En C++

```
// Interfaz del TAD. Pre-declaraciones:

template <typename Elemento> struct  Cola;

template <typename Elemento> void vacia(Cola<Elemento>& c);
template <typename Elemento> void encolar(Cola<Elemento>& c, const Elemento& dato);
template <typename Elemento> void desencolar(Cola<Elemento>& c);
template <typename Elemento> void primero(const  Cola<Elemento>& c, Elemento& dato, bool& error);
template <typename Elemento> bool esVacia(const  Cola<Elemento>& c);
template <typename Elemento> int longitud(const  Cola<Elemento>& c);
template <typename Elemento> void duplicar(const  Cola<Elemento>& cOrigen, Cola<Elemento>& cDestino);
template <typename Elemento> bool operator== (const Cola<Elemento>& c1, const Cola<Elemento>& c2);
template <typename Elemento> void liberar(Cola<Elemento>& c);
template <typename Elemento> void iniciarIterador(Cola<Elemento>& c);
template <typename Elemento> bool existeSiguiente(const Cola<Elemento>& c);
template <typename Elemento> bool siguiente(Cola<Elemento>& c, Elemento& dato);

…
```

En C++

```
// Declaración

template <typename Elemento> struct Cola{

friend void vacia<Elemento>(Cola<Elemento>& c);
friend void encolar<Elemento>(Cola<Elemento>& c, const Elemento& dato);
friend void desencolar<Elemento>(Cola<Elemento>& c);
friend void primero<Elemento>(const Cola<Elemento>& c, Elemento& dato, bool& error);
friend bool esVacia<Elemento>(const Cola<Elemento>& c);
friend int longitud<Elemento>(const Cola<Elemento>& c);
friend void duplicar<Elemento>(const  Cola<Elemento>& cOrigen, Cola<Elemento>& cDestino);
friend bool operator==<Elemento> (const Cola<Elemento>& c1, const Cola<Elemento>& c2);
friend void liberar<Elemento>(Cola<Elemento>& c);
friend void iniciarIterador<Elemento>(Cola<Elemento>& c);
friend bool existeSiguiente<Elemento>(const Cola<Elemento>& c);
friend bool siguiente<Elemento>(Cola<Elemento>& c, Elemento& dato);
…
```

En C++

```
// Representación de los valores del TAD

private:
   struct unDato {
                              Elemento dato;
                              unDato* sig;
   };

   unDato* pri ;
   unDato* ult;
   int longi;
   unDato* iter;
};

// Implementación de las operaciones
template<typename Elemento> void vacia(Cola<Elemento>& c) {
              c.longi = 0;
              c.pri = nullptr;
              c.ult = nullptr;
}
```

En C++

```
template <typename Elemento> void encolar(Cola<Elemento>& c, const Elemento& e) {
        if (c.longi == 0) {
                c.ult = new typename Cola<Elemento>::unDato;
                c.pri = c.ult;
        } else
                c.ult->sig = new typename Cola<Elemento>::unDato;
                c.ult = c.ult->sig;
        }
        c.ult->dato = e;
        c.ult->sig = nullptr;
        c.longi = c.longi +1;
}

// etc etc implementación de las demás operaciones
```