Homework #6 – palindrome & compress

In this homework you are asked to implement two algorithms: `palindrome` and `compress`.

palindrome

`palindrome` accepts a bidirectional iterator range [first, last) as input and returns true if the elements in the range form a palindrome (a sequence that reads the same forward and backward) and false otherwise. The following code shows the declaration for `palindrome` and several sample calls:

```
template <typename BidirectionalIterator>
bool palindrome(BidirectionalIterator first, BidirectionalIterator last);

int data[] = { 1, 2, 3, 4, 3, 2, 1 };
list<int> ls1(data, data + 7);
palindrome(ls1.begin(), ls1.end());  // Returns true

int data[] = { 1, 2, 3, 4, 5, 6, 7 };
list<int> ls2(data, data + 7);
palindrome(ls2.begin(), ls2.end());  // Returns false
```

compress

`compress` accepts a forward iterator range [first, last) and an output iterator. `compress` copies the elements from the iterator range to the output iterator eliminating all consecutive duplicates. The following code shows the declaration for `compress` and several sample calls:

```
template <typename ForwardIterator, typename OutputIterator>
void compress(ForwardIterator first, ForwardIterator last, OutputIterator result);

int data[] = { 1, 1, 2, 2, 1, 1 };
list<int> ls1(data, data + 6), ls2;
compress(ls1.begin(), ls1.end(), back_inserter(ls2)); // ls2 contains 1, 2, 1

int data[] = { 1, 2, 3, 1, 2, 3 };
list<int> ls3(data, data + 6), ls2;
compress(ls3.begin(), ls3.end(), back_inserter(ls4)); // ls4 contains 1, 2, 3, 1, 2, 3
```

1. **(5 points)** Implement `palindrome`. Provide tests showing palindrome working with `begin()`, `end()` iterators into the following containers:
   a. An **empty** `std::list`
   b. A non-empty `std::list` containing an **odd** number of elements that form a **palindrome**
   c. A non-empty `std::list` containing an **even** number of elements that form a **palindrome**
   d. A non-empty `std::list` containing an **odd** number of elements that form a **non-palindrome**
   e. A non-empty `std::list` containing an **even** number of elements that form a **non-palindrome**
2. **(5 points)** Implement `compress`. Provide tests showing compress working with `begin()`, `end()` iterators into the following containers:
   a. An **empty** `std::list`
   b. A non-empty `std::list` containing **no consecutive duplicates**
   c. A non-empty `std::list` containing **consecutive duplicates**

Place all source code and a screen capture of the output produced by your program in a single PDF document. Submit this document.