura (m)\n");
  scanf("%f %f", &peso, &altura);

  imc = peso / (altura * altura);
  printf("Tu índice de masa corporal es %f\n", imc);

  return 0;
}
```

Operaciones de asignación

```c
#include <stdio.h>
int main() {
    int a, b = 3;

    a = 5;
    printf("a = %d\n", a);
    a *= 4; // a = a * 4
    printf("a = %d\n", a);
    a += b; // a = a + b
    printf("a = %d\n", a);
    a /= (b + 1); // a = a / (b+1)
    printf("a = %d\n", a);
    a = b = 1;
    printf("a = %d, b = %d\n", a, b);

    return 0;
}
```

Operaciones de incremento

```c
#include <stdio.h>

int main()
{
    int b = 2, r;
    //Preincremento
    r = ++b;
    printf("b = %d, r = %d\n", b, r);
    //Postincremento
    r = b++;
    printf("b = %d, r = %d\n", b, r);

    return 0;
}
```

Operaciones de incremento

```c
#include <stdio.h>

int main()
{
    int a = 0;
    printf("a = %d\n", ++a);
    printf("a = %d\n", a++);
    printf("a = %d\n", a);
    printf("a = %d\n", --a);
    printf("a = %d\n", a--);
    printf("a = %d\n", a);

    return 0;
}
```

Precedencia y asociatividad

```c
#include <stdio.h>

int main() {
    double a = 4, b = 7, c = 3, g = 9, result;

    result = a + b * c;
    printf( "resultado = %f\n", result);

    result = (a + b) * c;
    printf( "resultado = %f\n", result);

    result = a * b / c * g;
    printf("resultado = %f\n", result);

    result = (a * b) / (c * g);
    printf("resultado = %f\n", result);

    return 0;
}
```

Operaciones relacionales

```
#include <stdio.h>
int main()
{
    int x = 10, y = 3;

    printf("x igual a y = %d\n",
          (x == y));
    printf("x distinto a y = %d\n",
          (x != y));
    printf("x mayor que y = %d\n",
          (x > y));
    printf("x menor o igual a y = %d\n",
          (x <= y));
    printf("x mayor o igual que y = %d\n",
          (x >= y));
    return 0;
}
```

Operaciones lógicas

```c
#include <stdio.h>
int main()
{
    int a = 3, b = 2, c = 4, d = 5;

    printf("resultado = %d\n",
          (a > b) && (c < d));

    printf("resultado = %d\n",
          (a < 10) || (d != 5));

    printf("resultado = %d\n",
          (a != b) && (2 * d < 8));

    return 0;
}
```

Operaciones lógicas

```c
#include <stdio.h>
int main()
{
  int x, resto;

  printf("Escribe un número entero: ");

  scanf("%d", &x);

  // Calcula el resto de dividir por 2
  resto = x % 2;

  // Si el resto es 0, x es par.
  printf("Es un número %s\n",
        (resto == 0) ? "par" : "impar");

  return 0;
}
```

1 Primeros pasos

2 Datos en C
- Introducción
- Almacenamiento de la Información
- Números enteros
- Números reales
- Caracteres

3 Lectura de datos en C desde el teclado

4 Operadores
- Tipos
- Operaciones con variables
- Conversión de tipos de datos

Conversión implícita

Asignaciones el valor de la derecha se convierte al tipo de la variable de la izquierda (posible aviso o error).

```c
#include <stdio.h>

int main() {
  float f1 = 3.7, f2;
  int i1 = 2, i2;
  // Real a entero: pierde decimales
  i2 = f1;
  printf("Un real %f convertido a entero %d\n",
        f1, i2);
  // Entero a real: no cambia valor
  f2 = i1;
  printf("Un entero %d convertido a real %f\n",
        i1, f2);
  return 0;
}
```

Conversión explícita

Conversión explícita o forzada `(tipo) expresión`

```
#include <stdio.h>

int main()
{
  float f1 = 3.7, f2;
  int i1 = 2, i2;

  f2 = (float) i1;
  printf("Un entero %d convertido a real %f\n",
        i1, f2);

  i2 = (int) f1;
  printf("Un real %f convertido a entero %d\n",
        f1, i2);
  return 0;
}
```

Conversión en expresiones

Expresiones los valores de los operandos se convierten al tipo del operando que tenga la precisión más alta.

```
#include <stdio.h>

int main()
{
  double f1 = 100;
  int i1 = 150, i2 = 100;
  printf("Un entero, %d, dividido por un real, %f,",
        i1, f1);
  printf(" produce un real, %f\n",
        i1 / f1);
  printf("Un entero, %d, por un entero, %d: %d\n",
        i1, i2, i1 / i2);
  return 0;
}
```