PROGRAMACIÓN 1

Grado en Ingeniería Informática e I2ADE

Cambios de C a C++ en Programación 2

Dept. de Ciència de la Computació i Intel·ligència Artificial
Dpto. de Ciencia de la Computación e Inteligencia Artificial

Universitat d'Alacant
Universidad de Alicante

Índice

1. Introducción

El **lenguaje de programación C++**:

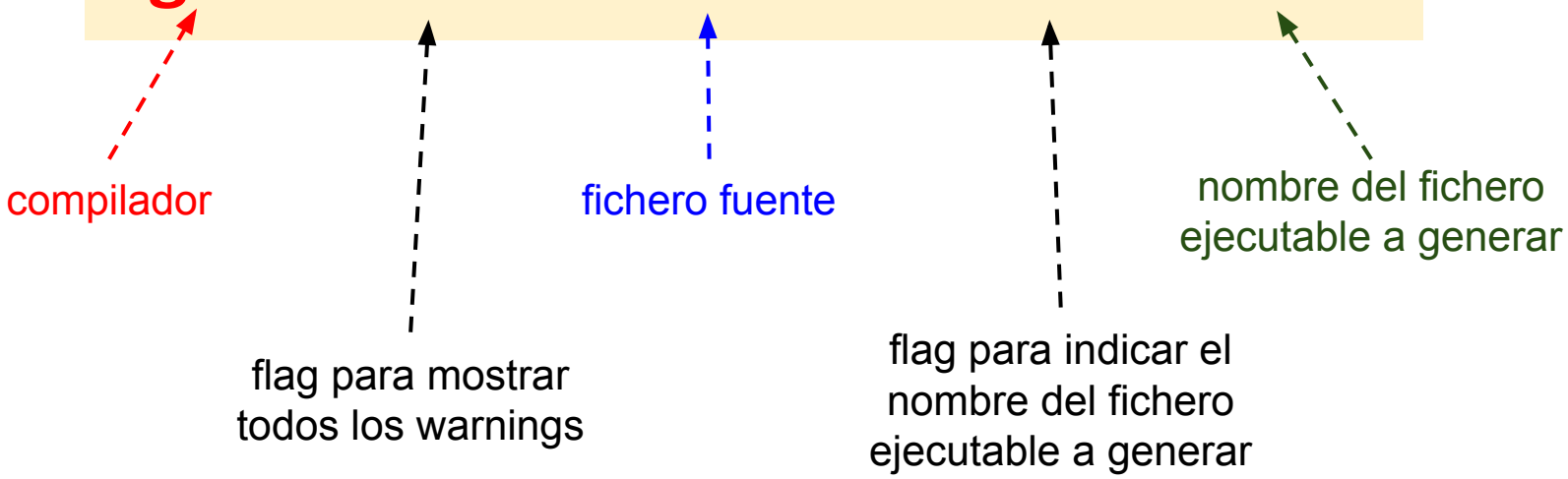
- Está **basado en C**, por lo que es un lenguaje de bajo nivel y de propósito general
- Se caracteriza por ser un lenguaje **orientado a objetos**
- Proporciona una **mayor variedad de librerías y herramientas** que C
- Permite **sobrecarga de operadores**
- Es **compatible con C**, pero C no es compatible con C++

2. Compilador

- El compilador de C++ pasa a ser **g++**
- La extensión de los ficheros fuente en C++ pasa a ser **.cc**, aunque también admite otras como .cpp

Ejemplo de compilación:

```
g++ -Wall fuente.cc -o fuente
```



3. Librerías y espacios de nombres

Librerías

Aunque C++ es compatible con las librerías de C (`stdlib.h`, `stdio.h`, …) existe una versión específica de las librerías para C++, que se escriben con “c” delante y sin la extensión `.h`.

C	C++
`<stdlib.h>`	`<cstdlib>`
`<time.h>`	`<ctime>`
`<math.h>`	`<cmath.h>`
`<string.h>`	`<cstring>`

3. Librerías y espacios de nombres

- **Espacio de nombres**
  - Característica agregada a C++ que no tiene C.
  - Es una región declarativa que proporciona un ámbito a los identificadores (nombres de tipos, funciones, variables, etc.) de su interior.
  - Se utilizan para organizar el código fuente en grupos lógicos y evitar conflictos de nombres, especialmente, cuando se utilizan varias librerías.

3. Librerías y espacios de nombres

Espacio de nombres

Cómo acceder a identificadores de un espacio de nombres

- Utilizando el nombre completo del identificador:

```
espacio_de_nombres::identificador
```

Ejemplo: `std::cout;`

- Utilizando la palabra reservada **`using`**:
  - Como declaración para un identificador único:

```
using espacio_de_nombres::identificador;
```

Ejemplo: `using std::cout;`

  - Como directiva para todos los identificadores del espacio de nombres:

```
using namespace espacio_de_nombres;
```

Ejemplo: `using namespace std;`

3. Librerías y espacios de nombres

Espacio de nombres

Ejemplo:

```
#include<iostream>

using namespace std; // usando la directiva para tener acceso a todos
                     // los identificadores del espacio de nombres
int main() {
    int edad;
    string nombre;

    cout << "Dime tu nombre: ";
    getline(cin, nombre);

    cout << "Dime tu edad: ";
    cin >> edad;

    cout << "Bienvenido " << nombre << ", tienes " << edad << " años" << endl;

    return 0;
}
```

4. Sentencias de entrada y salida estándar

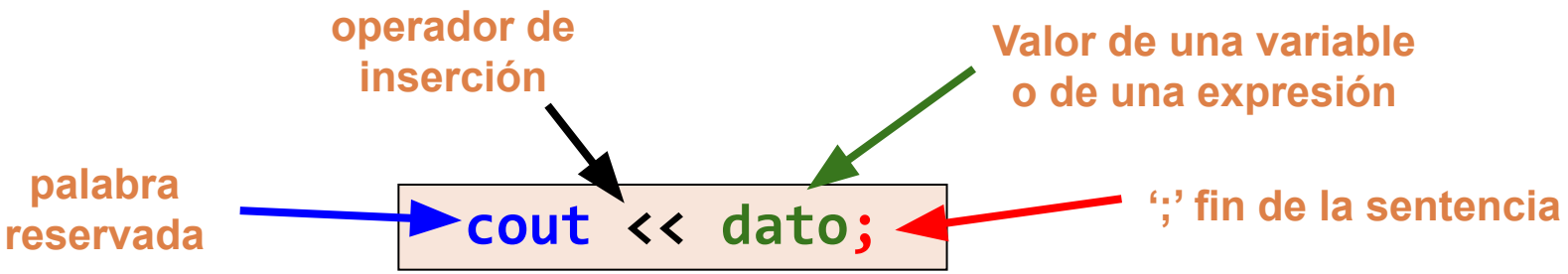
- En C++ la entrada y salida estándar se ha simplificado.
- Hay que incluir la librería **<iostream>**.
- Equivalencias:

C	C++
<stdio.h>	<iostream>
printf	cout
scanf	cin

4. Sentencias de entrada y salida estándar

cout

Permite escribir en pantalla cualquier combinación de valores de variables, constantes, expresiones y cadenas de texto.



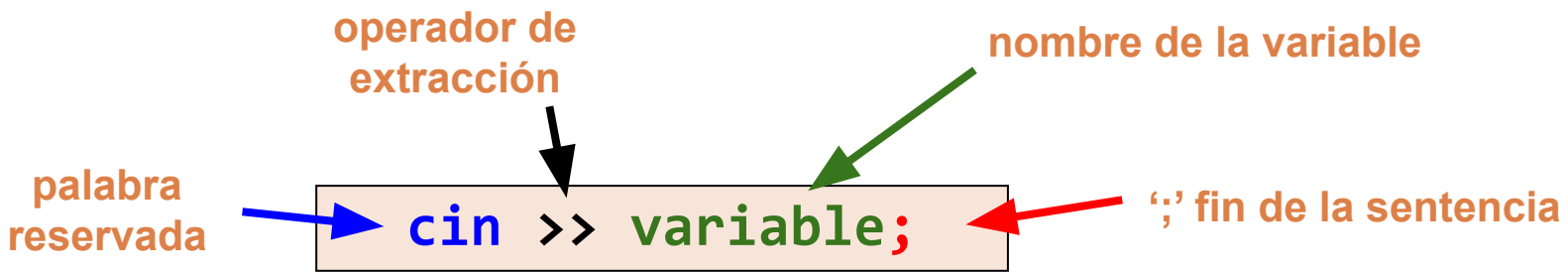
Ejemplos

```
cout << "El precio del ordenador portátil es de " << precio << " euros" << endl;
cout << "El precio total es: " << (precio1 + precio2);
cout << precio;
cout << "esto es una cadena de texto sin salto a una nueva línea";
cout << "esto es una cadena de texto con salto a una nueva línea\n";
cout << endl;
cout << "\n";
```

4. Sentencias de entrada y salida estándar

cin

Permite leer cualquier información introducida por teclado y guardarla en una variable.



Ejemplos

```
cout << "Introduce tu edad:";
cin >> edad;   // edad será una variable declarada de tipo int
cout << "introduce las notas de los dos parciales de prácticas:";
cin >> nota1 >> nota2;  // nota1 y nota2 serán variables declaradas de tipo
                        // float o double
cout << "¿Deseas introducir más datos? (S/N) : ";
cin >> respuesta;   // respuesta será una variable declarada de tipo char
```

5. Paso de parámetros por referencia

- En C++ el paso de parámetros por valor se hace igual que en C.
- En C++ se ha simplificado el uso de parámetros por referencia: sólo hay que poner **&** en los parámetros en la declaración del módulo.

Ejemplo:

```
void pedirDatos(string &nom, int &ed) {
    cout << "Dime tu nombre: ";
    getline(cin, nom);

    cout << "Dime tu edad: ";
    cin >> ed;
}
```

```
int main() {
    int edad;
    string nombre;

    pedirDatos(nombre, edad);

    cout << "Bienvenido " << nombre << ", tienes " << edad << " años" << endl;

    return 0;
}
```

6. Registros

- En C++ no es necesario el uso de `typedef` para evitar tener que incluir `struct` en la declaración de una variable de tipo registro.

  Ejemplo:

```c
// Lenguaje C
#include<stdio.h>

typedef struct {
    int x, y;
} TPunto;

int main() {

    TPunto p1, p2;

    return 0;
}
```

```cpp
// Lenguaje C++
#include<iostream>

struct TPunto {
    int x, y;
};

int main() {

    TPunto p1, p2;

    return 0;
}
```