Goal

The goal of this exercise is to highlight the advantages of generic programming and how it interfaces with the containers proposed by the STL.

In the videos you have seen that all stl containers can be visited using iterators. The capabilities of the iterator depend on the underlying container, but all of them can (at least) be dereferenced, incremented and (equality) compared.

As you have realized by now, these are exactly the operations needed to implement the selection sort algorithm, which is exactly what we are going to do!

You must implement the generic version of selection sort (my_selection_sort in the given file) which works for all containers of the STL (given that they hold weakly ordered objects (objects for which *operator<* exists) using the iterator interface.

Restrictions

You may not include *algorithm* or use any other third party lib. Your code must have one (and only one) generic implementation of the selection sort algorithm. (This will be checked)

Challenge

Implementations faster than our (suboptimal but not bad) reference implementation get extra points.

Bonus

Another bonus is given to an implementation if it is at least 10% faster than any of the other implementations (including our reference implem)