OOP (1010), Summer 2015

Lab 7

The goal of this lab is an experiment of the inheritance from STL container. First, design a class named `listX` that inherits from `std::list` and has a public member function named `sum()`, which can sum all data items of `listX`. Note that you only need to consider the standard numeric data types but you must avoid the overflow error in `listX::sum()`. Please design two exception classes: `OverLargest` and `OverLowest`. `listX::sum()` will throw an object of OverLargest if a maximum value overflow occurred; `listX::sum()` will throw an object of OverLowest if a minimum value overflow occurred. Both of them can provide a message string.

There is a global function named `SumVector` in main.cpp. Please modify `SumVector` to catch the exceptions thrown by `listX::sum()`and show the messages of exceptions.

Notice that you have to verify the overflow correctly. For example, if `sizeof(short)` is 2 byte, and an object of `listX<short>` contains {32760, 8, 20, -32760, 100}, no any overflow occurred and the summation result is 128. If that object contains {32760, 8, 20}, `listX::sum()` throws an exception of `OverLargest`. If it contains {-14, 10, -32760, -8}, `listX::sum()` throws an exception of `OverLowest`.

[Hint]

1. How to check the data type is signed or unsigned numeric type, integer or floating-point number? For signed type check, you can use
   `std::numeric_limits<T>::is_signed`.
   For floating-point number check, you can use
   `std::numeric_limits<T>::is_iec559`
   or template specialization.
2. Sort all data items if the data type is signed type.
3. Using two iterators to sum all data items: One from left goes to right, the other one from right goes to left.

Input / Output

The main.cpp should generate the following results:

```
--- unsigned long long test ---
vll1 = 0, 7, 18446744073709551612, 1, 3
vll2 = 6, 6, 6, 6, 6
vll3 = 0, 7, 18446744073709551612, 1, 3
```

```
largest overflow
ACC:7
vll3 = 0, 7, 18446744073709551612, 0, 3
largest overflow
--- integer test ---
vi1 =
vi2 = 6, 6, 6, 6, 6
vi3 = 0, -10, 0, -2147483645, 6, 7, 2147483644
vi4 = 0, -10, 0, -2147483645, 6, 7, 2147483644
2
ACC:2
vi4 = 0, 0, 0, -2147483645, 6, 7, 2147483644
12
vi4 = 0, -10, 0, -2147483645, 0, 0, 0
lowest overflow
vi4 = -2147483648, -2147483648, -2147483648, -2147483645,
2147483647, 2147483647, 2147483647
-2147483648
ACC:-2147483648
vi4 = -2147483648, 100, -1073741829, -1073741829, 2147483647,
2147483647, 2147483647
largest overflow
ACC:-2147483561
--- float test ---
vf1 = 1.73, 3.40282e+38, 0, -1, -3.40282e+38
vf2 = 1.73, 3.40282e+38, 0, -1, -3.40282e+38
vf3 = -3.40282e+38, -3.40282e+38, 0, 0, 3.40282e+38
0.73
ACC:0
-3.40282e+38
ACC:-inf
vf2 = 1.73, 0, 0, -1, -3.40282e+38
lowest overflow
vf2 = 1.73, 3.40282e+38, 0, -1, 0
largest overflow
--- double test ---
vd1 = 1.56, 1.79769e+308, 0, -1, -1.79769e+308
vd2 = 1.56, 1.79769e+308, 0, -1, -1.79769e+308
0.56
ACC:inf
```

```
vd2 = 1.56, 0, 0, -1, -1.79769e+308
lowest overflow
vd2 = 1.56, 1.79769e+308, 0, -1, 0
largest overflow
```