```
/* The following code example is taken from the book
 * "The C++ Standard Library - A Tutorial and Reference, 2nd Edition"
 * by Nicolai M. Josuttis, Addison-Wesley, 2012
 *
 * (C) Copyright Nicolai M. Josuttis 2012.
 * Permission to copy, use, modify, sell and distribute this software
 * is granted provided this copyright notice appears in all copies.
 * This software is provided "as is" without express or implied
 * warranty, and with no claim as to its suitability for any purpose.
 */
#include <iterator>

// class template for insert iterator for associative and unordered containers
template <typename Container>
class asso_insert_iterator
 : public std::iterator <std::output_iterator_tag,
                         typename Container::value_type>
{
  protected:
    Container& container;    // container in which elements are inserted

  public:
    // constructor
    explicit asso_insert_iterator (Container& c) : container(c) {
    }

    // assignment operator
    // - inserts a value into the container
    asso_insert_iterator<Container>&
    operator= (const typename Container::value_type& value) {
        container.insert(value);
        return *this;
    }

    // dereferencing is a no-op that returns the iterator itself
    asso_insert_iterator<Container>& operator* () {
        return *this;
    }

    // increment operation is a no-op that returns the iterator itself
    asso_insert_iterator<Container>& operator++ () {
        return *this;
    }
    asso_insert_iterator<Container>& operator++ (int) {
        return *this;
    }
};

// convenience function to create the inserter
template <typename Container>
inline asso_insert_iterator<Container> asso_inserter (Container& c)
{
    return asso_insert_iterator<Container>(c);
}
```