Exercise 12:

Submit your solutions (source code) to the questions below by email to your instructor and TA(s) by Monday, January 7th (16:30).

Question 1: Containers and iterators (30 points).

This exercise will help you to assess your understanding of the STL containers and iterators.

Your goal is to write a template function my_find() taking as input two iterators defining a range [iterator1; iterator2) and an element 'e' and returning an iterator to the first element equal to e in the container (if no elements are found, it returns iterator2).
Please create a file "my_find.h" and write your template function in this file.

To test your code, please create a file "test_my_find.cpp" and type into it the following code:

```
#include <list>
#include <iostream>
#include <cassert>

#include "my_find.h"

int main(void) {
 std::list<int> l;
 l.push_back(1);
```

```
 l.push_back(2);
 l.push_back(5);
 l.push_back(1);
 l.push_back(2);

 std::list<int>::iterator it;
 it = my_find(l.begin(), l.end(), 2);
 assert(it != l.end()); // found

 it = my_find(l.begin(), l.end(), 11);
 assert(it == l.end()); // not found

 std::cout << "Tests passed" << std::endl;

 return 0;
}
```

Question 2: Iterator (35 points).

Write a function reverse() that takes as argument a vector of chars (vector<char>) and returns as output a new vector of chars with the same values as the original vector but in reverse order.

```
// reverse.cpp
#include <vector>
#include <iostream>
#include <cassert>

using namespace std;
```

```
// COMPLETE: write an implementation of reverse()

int main(void) {
  vector<char> input;
  input.push_back('a');
  input.push_back('b');
  input.push_back('c');

  vector<char> reversed;
  reversed.push_back('c');
  reversed.push_back('b');
  reversed.push_back('a');

  assert(reverse(input) == reversed);
  cout << "Test passed" << endl;

  return 0;
}
```

Question 3: Iterator adapters (35 points).

Write a program which reads integer numbers from std::cin using a std::istream_iterator. The program should print all odd numbers, separated by spaces, to an output file. Use std::ostream_iterators for printing to the output file. Please write the corresponding code in a file named "even.cpp".