Iterative procedure to reverse a singly linked list; assume that list uses a dummy header.
Internal operations in lists, written using internal structure of lists.

LI: revList = head of reversed list, cursor = first element of unprocessed part of list

(dummy) head	reversed part of list ← ← ← ← revList	cursor→ → → →remaining list (unprocessed)

```
revList ← null
cursor ← head.next
while (cursor != null) {
    tmp ← cursor.next
    cursor.next = revList
    revlist = cursor
    cursor = tmp
}
head.next = revList
```

Algorithm in pseudocode to add two numbers stored as lists:

```
static<T> T next(Iterator<T> it) { it.hasNext() ? it.next() : null; }

add(l1, l2)
   it1 = l1.iterator();   it2 = l2.iterator()
   x1 = next(it1);   x2 = next(it2)
   carry = 0;
   out = new list
   while (x1 != null and x2 != null)
            sum = x1 + x2 + carry
            out.add(sum % B)
            carry = sum / B      // Integer division
             x1 = next(it1);   x2 = next(it2)
   while (X1 != null)
            sum = x1 + carry
            out.add(sum % B)
            carry = sum / B      // Integer division
             x1 = next(it1)
   while (X2 != null)
            sum = x2 + carry
            out.add(sum % B)
            carry = sum / B      // Integer division
             x2 = next(it2)
   if (carry > 0)  out.add(carry)
   return out
```