

```
1: // Copyright 2017 Sam Pickell
2: #include <boost/regex.hpp>
3: #include <cstdlib>
4: #include <iostream>
5: #include <string>
6: #include <fstream>
7: #include "boost/date_time/gregorian/gregorian.hpp"
8: #include "boost/date_time/posix_time/posix_time.hpp"
9:
10: int main(int argc, char* argv[]) {
11:     int line_number = 1;
12:     boost::regex start_up("[0-9]{4}-[0-9]{2}-[0-9]{2}\\s[0-9]{2}:[0-9]{2}:[0-9
] "
13:     "{2}:\\s[()log[.]c[.]166[)]\\sserver\\sstarted\\s");
14:     boost::regex success("[0-9]{4}-[0-9]{2}-[0-9]{2}\\s[0-9]{2}:[0-9]{2}:[0-9
{2} "
15:     "[.][0-9]{3}:INFO:oejs[.]AbstractConnector:Started\\sSelectChannelConnector@
[0-
16:     "9]{1}[.][0-9]{1}[.][0-9]{1}[.][0-9]{1}:[0-9]{4}");
17:     boost::posix_time::time_duration diff;
18:     std::string time_start;
19:     std::string time_end;
20:     std::ifstream fin;
21:     std::ofstream fout;
22:     std::string s, t;
23:     std::string input_file = argv[1];
24:     std::string output_file = input_file;
25:     output_file.append(".rpt");
26:
27:     // Open files
28:     fin.open(input_file.c_str());
29:     if (fin.fail()) {
30:         std::cout << "Failed to open file" << std::endl;
31:         exit(1);
32:     }
33:     fout.open(output_file.c_str());
34:     if (fout.fail()) {
35:         std::cout << "Failed to open output file" << std::endl;
36:         exit(1);
37:     }
38:     // End opening
39:
40:     std::getline(fin, s);
41:     while (!fin.eof()) {
42:         if (boost::regex_match(t, start_up) && boost::regex_match(s, success))
{
43:             // success
44:             fout << line_number << "(" << input_file << "): " << s.substr(0, 1
9)
45:             << " Boot Completed" << std::endl;
46:             // boot time
47:             time_end = s.substr(0, 19);
48:             boost::posix_time::ptime start =
49:                 boost::posix_time::time_from_string(time_start);
50:             boost::posix_time::ptime end =
51:                 boost::posix_time::time_from_string(time_end);
52:             diff = end - start;
53:             fout << "Boot Time: " << diff.total_milliseconds() << "ms"
54:             << std::endl;
55:             fout << std::endl;
56:             t = "empty";
```

```
57:         } else if (boost::regex_match(t, start_up) &&
58:             boost::regex_match(s, start_up)) {
59:             // failure
60:             fout << "**** Incomplete boot ****" << std::endl;
61:             fout << std::endl;
62:             t = "empty";
63:         }
64:
65:         if (boost::regex_match(s, start_up)) {
66:             fout << "=== Device boot ===" << std::endl;
67:             fout << line_number << "(" << input_file << "): " << s.substr(0, 1
9)
68:                 << " Boot Start" << std::endl;
69:             time_start = s.substr(0, 19);
70:             t = s;
71:         }
72:         std::getline(fin, s);
73:         line_number++;
74:     }
75:
76:     fin.close();
77:     fout.close();
78:
79:     return 0;
80: }
```

```
1: C=g++ -g -Wall --std=c++98 -Werror
2: E=.cpp
3: O=main.o
4: P=ps7a
5: BOOST= -lboost_regex -lboost_date_time
6: all: $(P)
7: $(P):$(O)
8:      $(C) -o $(P) $(O) $(BOOST)
9:
10: $(E).o:
11:      $(C) -c $< -o $@
12:
13: clean:
14:      rm $(O) $(P)
```