```
/* The following code example is taken from the book
 * "The C++ Standard Library - A Tutorial and Reference"
 * by Nicolai M. Josuttis, Addison-Wesley, 1999
 *
 * (C) Copyright Nicolai M. Josuttis 1999.
 * 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 <functional>

/* class for the compose_f_gx_hy adapter
 */
template <class OP1, class OP2, class OP3>
class compose_f_gx_hy_t
 : public std::binary_function<typename OP2::argument_type,
                               typename OP3::argument_type,
                               typename OP1::result_type>
{
  private:
    OP1 op1;    // process: op1(op2(x),op3(y))
    OP2 op2;
    OP3 op3;
  public:
    // constructor
    compose_f_gx_hy_t (const OP1& o1, const OP2& o2, const OP3& o3)
     : op1(o1), op2(o2), op3(o3) {
    }

    // function call
    typename OP1::result_type
    operator()(const typename OP2::argument_type& x,
               const typename OP3::argument_type& y) const {
        return op1(op2(x),op3(y));
    }
};

/* convenience function for the compose_f_gx_hy adapter
 */
template <class OP1, class OP2, class OP3>
inline compose_f_gx_hy_t<OP1,OP2,OP3>
compose_f_gx_hy (const OP1& o1, const OP2& o2, const OP3& o3) {
    return compose_f_gx_hy_t<OP1,OP2,OP3>(o1,o2,o3);
}
```