

4.3

The Template Class Chain

2018/9/15 © Ren-Song Tsay, NTHU, Taiwan 16

4.3

Template Chain Class

```

Template < class T > class Chain; // Forward declaration

template < class T >
class ChainNode {
friend class Chain <T>;
private:
    T data;
    ChainNode<T>* link;
};

template <class T>
class Chain {
public:
    // Constructor
    Chain(void) {first = last = NULL;}

    // Chain operations...

private:
    ChainNode<T> *first;
    ChainNode<T> *last;
};
        
```

17

Chain Operations

```

template < class T >
void Chain<T>::InsertBack(const T& e)
{
    if(first) { // Non-empty chain
        last->link = new ChainNode<T>(e);
        last = last->link;
    }
    else // Insert into an empty chain
        first = last = new ChainNode<T>(e);
}

template < class T >
void Chain<T>::Concatenate(Chain<T>& b)
{ // b is concatenated to the end of *this
    if ( first ) { last->link = b.first; last = b.last; }
    else { first = b.first; last = b.last; }
    b.first = b.last = 0;
}
        
```

18

Chain Operations

- Reverse a chain, such that (a_1, \dots, a_n) turns into (a_n, \dots, a_1) .

The diagram illustrates the reversal of a linked list chain. The original chain starts at a pointer labeled 'first' and contains nodes with values 10, 20, ..., 45, 3, 0. A blue arrow labeled 'Reverse' points to the new chain, which starts at 'first' and contains nodes with values 3, 45, ..., 20, 10, 0. The nodes are represented as yellow boxes with arrows pointing to the next node.
