ESTRUCTURAS DE DATOS

NOTAS SOBRE C++

Plantillas en funciones

Manuel Montenegro Montes

Departamento de Sistemas Informáticos y Computación
Facultad de Informática – Universidad Complutense de Madrid

Ejemplo

Implementamos una función que calcula el mínimo de dos enteros:

```
int min(int a, int b) {
   if (a ≤ b) {
     return a;
   } else {
     return b;
   }
}
```

- ¿Y si quiero calcular el mínimo de dos float? ¿y el mínimo de dos double?
- ¿Y si quiero calcular el mínimo de dos string utilizando el orden lexicográfico? Por ejemplo: min("AA", "AB") = "AA".

Ejemplo

```
float min(float a, float b) {
                                                                         double min(double a, double b) {
int min(int a, int b) {
 if (a \leq b) {
                                     if (a \leq b) {
                                                                           if (a \leq b) {
                                       return a:
                                                                             return a:
    return a;
                                     } else {
                                                                           } else {
  } else {
    return b:
                                       return b;
                                                                             return b;
const std::string & min(const std::string &a, const std::string &b) {
 if (a \leq b) {
    return a;
  } else {
    return b;
```

- iCuanta duplicidad!
- Todas tienen la misma implementación. iSolo difieren en los tipos!

Programación genérica

 Sería deseable tener una única versión genérica, que pudiese funcionar con varios tipos.

```
??? min( ??? a, ??? b) {
  if (a \le b) {
    return a;
  } else {
    return b;
  }
}
```

Solución: plantillas (templates) en C++.

Plantillas en C++

- Son definiciones con «huecos» (parámetros de plantilla).
- Se especifican mediante la palabra template, seguida de los parámetros de plantilla, y seguida de la definición de función paramétrica.

```
template <typename T>
T min(T a, T b) {
   if (a \le b) {
      return a;
   } else {
      return b;
   }
}
```

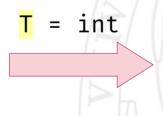
Llamada a funciones plantilla

Basta con indicar el tipo con el que queremos «rellenar» el marcador.

```
min<int>(6, 2)
min<double>(3.3, 5.5)
min<std::string>("Pepito", "Paula")
```

 Cada vez que se hace una llamada a la función genérica, se hace una versión específica para el tipo indicado en el marcador. A esto se le llama instanciación de plantillas.

```
template <typename T>
T min(T a, T b) {
  if (a \le b) {
    return a;
  } else {
    return b;
  }
}
```



```
int min<int>(int a, int b) {
  if (a \le b) {
    return a;
  } else {
    return b;
  }
}
```

Instanciación de plantillas

```
template <typename T>
    T min(T a, T b) {
      if (a \le b) {
        return a;
                                      T = int
      } else {
        return b;
                                     T = double
std::string min(std::string a, std::string b) {
  if (a \le b) {
    return a;
  } else {
    return b;
```

```
int min<int>(int a, int b) {
      if (a \leq b) {
        return a;
      } else {
        return b;
double min<double>(double a, double b) {
  if (a \leq b) {
    return a;
   else {
    return b;
```

Instanciación de plantillas

 En esta última instanciación (con un string) podemos indicar un tipo más preciso:

```
std::cout << min<const std::string &>("Pepito", "Ramiro") << std::endl;</pre>
```

O bien modificar nuestra función genérica:

```
template <typename T>
const T & min(const T &a, const T &b) {
  if (a \le b) {
    return a;
  } else {
    return b;
  }
}
```

Deducción de argumentos de plantilla

 Cada vez que hemos llamado a una función genérica, hemos indicado el tipo con el que debe instanciarse:

```
std::cout << min<std::string>("Pepito", "Ramiro") << std::endl;</pre>
```

- C++ permite omitirlo en la mayoría de los casos.
 - En ese caso intenta deducir el argumento de la plantilla.

```
std::cout << min("Pepito", "Ramiro") << std::endl;</pre>
```



iCuidado con las instanciaciones!

¿Qué pasa si instancio la plantilla con dos complejos?

```
Complejo z1(1.0, 3.0), z2(4.0, -5.0);
std::cout << min(z1, z2) << std::endl;
```

C++ realiza esta instanciación:

```
template <typename T>
const T & min(const T &a, const T &b) {
  if (a \le b) {
    return a;
  } else {
    return b;
  }
}
```

iCuidado con las instanciaciones!

 Los errores provocados por instanciaciones incorrectas suelen ser crípticos, largos, y difíciles de interpretar: