1 Part: Set container with approximate find 50%

In this exercise you will implement a simple set container, storing elements of some type `T`. For the elements of type `T`, we assume that copy constructors and assignment operators have been implemented for it, as well as `operator<`, `operator==`, `operator<<` and `operator>>`. You are allowed to use any STL containers for storing information.

In addition to some standard container operations, the set container you have to implement has several interesting functions. These are as follows.

- The function `findClosest(const T& val, unsigned int K)` searches for a value in the container, and if it finds it, returns a vector of size 1 containing this element. Otherwise, it returns a vector $(l_1, ..., l_i, g_1, ...g_j)$, where $l_1, ..., l_i$ are the largest K elements of the set that are smaller the sought value, and $g_1, ...g_j$ are the smallest K elements of the set that are greater than element. If the container contains fewer than K elements smaller than val, then $l_1, ..., l_i$ are all the elements of the set that are smaller than val. If the container contains fewer then K elements greater than element, then $g_1, ..., g_j$ are all the elements of the set that are greater than element. Furthermore, the elements of the returned vector should be ordered in ascending order according to $<$, that is, we should have $l_1 < ... < l_i < g_1 < ... < g_j$.

  For example, suppose we have a container `s` consisting of the integers

  `1,2,3,5,6,7,9,10.`

  Then, `s.findClosest(5,4)` should return a vector containing only the value `5`.

  The call `s.findClosest(8,3)`, on the other hand, should return the vector consisting of the numbers `5,6,7,9,10`, in this order.

  The call `s.findClosest(12,3)`, should return `7,9,10`.

- The function `getRange(const T& start_value,const T& end_value)` returns the elements of the set container that are greater or equal to `start_value` and smaller or equal to `end_value`. The elements of the returned vector should be in ascending order according to `<`. If `start_value > end_value`, then the returned vector should be empty. In any case, `start_value` and `end_value` do not have to exist in the container. In the example of the above container,

  `s.getRange(4,9)` should return a vector consisting of `5,6,7,9`,

  `s.getRange(4,8)` should return a vector consisting of `5,6,7`, and

  `s.getRange(7,2)` and `s.getRange(const 11,13)` return an empty vector.

A detailed description of the methods of the class is given in `Set.h`.