- Exercise

In this lesson, we'll execute and analyze the given code.

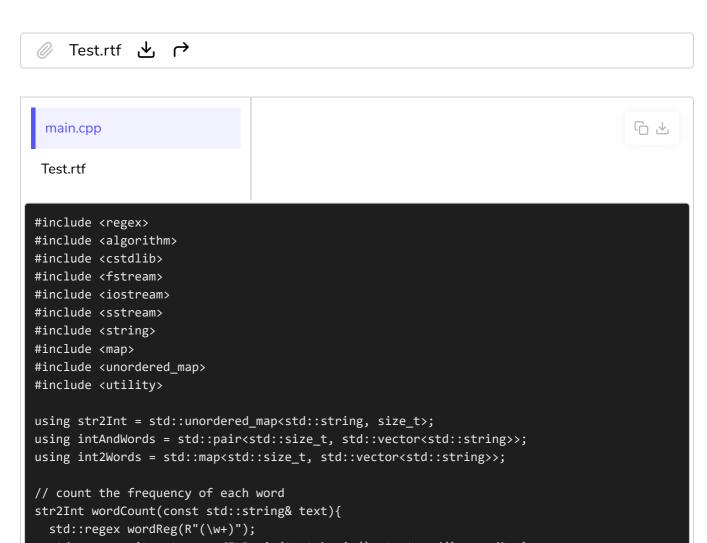
```
we'll cover the following ^

Try it out
```

Try it out

The program how frequently a word is used in a text. This is a typical use-case for an ordered associative container.

- Execute the program
- Analyze the source code



```
Julian Car
                       wordicbegin(cext.begin(), text.end(), wordkeg),
  const std::sregex_iterator wordItEnd;
  str2Int allWords;
  for (; wordItBegin != wordItEnd;++wordItBegin){
   ++allWords[wordItBegin->str()];
  return allWords;
// get to all frequencies the appropriate words
int2Words frequencyOfWords(str2Int& wordCount){
  int2Words freq2Words;
  for ( auto wordIt: wordCount ){
    auto freqWord= wordIt.second;
    if ( freq2Words.find(freqWord) == freq2Words.end() ){
      freq2Words.insert( intAndWords(freqWord, {wordIt.first} ));
   else {
      freq2Words[freqWord].push_back(wordIt.first);
  return freq2Words;
int main(int argc, char* argv[]){
  std::cout << std::endl;</pre>
  // get the filename
  std::string myFile = "Test.rtf";
  // open the file
  std::ifstream file(myFile, std::ios::in);
  if (!file){
   std::cerr << "Can't open file "+ myFile + "!" << std::endl;</pre>
    exit(EXIT_FAILURE);
  // read the file
  std::stringstream buffer;
  buffer << file.rdbuf();</pre>
  std::string text(buffer.str());
  // get the frequency of each word
  auto allWords= wordCount(text);
  std::cout << "The first 20 (key, value)-pairs: " << std::endl;</pre>
  auto end= allWords.begin();
  std::advance(end, 20);
  for (auto pair= allWords.begin(); pair != end; ++pair){
    std::cout << "(" << pair->first << ": " << pair->second << ")";</pre>
  std::cout << "\n\n";</pre>
  std::cout << "allWords[Web]: " << allWords["Web"] << std::endl;</pre>
  std::cout << "allWords[The]: " << allWords["The"] << "\n\n";</pre>
  std::cout << "Number of unique words: ";</pre>
  std::cout << allWords.size() << "\n\n";</pre>
  size_t sumWords=0;
  for (auto wordIt: allWords) sumWords+= wordIt.second;
  std::cout << "Total number of words: " << sumWords <<"\n\n":</pre>
```

```
auto allFreq= frequencyOfWords(allWords);
std::cout << "Number of different frequencies: " << allFreq.size() << "\n\n";</pre>
std::cout << "All frequencies: ";</pre>
for (auto freqIt: allFreq) std::cout << freqIt.first << " ";</pre>
std::cout << "\n\n";</pre>
std::cout << "The most frequently occurring word(s): " << std::endl;</pre>
auto atTheEnd= allFreq.rbegin();
std::cout << atTheEnd->first << " :";</pre>
for (auto word: atTheEnd->second) std::cout << word << " ";</pre>
std::cout << "\n\n";</pre>
std::cout << "All word which appears more then 1000 times:" << std::endl;</pre>
auto biggerIt= std::find_if(allFreq.begin(), allFreq.end(), [](intAndWords iAndW){return i
if (biggerIt == allFreq.end()){
  std::cerr << "No word appears more then 1000 times !" << std::endl;</pre>
  exit(EXIT FAILURE);
else{
  for (auto allFreqIt= biggerIt; allFreqIt != allFreq.end(); ++allFreqIt){
     std::cout << allFreqIt->first << " :";</pre>
     for (auto word: allFreqIt->second) std::cout << word << " ";</pre>
     std::cout << std::endl;</pre>
  }
std::cout << std::endl;</pre>
                                                                                 \triangleright
```

In this chapter, we have studied ordered associative containers. In the next chapter, we'll study unordered associative containers in detail.