Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Lecturer: Prof. B. Gärtner

Informatik für Mathematiker und Physiker HS14

Exercise Sheet 9

Submission deadline: 15:15 - Tuesday 18th November, 2014
Course URL: http://lec.inf.ethz.ch/ifmp/2014/

Short Summary

	Array	Vector
Array/Vector Initialization	`int my_arr[] = {1,2,3};`	`std::vector<int> my_vec = {1,2,3}; // 1 2 3` `std::vector<int> my_vec (3, 1); // 1 1 1`
Iterator to Begin	`int* ptr = my_arr;`	`std::vector<int>::iterator itr = my_vec.begin();`
const	`const int* cptr = my_arr;`	`std::vector<int>::const_iterator citr = my_vec.begin();`
Iterator to Past-the-End	`int* ptr = my_arr + 3;`	`std::vector<int>::iterator itr = my_vec.end();`
const	`const int* cptr = my_arr + 3;`	`std::vector<int>::const_iterator citr = my_vec.end();`
to Next Elt	`++ptr`	`++itr`
to Previous Elt	`--ptr`	`--itr`
Distance	`ptr1 - ptr2`	`itr1 - itr2`
Comparisons	`ptr1 < ptr2`	`itr1 < itr2`
	`ptr1 != ptr2`	`itr1 != itr2`
	...	...

Assignment 1 (4 points)

a) (Skript-Aufgabe 115 a) What does the following program output, and why?

```
#include<iostream>
int main() {
  int a[] = {5, 6, 2, 3, 1, 4, 0};
  int* p = a;
  do {
    std::cout << *p << " ";
    p = a + *p;
  } while (p != a);

  return 0;
}
```

b) On the course website you find a template `array_iteration_template.cpp`. Extend it with a code snippet which outputs every second element (starting from 0) in the first given array using pointers.

c) Extend `array_iteration_template.cpp` with a code snippet which outputs the second given array in reverse order using pointers.

Assignment 2 (4 points)

`std::vectors` are very powerful: You can use `push_back` to append a new element after the last element of the vector. For example:

```
std::vector<int> my_vec (5,0);   // my_vec is 0 0 0 0 0
my_vec.push_back(8);             // my_vec is 0 0 0 0 0 8
```

This functionality can help you with the tasks below. Also, there is a template `ds_examination_template.cpp` on the lecture website. You can use it if you want. For this exercise description you shall assume:
`typedef std::vector<int>::const_iterator cIt;`

a) Write the following function. For this subtask you are *not* allowed to use any Standard Library functions.

```
// PRE: [begin, end) is a valid range with at least one element
// POST: the minimum is written to min, and the maximum is
//       written to max
void min_max (cIt begin, cIt end, int& min, int& max);
```

b) (Skript-Aufgabe 111) Write the following function. For this subtask you are *not* allowed to use any Standard Library functions.

```
// PRE:  [begin, end) is a valid range and describes a sequence
//       of elements that are sorted in nondecreasing order
// POST: the return value is true if and only if no element
//       occurs twice in the sequence
bool all_unique (cIt begin, cIt end);
```

c) Use `push_back` to read a dataset of unknown length into a vector of type `int`. Then apply the functions from parts a) and b) to your vector. And finally output these values. For this task you may use the function `std::sort` to sort your vector.

Example for `std::sort` (need to `#include <algorithm>`):

```
std::vector<int> my_vec = {5,2,3,1,4};
std::sort(my_vec.begin(), my_vec.end());  // my_vec is 1 2 3 4 5
```

d) (Skript-Aufgabe 118) Improve your function from subtask a) such that it needs at most $\frac{3}{2}n$ comparisons in total to find both, the minimum and the maximum.

Have a look at the programming project!