# 4.3 The Template Class Chain



## 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;
};
```



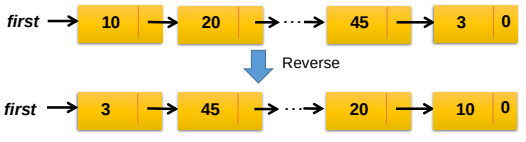
## Chain Operations

```
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;
}
```

## Chain Operations

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










19