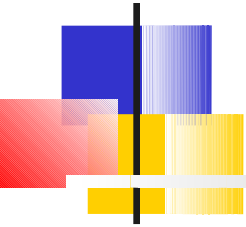


Plantillas (*Templates*)



Programación Avanzada. Cuatrimestre 1 Año 2009



Programación genérica

- Paradigma de programación centrado en los algoritmos más que en los datos.

- Generalización

Significa que, en la medida de lo posible, los algoritmos deben ser parametrizados al máximo y expresados de la forma más independiente posible de detalles concretos, permitiendo así que puedan servir para la mayor variedad posible de tipos y estructuras de datos.

Comparación de paradigmas de programación



- Orientada al dato

Representemos un tipo de dato genérico (por ejemplo **int**) que permita representar objetos con ciertas características comunes (peras y manzanas por ejemplo). Definamos también que operaciones pueden aplicarse a este tipo (por ejemplo aritméticas) y sus reglas de uso, independientemente que el tipo represente peras o manzanas en cada caso.

- Funcional

Construyamos un algoritmo genérico (por ejemplo **sort**), que permita representar algoritmos con ciertas características comunes (ordenación de cadenas alfanuméricas y vectores por ejemplo). Definamos también a que tipos pueden aplicarse a este algoritmo y sus reglas de uso, independientemente que el algoritmo represente la ordenación de cadenas alfanuméricas o vectores.



¿Para qué templates?

Si se implementa el comportamiento una y otra vez se reinventa la rueda. Los errores que pueden existir en un código pueden llevarse a otros y la corrección es dificultosa.

de datos.

Si se escribe código para tipos genéricos se pierde la ventaja del control de tipos que hace el lenguaje. Las clases bases hacen en general más difícil de mantener el código.

Si se utilizan preprocesadores se pierde la ventaja del trabajar con formato de código propio del lenguaje.



Template son la solución

- Los templates son funciones o clases definidas para tipos no especificados.
- Al utilizarlo se pasa el tipo como un argumento ya sea explícita o implícitamente.
- Al ser componentes del lenguaje proveen control total de tipos y de ámbito.



Funciones Template

- Proveen un comportamiento que puede ser invocado con distintos tipos.
- Tiene una representación similar a una función.

```
template <typename T>
inline T const& max (T const& a, T const& b)
{
    // if a < b then use b else use a
    return a < b ? b : a;
}
```



Deducción de argumentos

- El sistema no hace la conversión automática de los tipos que uno pasa.

Por ejemplo:

`max(4,4.3)` Generará un error!!

`max(static_cast<double>(4),4.3)` //bien

- Se puede calificar la función

`max<double>(4,4.3);`

Sobrecarga de funciones template



- Se aplican las mismas reglas que a la sobrecarga de funciones.
- Las funciones no template pueden coexistir con funciones template pero ellas son preferidas.
- Revisar ejemplo max2.cpp



Clases template

- Similares a las funciones template.
- Caso típico de clases contenedoras.
- Pueden indicarse con `typename` o `class` indistintamente

```
template <typename T>
class conjunto{
private:
    std::vector<T> set;
public:
    void agregar(T x);
    conjunto<T> unionC(conjunto<T> c);
    conjunto<T> interseccionC(conjunto<T> c);
    bool pertenencia(T x);
    void mostrar();
    conjunto<T> operator *(conjunto<T> c);
    friend std::istream & operator
    >>(std::istream & i, conjunto<T> & c);
};
```

Implementación de las funciones miembro

- Se debe indicar que se trata de una función template.
- También deben indicarse explícitamente el tipo de datos que maneja.

```
template <class T>
void conjunto<T>::agregar(T x){
    bool esta=false;
    for(int k=0;(!esta && k<set.size());k++)
        if (set[k]==x) esta=true;
    if (!esta) set.push_back(x);
}
template <typename T>
conjunto<T> conjunto<T>::unionC
(conjunto<T> c){
    conjunto a=c; int k;
    for(k=0;k<set.size();k++)
        a.agregar(set[k]);
    return a;
}
```



Revisión de ejemplos

- Comentarios de la implementaciones:
 - Complejo
 - Stack1 – Manejo de excepciones
 - Stack2 – Especialización de clases template
 - Stack3 – Templates con valores por defecto.
 - Stack4 – Templates con parámetros no tipos de datos.