Prepare your answers to the following questions in a plain text file. Submit your file to the Curator system by the posted deadline for this assignment. No late submissions will be accepted.

You will submit your answers to the Curator System (www.cs.vt.edu/curator) under the heading HW1.

1. [20 pts] For this question, you will design and implement a new member function for the `DListT` template given in the specification for Minor Project 1. The member function must conform to the following interface:

```
template <typename T>
void DListT<T>::alternatingSplit(DListT<T>& L1, DListT<T>& L2) const {
    . . .
}
```

The function will initialize the two `DListT` objects passed to it as parameters so that the first contains the even-numbered elements of the original list, and the second contains the odd-numbered elements of the original list. That is, if the original list contains:

`{4, 17, 3, 12, 5, 21, 23, 41, 7}`

then `L1` and `L2` will be the following lists, respectively:

`{4, 3, 5, 23, 7}` and `{17, 12, 21, 41}`

The order of the elements in the two splits of the original list does matter. Your solution should make no assumptions about the contents of the original list or about the contents of `L1` and `L2` when they are passed in.

2. [10 pts] Write an implementation for a `DListT` client function (i.e., not a member or a friend) that will determine whether the elements in a given `DListT` object are stored in non-strict ascending order.

Your solution must conform to the following interface:

```
template <typename T> bool isAscending(const DListT<T>& List) {
    . . .
}
```

3. [10 pts] Write an implementation for a `DListT` client function (i.e., not a member or a friend) that will merge two ascending-ordered `DListT` objects into a single ascending-order `DListT` object, without modifying the original objects. For example, if `L1` and `L2` are the following lists, respectively:

`{4, 5, 11, 17}` and `{2, 7, 11, 41}`

then the returned list should be:

`{2, 4, 5, 7, 11, 11, 17, 41}`

If either of the given lists is not in ascending order, an empty list should be returned. Your solution must conform to the following interface:

```
template <typename T>
DListT<T> mergeAscending(const DListT<T>& L1, const DListT<T>& L2) {
    . . .
}
```