(Plantillas)

TEMPLATES

Ejemplo

Plantillas

Código

Ejemplo C++

```
int duplicar(int data)
{
    cout << "El doble es: " << data * 2 << endl;
    return data*2;
}
```

Ejemplo C++

```
int duplicar(int data)
{
   return data*2;
}
```

¿Qué se debe hacer para que reciba algo de tipo double?

Ejemplo C++

```
double duplicar(double data)
{
   return data*2;
}
```

Ejemplo C++

```
template <class TIPO>
template <typename TIPO>
```

Ejemplo C++

```
template <class TIPO>
template <typename TIPO>
```

Ejemplo C++

```
template <typename TIPO>
TIPO duplicar(TIPO data)
{
   return data*2;
}
```

Ejemplo C++

```
template <typename TIPO>
TIPO duplicar(TIPO data)
{
    return data*2;
}
```

Usándolo

```
cout<<duplicar(4);
cout<<duplicar(3.34);
cout<<duplicar(duplicar(55));
```

Usando múltiples templates

```
template <typename TIPO1, typename TIPO2>
void sumar(TIPO1 dato1,TIPO2 dato2)
{
    cout<<"La suma es: "<<dato1+dato2;
}
```

Usándolo

```
sumar(4,2.5);
sumar<double,int>(1.2,56);
```

Usándolo

Implícita

```
sumar(4,2.5);
sumar<double,int>(1.2,56);
```

Usándolo

```
sumar(4,2.5);
sumar<double,int>(1.2,56);
```

Explícita

Templates en structs

```
template <typename T>
struct Contenedor{
   T datoT;
   int datoint;
   string datostring;
};
```

Templates en structs

```
template <typename T>
struct Contenedor{
   T datoT;
   int datoint;
   string datostring;
};
```

Usándolo

```
int main()
{
        Contenedor <string> contString;
        Contenedor <int> contInt;
        Contenedor <double> contDouble;

        contString.dato1 = "Un String";
        contInt.dato1 = 4;
        contDouble.dato1 = 3.1416;

}
```

Usándolo

```
int main()
{
        Contenedor <string> contString;
        Contenedor <int> contInt;
        Contenedor <double> contDouble;

        contString.dato1 = "Un String";
        contInt.dato1 = 4;
        contDouble.dato1 = 3.1416;

}
```

Usándolo

```
int main()
{
        Contenedor <string> contString;
        Contenedor <int> contInt;
        Contenedor <double> contDouble;

        contString.dato1 = "Un String";
        contInt.dato1 = 4;
        contDouble.dato1 = 3.1416;

}
```

Usándolo

```
int main()
{
        Contenedor <string> contString;
        Contenedor <int> contInt;
        Contenedor <double> contDouble;

        contString.dato1 = "Un String";
        contInt.dato1 = 4;
        contDouble.dato1 = 3.1416;

}
```

Tipos Complejos

- Es posible con el mismo *template* alojar tipos complejos de dato.
- EJ:

```
template <typename T>
struct Contenedor{
   T dato1;
   int datoint;
   string datostring;
};

struct Fecha{
            int dia;
            int mes;
            int anio;
};
```

```
int main()
{
        Contenedor <string> contString;
        Contenedor <int> contInt;
        Contenedor <double> contDouble;
        Contenedor <Fecha> contFecha;

        contString.dato1 = "Un String";
        contInt.dato1 = 4;
        contDouble.dato1 = 3.1416;

        contFecha.dato1.dia = 20;
        contFecha.dato1.mes = 9;
        contFecha.dato1.anio = 2013;

        cout<<contString.dato1<<endl;
        cout<<contInt.dato1<<endl;
        cout<<contDouble.dato1<<endl;
        cout<<endl;
        cout<<contFecha.dato1.dia<<endl;
        cout<<contFecha.dato1.mes<<endl;
        cout<<contFecha.dato1.anio<<endl;
        return 0;
}
```

Estructura Recursiva

```
int main()
{
    Contenedor < Contenedor < string > > contContInt;

    contContInt.dato1.dato1="string dato1 de dato1";
    contContInt.dato1.datoint = 4;
    contContInt.dato1.datostring = "string dato1";

    contContInt.datoint = 5;
    contContInt.datostring = "tercer String";

    cout<<contContInt.dato1.dato1<<endl;
    cout<<contContInt.dato1.datoint<<endl;
    cout<<contContInt.dato1.datostring<<endl;
    cout<<endl;
    cout<<contContInt.datoint<<endl;
    cout<<contContInt.datostring<<endl;
}
```

Múltiple Template

```
template <typename T, typename U, typename V>
struct Contenedor{
  T dato1;
  U dato2;
  V dato3;
};
```

```
int main()
{
    Contenedor <int,float,string> c1;
    c1.dato1 = 5;
    c1.dato2 = 4.3;
    c1.dato3="Hola";
    cout<<c1.dato1<<endl;
    cout<<c1.dato2<<endl;
    cout<<c1.dato3<<endl;

    Contenedor<Contenedor<int,float,string>,int,bool> c2;
    c2.dato1.dato1 = 4;
    c2.dato1.dato2 = 3.2;
    c2.dato1.dato3 = "Inception";
    c2.dato2=4;
    c2.dato3=false;

    cout<<c2.dato1.dato1<<endl;
    cout<<c2.dato1.dato2<<endl;
    cout<<c2.dato1.dato3<<endl;
    cout<<c2.dato2<<endl;
    cout<<c2.dato3<<endl;

    return 0;
}
```