```
    1: // Copyright 2022 Matthew Lorette Anaya
    2: #include <iostream>
    3: #include <fstream>
    4: #include <string>
    5: #include <exception>
    6: #include <boost/regex.hpp>
    7: #include <boost/date_time.hpp>
    8:
    9: using boost::regex;
   10: using boost::regex_search;
   11: using boost::smatch;
   12: using boost::posix_time::ptime;
   13: using boost::posix_time::time_duration;
   14: using boost::posix_time::time_from_string;
   15:
   16: int main(int argc, char* argv[]) {
   17:
   18:     int lineNum = 1,
   19:         bootStCnt = 0,
   20:         bootDoneCnt = 0;
   21:
   22:     bool bootStarted = false;
   23:
   24:     const std::string bootStMsg = "(log.c.166) server started";
   25:     const std::string bootDoneMsg = "oejs.AbstractConnector:Started "
   26:     "SelectChannelConnector@0.0.0.0:9080";
   27:     regex e("^\\d{4}[-](0[1-9]|1[012])[-](0[1-9]|[12][0-9]|3[01])\\s\\d{2
}"
   28:     "[:]\\d{2}[:]\\d{2}");
   29:     smatch m;
   30:
   31:     ptime tBST, tBDT;
   32:
   33:     std::string s;
   34:     std::string fileName;
   35:     std::ifstream inFile;
   36:     std::ofstream outFile;
   37:
   38:     if (argc != 2) {
   39:         std::cerr << "Usage: ./ps7 device1_intouch.log" << std::endl;
   40:         return -1;
   41:     }
   42:
   43:     inFile.open(argv[1]);
   44:     if (!inFile.is_open()) {
   45:         std::cerr << "Could not open file: " << argv[1] << std::endl;
   46:         return -1;
   47:     }
   48:     s = fileName = argv[1];
   49:     outFile.open(s.append(".rpt.tmp"));
   50:     fileName = fileName.substr(fileName.find_last_of("\\/") + 1);
   51:
   52:     // Temp report file = scanned boot
   53:     while (std::getline(inFile, s)) {
   54:         if (bootStarted) {
   55:             if (s.find(bootDoneMsg) != std::string::npos) {
   56:                 // Boot Done
   57:                 regex_search(s, m, e);
   58:                 tBDT = ptime(time_from_string(m[0]));
   59:                 time_duration td = tBDT - tBST;
   60:
   61:                 outFile << lineNum << "(" << fileName << ") " << m[0]
   62:                 << " Boot Completed" << std::endl
   63:                 << "\tBoot Time: " << td.total_milliseconds() << "ms"
   64:                 << std::endl << std::endl;
```

```
   65:
   66:                 bootStarted = false;
   67:                 bootDoneCnt++;
   68:             } else if (s.find(bootStMsg) != std::string::npos) {
   69:                 // Failed boot
   70:                 regex_search(s, m, e);
   71:                 tBST = ptime(time_from_string(m[0]));
   72:
   73:                 outFile << "**** Incomplete boot ****" << std::endl <<
   74:                 std::endl
   75:                 << "=== Device boot ===" << std::endl
   76:                 << lineNum << "(" << fileName << ") " << m[0]
   77:                 << " Boot Start" << std::endl;
   78:
   79:                 bootStCnt++;
   80:             }
   81:         } else if (s.find(bootStMsg) != std::string::npos) {
   82:             // Sucessfull boot
   83:             regex_search(s, m, e);
   84:             tBST = ptime(time_from_string(m[0]));
   85:             outFile << "=== Device boot ===" << std::endl
   86:             << lineNum << "(" << fileName << ") " << m[0]
   87:             << " Boot Start" << std::endl;
   88:             bootStarted = true;
   89:             bootStCnt++;
   90:         }
   91:         lineNum++;
   92:     }
   93:     inFile.close();
   94:     outFile.close();
   95:
   96:     // Report file done
   97:     s = argv[1];
   98:     s.append(".rpt");
   99:     outFile.open(s);
  100:
  101:     s.append(".tmp");
  102:     inFile.open(s);
  103:     if (!inFile.is_open()) {
  104:         std::cerr << "Could not open file: " << s << std::endl;
  105:         return -1;
  106:     }
  107:
  108:     outFile << "Device Boot Report" << std::endl << std::endl
  109:     << "InTouch log file: " << fileName << std::endl
  110:     << "Lines Scanned: " << lineNum - 1 << std::endl << std::endl
  111:     << "Device boot count: initiated: " << bootStCnt << ", completed: "
  112:     << bootDoneCnt << std::endl << std::endl << std::endl;
  113:
  114:     outFile << inFile.rdbuf();
  115:     inFile.close();
  116:     outFile.close();
  117:
  118:     // remove temp report file
  119:     if (std::remove(s.c_str()) != 0) {
  120:         std::cerr << "Error deleting temp file: " << s << std::endl;
  121:         return -1;
  122:     }
  123:
  124:     return 0;
  125: }
```