

# Structs y Member Classes

2 de junio de 2020

# Struct

```
struct Datos {  
    int a;  
    string b;  
};  
  
int main() {  
    Datos d;  
    d.a = 5;  
    d.b = "30";  
    cout << d.b << endl; // "30"  
}
```

# Struct

```
class Datos {  
    int a;  
    string b;  
};  
  
int main() {  
    Datos d;  
    d.a = 5;  
    d.b = "30";  
    cout << d.b << endl; // "30"  
}
```

```
main.cpp: In function 'int main()':  
main.cpp:12:6: error: 'int Datos::a' is private within this context  
    ds.a = 5;  
    ^  
main.cpp:6:7: note: declared private here  
    int a;  
    ^
```

# Struct

```
struct Datos {  
    private:  
    int a;  
    string b;  
};  
  
int main() {  
    Datos d;  
    d.a = 5;  
    d.b = "30";  
    cout << d.b << endl; // "30"  
}
```

```
main.cpp: In function 'int main()':  
main.cpp:12:6: error: 'int Datos::a' is private within this context  
    ds.a = 5;  
    ^  
main.cpp:6:7: note: declared private here  
    int a;  
    ^
```

```
struct Nodo {  
    int valor;  
    Nodo* siguiente;  
};  
  
class Lista {  
  
    public:  
  
        // ...  
  
        void agregar(int v);  
  
    private:  
  
        Nodo* primero_;  
  
};  
  
void Lista::agregar(int v) {  
    // caso vacío  
    primero_ = new Nodo();  
    primero_.valor = v;  
    primero_.siguiente = nullptr;  
}
```

```

struct Nodo {
    int valor;
    Nodo* siguiente;

    Nodo(int valor);
};

Nodo::Nodo(int v) : valor(v), siguiente(nullptr) {}

class Lista {

public:

    // ...

    void agregar(int v);

private:

    Nodo* primero_;

};

void Lista::agregar(int v) {
    // caso vacío
    primero_ = new Nodo(v);
}

```

```
struct Nodo {  
    int valor;  
    Nodo* izq;  
    Nodo* der;  
  
    Nodo(int valor);  
};  
  
class Conj {  
  
    public:  
  
        // ...  
  
        void insertar(int v);  
  
    private:  
  
        Nodo* primero_;  
  
};
```

```

class Lista {

    public:

        // ...

        void agregar(int v);

        struct Nodo {
            int valor;
            Nodo* siguiente;

            Nodo(int valor);
        };

    private:

        Nodo* primero_;

};

Lista::Nodo::Nodo(int v) : valor(v), siguiente(nullptr) {}

void Lista::agregar(int v) {
    // caso vacío
    primero_ = new Nodo(v);
}

```



```
int main() {  
    Nodo x(4)  
}
```

```
lista.cpp: In function 'int main()':  
lista.cpp:30:3: error: 'Nodo' was not declared in this scope  
    Nodo n;  
    ^~~~~
```

```
int main() {  
    Lista::Nodo x(4)  
}
```

```

class Lista {

    public:

        // ...

        void agregar(int v);

    private:

        struct Nodo {
            int valor;
            Nodo* siguiente;

            Nodo(int valor);
        };

        Nodo* primero_;

};

Lista::Nodo::Nodo(int v) valor(v), siguiente(nullptr) {}

void Lista::agregar(int v) {
    // caso vacío
    primero_ = new Nodo(v);
}

```

```
int main() {  
    Lista::Nodo x(4)  
}
```

```
lista.cpp: In function 'int main()':  
lista.cpp:31:10: error: 'struct Lista::Nodo' is private within this context  
    Lista::Nodo n;  
           ^~~~~
```

```

class Lista {

    public:

        // ...

        void agregar(int v);

    private:

        Nodo* primero_;

        struct Nodo {
            int valor;
            Nodo* siguiente;

            Nodo(int valor);
        };
};

```

```

lista.cpp:16:5: error: 'Nodo' does not name a type
    Nodo* primero_;
    ^~~~~

```

# Friendness

```
class Lista {
public:
    // ...
    void agregar(int v);

    friend int main();

private:

    struct Nodo {
        int valor;
        Nodo* siguiente;

        Nodo(int valor);
    };

    Nodo* primero_;

};

int main() {
    Lista::Nodo n(5);
    Lista l;
    l.primero_ = nullptr;
}
```

# Friendness

```
class Lista {
public:
    // ...

    void agregar(int v);

    friend ostream& operator<<(ostream& os, const Lista& l) {
        l.mostrar(os);
    }

private:

    void mostrar(ostream& os);

    struct Nodo {
        int valor;
        Nodo* siguiente;
    };

    Nodo* primero_;
};

void Lista::mostrar(ostream& os) {
    // while ...
}
```