Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Programming Techniques for Scientific Simulations Exercise 7

HS 15
Prof. M. Troyer

Problem 7.1 Benchmarking standard containers

We want to benchmark the time required to randomly insert and delete in the three types of containers provided by the standard library: `std::vector`, `std::list`, `std::set`.
In order to achieve this, you should:

1. Create an array of size n and assign its entries with a strictly monotone function of their index, i.e. `array[i] = 2*i`.
2. Copy the content of the array into a `std::vector`, `std::list` and `std::set`.
3. For each container record the time to:
    - Insert a new element j from range 0 to $2n$ to the container such that the order is preserved.
    - Undo the previous operation by erasing the inserted element.

Perform benchmarks for many system sizes n, you should be able to see the cross over of performance of the examined containers.
Hints:

- Make use of `std::advance` to have a generic random access iterator.
- Special treatment is required for the `std::set`. Note that `std::set` does not store multiple copies of same entry. Take care that the size of the container does not shrink during the benchmark.
- To record a proper time you should measure the time between k repetitions of step 3., where $k \approx 1'000'000$.

Problem 7.2 Iterators

As we will see iterators provide the interface between containers and algorithms and thus play a key role in most scientific programs. In this exercise we will provide an `Iterator` for the `Array<T>` class discussed during the lecture (check `examples/week4/array.hpp` for the code). Your goals for today will be:

i) Implement a *forward iterator* class.

ii) Make it a checked iterator.

iii) (Bonus) Upgrade your forward iterator to a *bidirectional iterator*.

Check `exercises/week7/iterator-skeleton.hpp` for a skeleton of the code and the precise requirements on forward and bidirectional iterators. Make sure to add `begin()` and `end()` member functions to the array class so that you may easily obtain iterators for your array. Check the online repository for a simple program (`exercises/week7/main.cpp`) with which you may test some of the fuctionality of your iterator.