Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Теоретическая информатика и компьютерные технологии»

Лабораторная работа № 12

по курсу «Языки и методы программирования»

«Обработка текстовых файлов»

Студент группы ИУ9-22Б Федуков А. А.

Преподаватель Посевин Д. П.

22 мая 2024 г.

Цель работы

Целью лабораторной работы является приобретение навыка разработки на языке C++ программ, осуществляющих анализ и преобразование текстовых файлов, записанных в различных форматах.

Задание

Варианты заданий для программ, которые нужно разработать в ходе выполнения данной лабораторной работы, приведены в таблицах 1–7. Каждая программа должна принимать через аргумент командной строки путь к каталогу, в котором располагаются подлежащие обработке файлы.

Задание 1

Найти все файлы с расширением «md» в указанном каталоге, и в тексте каждого файла пронумеровать заголовки целыми числами. Изменённые файлы следует сохранить в текущем каталоге. Заголовки в формате MarkDown обозначаются с помощью «подчёркивания» последовательностью знаков «=» или «-». Например:

Введение ========

Работоспособность программы нужно проверить на наборе предварительно составленных MD-файлов.

Реализация

Я создал тестовые файлы 1.md и 2.md в подкаталоге FullOfMD.
И написал программу, реализующую заявленные функции в файле main.cpp

Входные файлы

Листинг 1: Файл 1.md

```
FIRST FILE
==
```

```
MD
END
```

Листинг 2: Файл 2.md

```
AGAIN AND AGAIN
==

text text
 - kicdijdci
 - kmdskmsdkm

====

==c

BIG
===
TExt
====
```

Листинг 3: Файл(отвлекающий) 1.txt

```
text
====
```

Код

Листинг 4: Файл main.cpp

```
#include <string>
#include <iostream>
#include <filesystem>
#include <fstream>
#include <vector>

using namespace std;


bool LineSuitableForHeader(string line)
{
    return line != "" && line.find("- ") == string::npos;
}

bool LineAfterHeader(string line)
{
```

```
17      char mode = '!';
18      bool out = true;
19      for (auto &&i : line)
20      {
21          switch (i)
22          {
23          case '=':
24              if (mode == '!' || mode == '=')
25                  mode = '=';
26              else
27                  out = false;
28              break;
29          case '-':
30              if (mode == '!' || mode == '-')
31                  mode = '-';
32              else
33                  out = false;
34              break;
35          default:
36              out = false;
37              break;
38          }
39          if (!out)
40              break;
41      }
42
43      return out;
44  }
45
46  void handleFile(filesystem::path path)
47  {
48      cout << "Found file: " << filesystem::absolute(path) << endl;
49
50      string line;
51      vector<string> lines;
52      fstream fileIN, fileOUT;
53      fileIN.open(path, ios::in);
54      fileOUT.open(path.filename(), ios::out);
55
56      if (fileIN.is_open() && fileOUT.is_open())
57      {
58          size_t i = 0;
59          int headerCount = 0;
60          string lastLine = "";
61          while (getline(fileIN, line))
62          {
```

```
63
64              lines.push_back(line);
65              if (line != "" && i != 0 && LineAfterHeader(lines[i]) &&
    LineSuitableForHeader(lines[i - 1]))
66                  fileOUT << ++headerCount << ". ";
67
68              i += 1;
69              if (lines.size() != 1)
70                  fileOUT << lastLine << endl;
71              lastLine = line;
72          }
73      }
74      fileIN.close();
75      fileOUT.close();
76  }
77
78  int main(int argc, char const *argv[])
79  {
80      if (argc == 1)
81      {
82          filesystem::path path = "./FullOfMD/";
83
84          for (const auto &entry : std::filesystem::directory_iterator(
    path))
85          {
86              filesystem::path file = entry;
87              if (file.extension() == ".md")
88                  handleFile(file);
89          }
90      }
91      else
92      {
93          for (size_t i = 1; i < argc; i++)
94          {
95              if (!std::filesystem::is_directory(argv[i]))
96              {
97                  cerr << "Это не папка!" << endl;
98                  exit(0);
99              }
100
101             for (const auto &entry : std::filesystem::directory_iterator
    (argv[i]))
102             {
103                 filesystem::path file = entry;
104                 if (file.extension() == ".md")
105                     handleFile(file);
```

```
106                 }
107             }
108         }
109 }
```

Вывод программы

Программа обнаружила файлы .md и создала их отредактированные по заданию копии 1.md и 2.md

Листинг 5: Вывод программы

```
1 Found file: "/home/chinalap/Документы/YIMP_labs/lab12/files/code/task1
      /./FullOfMD/1.md"
2 Found file: "/home/chinalap/Документы/YIMP_labs/lab12/files/code/task1
      /./FullOfMD/2.md"
```

Выходные файлы

Листинг 6: Файл 1.md

```
1 1. FIRST FILE
2 ===
3 MD
4 END
```

Листинг 7: Файл 2.md

```
1  1. AGAIN AND AGAIN
2  ==
3
4  text text
5   - kicdijdci
6   - kmdskmsdkm
7
8  ====
9
10 ==c
11
12  2. BIG
13 ===
14  3. TExt
15 ====
```

Задание 2

Найти все файлы с расширением «html» в указанном каталоге, для каждого файла построить три множества заголовков типа h1, h2 и h3 и сохранить объединённые множества заголовков разного типа из всех файлов в файлах h1.txt, h2.txt и h3.txt в текущем каталоге. Каждый заголовок в сформированном файле должен располагаться в отдельной строке, для чего может потребоваться удаление из заголовка символов перевода строки. Кроме того, заголовки должны быть отсортированы лексикографически. Заголовки в HTML-файле задаются тегами «h1», «h2» и «h3». Например:

<h1>Введение</h1>

Работоспособность программы нужно проверить на наборе HTML-файлов, загруженных из интернета.

Реализация

Я создал тестовые файлы my.html и example.com.html в подкаталоге FullOfHTML.
И написал программу, реализующую заявленные функции в файле main.cpp

Входные файлы

Листинг 8: Файл my.html

```
<!DOCTYPE html>
<html>
    <body>
        dlcdncd
        <h1>zzzH1h1h1h1h1h1

            h1h1</h1>
        <h2>H@H@H@H@HH222222</h2>
        <h3>iufosokfdh3</h3>
        <H1>aaaH1h1h1h1h1h1h1h1</h1>
    </body>
</html>
```

Листинг 9: Файл example.com.html

```
<html><head>
    <title>Example Domain</title>

    <meta charset="utf-8">
    <meta http-equiv="Content-type" content="text/html; charset=utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <style type="text/css">
    body {
        background-color: #f0f0f2;
        margin: 0;
        padding: 0;
        font-family: -apple-system, system-ui, BlinkMacSystemFont, "
    Segoe UI", "Open Sans", "Helvetica Neue", Helvetica, Arial, sans-
    serif;

    }
    div {
        width: 600px;
        margin: 5em auto;
        padding: 2em;
        background-color: #fdfdff;
        border-radius: 0.5em;
        box-shadow: 2px 3px 7px 2px rgba(0,0,0,0.02);
    }
    a:link, a:visited {
        color: #38488f;
        text-decoration: none;
    }
    @media (max-width: 700px) {
        div {
            margin: 0 auto;
            width: auto;
        }
    }
    </style>
</head>

<body>
<div>
    <h1>Example Domain</h1>
    <p>This domain is for use in illustrative examples in documents. You
     may use this
    domain in literature without prior coordination or asking for
    permission.</p>
    <p><a href="https://www.iana.org/domains/example">More information
    ...</a></p>
```

```
42 </div>
43
44
45 </body></html>
```

Код

Листинг 10: Файл main.cpp

```
1  #include <string>
2  #include <iostream>
3  #include <filesystem>
4  #include <fstream>
5  #include <vector>
6  #include <cctype>
7  #include <algorithm>
8
9  using namespace std;
10 vector<string> h1;
11 vector<string> h2;
12 vector<string> h3;
13
14 bool stringCmp(string s1, string s2){
15     if (s1.size() != s2.size())
16         return s1.size() < s2.size();
17     else
18     {
19         for (size_t i = 0; i < s1.size(); i++)
20         {
21             if (s1[i] < s2[i])
22                 return true;
23             if (s1[i] > s2[i])
24                 return false;
25         }
26         return true;
27     }
28
29 }
30 void writeHeaders()
31 {
32     sort(h1.begin(), h1.end(), stringCmp);
33     sort(h2.begin(), h2.end(), stringCmp);
34     sort(h3.begin(), h3.end(), stringCmp);
35
36
```

```
37      fstream fileH1, fileH2, fileH3;
38      fileH1.open("h1.txt", ios::out);
39      fileH2.open("h2.txt", ios::out);
40      fileH3.open("h3.txt", ios::out);
41      if (fileH1.is_open() && fileH2.is_open() && fileH3.is_open())
42      {
43          for (auto &&i : h1)
44              fileH1 << i << endl;
45          for (auto &&i : h2)
46              fileH2 << i << endl;
47          for (auto &&i : h3)
48              fileH3 << i << endl;
49      }
50      fileH1.close();
51      fileH2.close();
52      fileH3.close();
53  }
54
55  void handleFile(filesystem::path path)
56  {
57      cout << "Found file: " << filesystem::absolute(path) << endl;
58
59      string line;
60      fstream fileIN;
61      fileIN.open(path, ios::in);
62
63      if (fileIN.is_open())
64      {
65          bool inChevrons = false;
66          int headerType = 0;
67          string chevronsContent = "";
68          while (getline(fileIN, line))
69          {
70              for (auto &&i : line)
71              {
72
73                  switch (i)
74                  {
75                  case '<':
76                      inChevrons = true;
77                      break;
78                  case '>':
79                      inChevrons = false;
80                      if (chevronsContent == "/h1")
81                          headerType = 0;
82                      if (chevronsContent == "/h2")
```

```
83                          headerType = 0;
84                      if (chevronsContent == "/h3")
85                          headerType = 0;
86
87                      if (chevronsContent == "h1")
88                      {
89                          headerType = 1;
90                          h1.push_back("");
91                      }
92
93                      if (chevronsContent == "h2")
94                      {
95                          headerType = 2;
96                          h2.push_back("");
97                      }
98                      if (chevronsContent == "h3")
99                      {
100                         headerType = 3;
101                         h3.push_back("");
102                     }
103
104                     chevronsContent = "";
105                     break;
106                 default:
107                     if (inChevrons)
108                         chevronsContent += tolower(i);
109                     else
110                     {
111                         if (headerType == 1)
112                             h1.at(h1.size() - 1) += i;
113                         if (headerType == 2)
114                             h2.at(h2.size() - 1) += i;
115                         if (headerType == 3)
116                             h3.at(h3.size() - 1) += i;
117                     }
118                     break;
119                 }
120             }
121         }
122     }
123     fileIN.close();
124 }
125
126 int main(int argc, char const *argv[])
127 {
128     if (argc == 1)
```

```
129     {
130         filesystem::path path = "./FullOfHTML/";
131         for (const auto &entry : std::filesystem::directory_iterator(
    path))
132         {
133             filesystem::path file = entry;
134             if (file.extension() == ".html")
135                 handleFile(file);
136         }
137         writeHeaders();
138     }
139     else
140     {
141         for (size_t i = 1; i < argc; i++)
142         {
143             if (!std::filesystem::is_directory(argv[i]))
144             {
145                 cerr << "Это не папка!" << endl;
146                 exit(0);
147             }
148 
149             for (const auto &entry : std::filesystem::directory_iterator
    (argv[i]))
150             {
151                 filesystem::path file = entry;
152                 if (file.extension() == ".html")
153                     handleFile(file);
154             }
155             writeHeaders();
156         }
157     }
158 }
```

Вывод программы

Программа обнаружила файлы .html и, исходя из задания, создала файлы h1.txt, h2.txt и h3.txt

Листинг 11: Вывод программы

```
1 Found file: "/home/chinalap/Документы/YIMP_labs/lab12/files/code/task2
    /./FullOfHTML/example.com.html"
2 Found file: "/home/chinalap/Документы/YIMP_labs/lab12/files/code/task2
    /./FullOfHTML/my.html"
```

Выходные файлы

Листинг 12: Файл h1.txt

```
1 Example Domain
2 aaaH1h1h1h1h1h1h1h1
3 zzzH1h1h1h1h1h1                                    h1h1
```

Листинг 13: Файл h2.txt

```
1 H@H@H@H@HH222222
```

Листинг 14: Файл h3.txt

```
1 iufosokfdh3
```

Вывод

По ходу выполнения данной лабораторной работы, я снова научился работать с файлами (мне уже попадалось такое задания в лабораторной работе №7.2), а также попрактиковался в написании парсеров на C++)).