Lab 9 – STL

Exercise 1. STL algorithms

In the file **ex_9_1_stdalgorithms.cpp** you will find comments with tasks descriptions. They should be implemented without any loop, only by calling appropriate STL algorithm (one line is needed in most cases).

Exercise 2. Word count – Standard associative containers

Implement program **words** that:

- counts number of distinct words in file redirected to standard input ,
- prints the 20 most popular words with the number of occurrences.

The result of calling

```
./words < hamletEN.txt
```

should be similar to hamletOUT.txt

Hints:

- Use toLowerAlpha function from previous exercise to standardize words (lower case, alphanumeric only).
- Use standard associative container `map<string,int>` to eliminate duplicates and to count occurrences.
- Compare the speed of `map` and `unordered_map`.
- Use `multimap<int, string>` to "reverse" map and find most popular words.

Exercise 3. Trie

When we split sentences into words we can group sentences that have the same beginning. To store sentences we can use trie data structure (called also prefix tree). We can easily obtain possible continuations of given sentence.

Start with file ex_9_3_Trie.cpp implement

- Trie data structure using std::map,
- method add that adds sentence to trie,
- method printPossibleEndings that for given beginning of sentence prints all possible endings.

Part of a graph of Tie structure for file sample.txt