

Part I: Practice and Theory

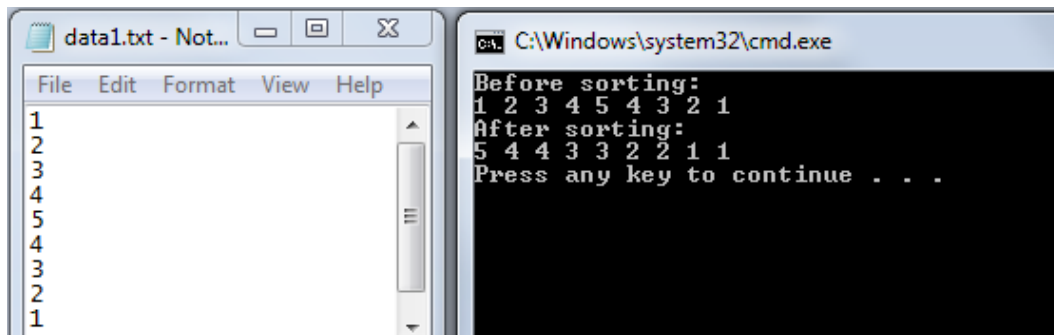
The following problems are for practice only and will **not be collected**.

Review problems: All. **Practice Problems:** P11.2, P11.5, P11.9, P11.10.

Part II: Programming. The following problems will be **collected** and two of them graded. Each graded problem will be worth 25 points. Read instructions carefully!

(1) Based on **Problem P11.1**

- Modify the selection sort algorithm to sort a vector of integers in descending order.
- Write a program that reads the list of integers from a file `data1.txt` into a vector and then uses the above algorithm to sort it. Then display the sorted vector.
- You can assume that `data1.txt` contains only integers (no floating point numbers or strings that are not numbers) and that the file has at least one number.
- [Submit the solution as hmw_5_1.cpp](#).
- Sample input-output:



The screenshot shows two windows side-by-side. The left window is a Notepad++ editor titled 'data1.txt - Notepad++'. It contains a list of integers: 1, 2, 3, 4, 5, 4, 3, 2, 1. The right window is a Windows command prompt titled 'C:\Windows\system32\cmd.exe'. It displays the following text: 'Before sorting:', '1 2 3 4 5 4 3 2 1', 'After sorting:', '5 4 4 3 3 2 2 1 1', and 'Press any key to continue . . .'. The command prompt has a black background and white text.

(2) Based on **Problem P11.4**

- Modify the merge sort algorithm to sort a vector of Employees by salary. The class Employee in this problem is assumed to be

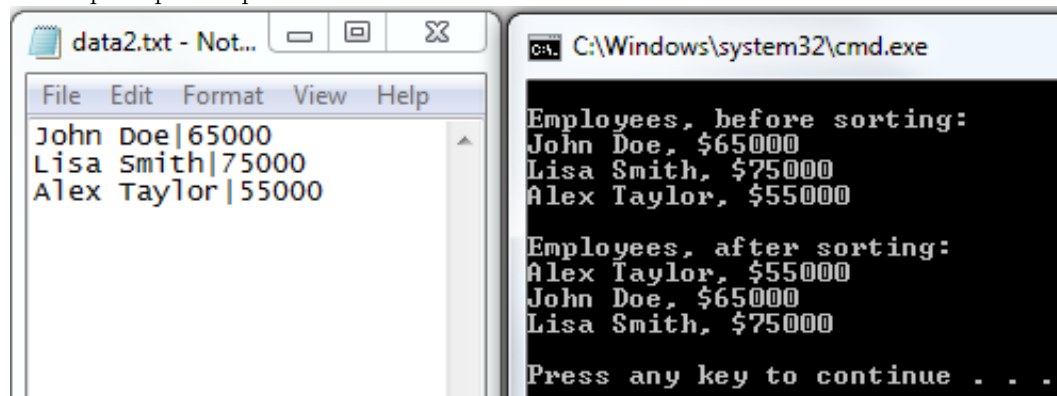
```
class Employee {  
public:  
    Employee(string e_name, double e_salary)  
        : name(e_name), salary(e_salary) { }  
    string get_name() const { return name; }  
    double get_salary() const { return salary; }  
protected:  
    string name;  
    double salary;  
};
```

- Write a program that reads the list of people's records from a file data2.txt, creates a corresponding vector<Employee> and displays the name and salary for each employee. After that the program should sort vector<Employee> and display the name and the salary of employees in the sorted vector (for details see IO sample below).

- You can assume that data for each employee is stored in data2.txt as name|salary (one data record per line) and that the file has at least one line.

- [Submit the solution as hmw_5_2.cpp.](#)

- Sample input-output:



(3) Based on **Problem P11.6**

- Implement a function

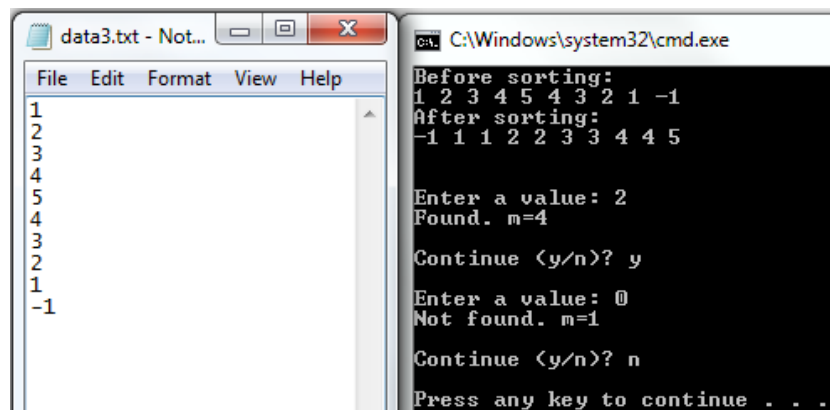
```
bool binary_search(vector<int> & v, int value, int & m),
```

that uses binary search to find out whether there is an integer value in the vector `v` of integers. If `vector<int> & v` is not sorted, then the function must return `false` and set `m=-1`. If `v` is sorted and a match is found, the function must return `true`, otherwise `false`. In the above function `m` is a reference parameter, which must be set to the location of the match if the search was successful. If `value` were not found, set `m` to the index of the next larger value instead, or to `v.size()` if `value` is larger than all the elements of the vector.

- Write a program that reads the list of integers from a file `data3.txt` and displays them on the screen. Then the program must sort the numbers using `sort` function from C++ library and then display the sorted vector.
- Finally, implement a loop in which the user is asked to enter a value, which is then searched in the sorted array using the above `binary_search` function. If `value` is found display “Found. m=” followed by the value of `m`. Otherwise display “Not found. m=” followed by the value of `m`.
- You can assume that `data3.txt` contains only integers (no floating point numbers or strings that are not numbers) and that the file has at least one number.

- [Submit the solution as hmw_5_3.cpp.](#)

- Sample input-output:



Note: The value of `m` in the output sample and the one you obtain may differ due to the implementation of `binary_search`.

(3) Based on **Problem P11.6**

- Implement a function

```
vector< size_t > remove_duplicates( vector<int> & v)
```

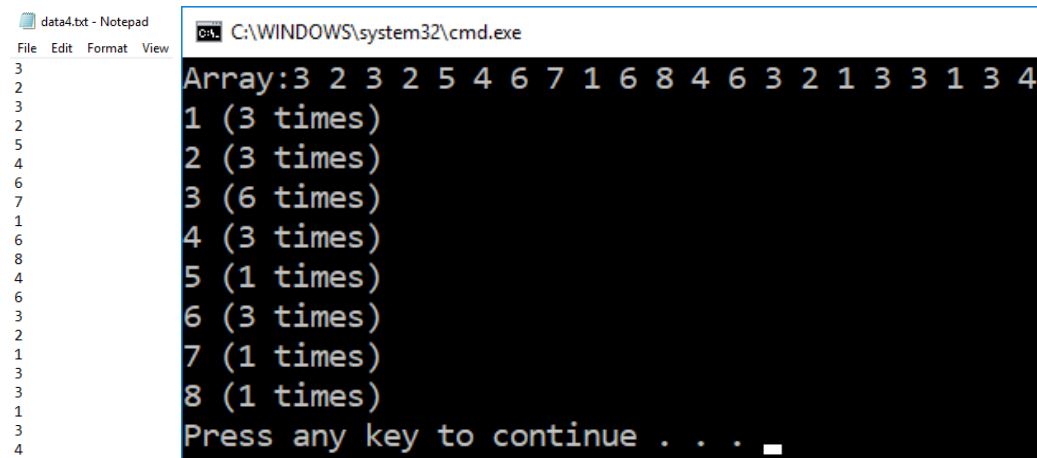
that removes the duplicates from `vector<int> & v` and returns the `vector<size_t>` containing the number of times each distinct value appeared in the original `v`. The algorithm should have complexity $O(n \log n)$. **Hint:** Sort `v` using built-in sort function. **Note:** The elements in `v` may have a different order after removal of duplicates.

- Write a program that reads the list of integers from the file `data4.txt` into a vector. Display the numbers on the screen. Then call the function `remove_duplicates` and display the content of the modified vector indicating how many times each number appeared in the original vector. The order of the displayed numbers does not matter. See the sample of the input-output.

- You can assume that `data4.txt` contains only integers (no floating point numbers or strings that are not numbers) and that the file has at least one number.

- [Submit the solution as hmw_5_4.cpp](#).

- Sample input-output:



The screenshot shows two windows. On the left is a Notepad window titled 'data4.txt - Notepad' with the following content:

```
3
2
3
2
5
4
6
7
1
6
8
4
6
3
2
1
3
3
1
3
4
```

On the right is a Windows command prompt window titled 'C:\WINDOWS\system32\cmd.exe' showing the output of a program:

```
Array:3 2 3 2 5 4 6 7 1 6 8 4 6 3 2 1 3 3 1 3 4
1 (3 times)
2 (3 times)
3 (6 times)
4 (3 times)
5 (1 times)
6 (3 times)
7 (1 times)
8 (1 times)
Press any key to continue . . .
```