# uva-tool

*Command Line Based Tool*
*for uva.onlinejudge.org*
*with features Submitting and uHunting.*

`https://github.com/AHJenin/uva-tool/`

## Developed By

Md. Arafat Hasan Jenin
Email: arafathasanjenin@gmail.com
ID: CE16024
2nd Year 1st Semester
Department of Computer Science and Engineering
Mawlana Bhashani Science and Technology University

## With Advice and instructions of

Mr. Abu Sayem Mohammad Delowar Hossain
Assistant Professor
Department of Computer Science and Engineering
Mawlana Bhashani Science and Technology University

# README.md

## uva-tool

Command Line Based Tool for `uva.onlinejudge.org` with features Submitting and uHunting.

## Dependency

- Command line based tool `curl (https://curl.haxx.se/)`
- nlohmann json (`https://github.com/nlohmann/json`) C++ library

## Installation

- **Install curl**: Download and install curl (`https://curl.haxx.se/`). On ubuntu to install curl execute
```
sudo apt install curl
```
- **Download uva-tool**: Execute command
```
git clone https://github.com/AHJenin/uva-tool.git
```
- **Install uva-tool**: In the downloaded folder `uva-tool`, look for `install.sh` file and execute it (C++ compiler needed here, GNU g++ used in testing).
```
chmod +x install.sh
sudo ./install.sh
```

## Usage Example

Run corresponding commands

- **Login:** `uva-tool -login`
- **Logout:** `uva-tool -logout`
- **Hunt:** `uva-tool -hunt -u felix_halim -r 10`
  This will show last 10 submissions of user `felix_halim`
- **Submit:** Sysntax `uva-tool -submit problem_number problem_path language`
  Example `uva-tool -submit 100 ~/Desktop/100.cpp 5`
  This will submit `100.cpp` named file from `~/Desktop` directory in C++11 (5 for C++11) language with problem number 100.

Language codes: 1 for ANSI, 2 for JAVA, 3 for C++, 4 for Pascal, 5 for C++11, 6 for Python.

# uva-tool.cpp

```
/*
 * FILE: uva-tool.cpp
 *
 * @author: Arafat Hasan Jenin <arafathasanjenin[at]gmail[dot]com>
 *
 * LINK: https://github.com/AHJenin/uva-tool
 *
 * DATE CREATED: 29 Feb 2017
 * LAST MODIFIED: 19-10-17 01:35:51 (+06)
 *
 * DESCRIPTION:
 *
 * DEVELOPMENT HISTORY:
 * Date         Version     Description
 * --------------------------------------------------------------------
 *  29 Feb 2017 0.2         Completed, not tested, BASH
 *  02 Apr 2017 0.4         Without bash script, fully C++, system() used
 *  05 Oct 2017 0.6         Cleanize, nlohmann/json used
 *
 *
 *                 ██╗███████╗███╗   ██╗██╗███╗   ██╗
 *                 ██║██╔════╝████╗  ██║██║████╗  ██║
 *                 ██║█████╗  ██╔██╗ ██║██║██╔██╗ ██║
 *            ██   ██║██╔══╝  ██║╚██╗██║██║██║╚██╗██║
 *            ╚█████╔╝███████╗██║ ╚████║██║██║ ╚████║
 *             ╚════╝ ╚══════╝╚═╝  ╚═══╝╚═╝╚═╝  ╚═══╝
 */

///////////////////////////////////////////////////////////////////////

#include <iostream>
#include <climits>
#include <cmath>
#include <cstring>
#include <cctype>
#include <cstdio>
#include <cstdlib>
#include <iomanip>
#include <utility>
#include <sstream>
#include <algorithm>
#include <stack>
#include <set>
#include <list>
#include <map>
#include <unordered_map>
#include <queue>
#include <deque>
```

```
#include <vector>
#include <stdint.h> //uint32_t
#include <functional>
#include <bitset>

#include <cstdio>
#include <fstream>
#include <iostream>
#include <memory>
#include <stdexcept>
#include <string>
#include <array>
#include "include/json.hpp"
#include <unistd.h> // getlogin()

using namespace std;
using json = nlohmann::json;

typedef long long           ll;
typedef double              lf;
typedef unsigned long long  ull;
typedef pair<int, int>      pii;
typedef vector<pii>         vpii;
typedef vector<int>         vi;

#define __FastIO        ios_base::sync_with_stdio(false); cin.tie(0)

#define forr(i, a, b)   for (__typeof (a) i=a; i<=b; i++)
#define rof(i, b, a)    for (__typeof (a) i=b; i>=a; i--)
#define rep(i, n)       for (__typeof (n) i=0; i<n; i++)
#define forit(i, s)     for (__typeof ((s).end ()) i = (s).begin (); i != (s).end(); ++i)
#define all(ar)         ar.begin(), ar.end()
#define fill(ar, val)   memset(ar, val, sizeof(ar))
#define clr(a)          memset(a, 0, sizeof(a))

#define nl              cout << '\n';
#define sp              cout << ' ';
#define gc              getchar
#define chk             cout << "##########\n"
#define pb              push_back
#define debug1(x)       cout << #x << ": " << x << endl
#define debug2(x, y)    cout << #x << ": " << x << '\t' << #y << ": " << y << endl
#define debug3(x, y, z) cout << #x << ": " << x << '\t' << #y << ": " << y << '\t' << #z << ": " << z << endl

#define max(a, b)       (a < b ? b : a)
#define min(a, b)       (a > b ? b : a)
#define sq(a)           (a * a)

#define PI              acos(-1.0)
#define INF             0x7fffffff
```

```
#define MOD             1000000007
#define EPS             1e-7
#define MAX             10000005

#ifdef __linux__
const string szHome = getlogin();
const string pid_num_cvs = "/usr/share/uva-tool/pid-to-num.cvs";
const string cookie_file =
    "/home/" + szHome + "/.cache/uva-tool/uva.onlinejudge.org_cookie.txt";
//const string err = " 2> ~/.uva-tool/err.log";
const string err = " 2> /dev/null";
const string curlfunc = "curl";
#elif __WIN32__
// win
#endif

/////////////////////////// START HERE //////////////////////////

string system_exec(const char *cmd) {
    char buffer[128];
    string result = "";
    FILE *pipe = popen(cmd, "r");
    if (!pipe) throw runtime_error("popen() failed!");
    try {
        while (!feof(pipe)) {
            if (fgets(buffer, 128, pipe) != NULL)
                result += buffer;
        }
    } catch (...) {
        pclose(pipe);
        throw;
    }
    pclose(pipe);
    return result;
}

class Problem {
private:
    string line, pname;
    int pid, pnum;
    map<int, pair<int, string> > pid_name;
    void split(string splitit) {
        istringstream ss(splitit);
        vector<string> result;
        string token;
        while (getline(ss, token, '~')) {
            result.push_back(token);
        }
        pid = stoi(result[0]);
        pnum = stoi(result[1]);
        pname = result[2];
    }
```

```

public:
    Problem() {
        ifstream fp(pid_num_cvs);

        if (!fp) {
            cerr << "Unable to open file : " << pid_num_cvs << "\n";
            throw 1;
        }

        while (getline(fp, line)) {
            split(line);
            pid_name[pid] = make_pair(pnum, pname);
        }
        fp.close();
    }
    string name(int pidd) {
        return pid_name[pidd].second;
    }

    int number(int pidd) {
        return pid_name[pidd].first;
    }
};


class Submission {

private:
    int range;
    json json_subs;
    Problem *problem;
    void convertunixtime(long long u) {
        time_t     now;
        struct tm *ts;
        char       buf[80];
        now = u;
        /* Format and print the time, "ddd yyyy-mm-dd hh:mm:ss zzz" */
        ts = localtime(&now);
        strftime(buf, sizeof(buf), "%d-%m-%y %I:%M:%S%p %Z", ts);
        cout << buf;
    }

    void sid(long long sub_id) {
        cout << left << setw(11) << sub_id;
    }

    void problem_name(int pid) {
        cout << problem->number(pid) << '-' \
             << left << setw(36) << problem->name(pid) << '\t';
    }

    void verdict(int ver) {
        if (ver == 90)
            cout << "\x1b[38;2;0;195;0m\033[1mAccepted\033[0m\x1b[0m\t";
        else if (ver == 80)
            cout << "\x1b[38;2;102;102;0m\033[1mPresentationE\033[0m\x1b[0m\t";
        else if (ver == 70)
            cout << "\x1b[38;2;255;0;0m\033[1mWrong answer\033[0m\x1b[0m\t";
        else if (ver == 60)
            cout << "\x1b[38;2;51;51;255m\033[1mMemory limit\033[0m\x1b[0m\t";
        else if (ver == 50)
            cout << "\x1b[38;2;0;0;255m\033[1mTime limit\033[0m\x1b[0m\t";
        else if (ver == 45)
            cout << "\x1b[38;2;96;96;96m\033[1mOutput limit\033[0m\x1b[0m\t";
        else if (ver == 40)
            cout << "\x1b[38;2;0;204;204m\033[1mRuntime error\033[0m\x1b[0m\t";
        else if (ver == 30)
            cout << "\x1b[38;2;204;204;0m\033[1mCompile error\033[0m\x1b[0m\t";
        else if (ver == 35)
            cout << "\x1b[38;2;96;96;96m\033[1mRestricted function\033[0m\x1b[0m\t";
        else if (ver == 20 || ver == 0)
            cout << "\x1b[38;2;96;96;96m\033[1mIn queue\033[0m\x1b[0m\t";
        else if (ver == 15)
            cout << "\x1b[38;2;96;96;96m\033[1mCan't be judged\033[0m\x1b[0\t";
        else if (ver == 10)
            cout << "\x1b[38;2;96;96;96m\033[1mSubmission error\033[0m\x1b[0m ";
        else  cout << "UNKNOWN VERDICT\t";
    }

    void runtime(int rt, int ver) {
        double runt = (double) (rt) / 1000.0;
        if (ver == 40 || ver == 45 || ver == 50 || ver == 60 || ver == 70 || \
                ver == 80 || ver == 90 )
            cout << left << setw(6) << fixed << setprecision(3) << runt << "\t";
        else cout << left << setw(8) << "-";
    }

    void uva_rank(int rnk) {
        if (rnk != -1) cout << left << setw(6) << rnk << "\t";
        else cout << left << setw(8) << "-";
    }

    void language(int lan) {
        if (lan == 6) cout << "Python" << "\t";
        else if (lan == 5) cout << "C++11" << "\t";
        else if (lan == 4) cout << "Pascal" << "\t";
        else if (lan == 3) cout << "C++" << "\t";
        else if (lan == 2) cout << "Java" << "\t";
        else if (lan == 1) cout << "ANSI C" << "\t";
        else cout << "UNKNOWN" << "\t";
        cout << "   ";
    }
    void date(long long utime) {
        convertunixtime(utime);
        cout << '\t';
        cout << "\n";
    }

public:
    Submission(string &input) {
        try {
            json_subs = json::parse(input);
        } catch (json::parse_error &e) {
            std::cerr << e.what() << std::endl;
            throw 2;
        }
        range = (int) json_subs["subs"].size();
    }

    void show(Problem &input) {
        problem = &input;
        cout << left << setw(11) << "Sub ID" << setw(40) << "Problem\t" << \
             "Verdict\t\t" << "Runtime\t" << "Rank\t" << "Language   " << \
             "Date\n";
        cout << "=============================================================";
        cout << "=============================================================\n";
        for (int i = range - 1; i >= 0; i--) {
            sid(json_subs["subs"][i][0]);
            problem_name(json_subs["subs"][i][1]);
            verdict((json_subs["subs"][i][2]));
            runtime(json_subs["subs"][i][3], json_subs["subs"][i][2]);
            uva_rank(json_subs["subs"][i][6]);
            language(json_subs["subs"][i][5]);
            date(json_subs["subs"][i][4]);
        }
    }
};


void hunt(string name, string range) {
    string cmd, uid, subs_usr_last, tmp;
    Problem problem;
    cmd = curlfunc + " http://uhunt.onlinejudge.org/api/uname2uid/" + \
          name + err;
    uid = system_exec(cmd.c_str());
    if (uid == "") {
        cerr << "Unable to connect or curl executing error\n";
        throw 3;
    }

    cmd.clear();
    cmd = curlfunc + " http://uhunt.onlinejudge.org/api/subs-user-last/" + \
          uid + "/" + range + err;
    subs_usr_last = system_exec(cmd.c_str());
    Submission usr_last_subs(subs_usr_last);
    usr_last_subs.show(problem);
}

class submit {
private:
    string cookie_jar = cookie_file;
    string submitlink = "https://uva.onlinejudge.org/index.php\
?option=com_onlinejudge&Itemid=25&page=save_submission";
    string formdata() {
        string cmd = curlfunc + " -f -L -s http://uva.onlinejudge.org |";
        cmd += " grep -B8 'id=\"mod_login_remember\"' | awk '{print $3 \" \"$4}'";
        string str = system_exec(cmd.c_str());
        if (str == "") return str;
        size_t start = str.find("name=\"");
        while (start != string::npos) {
            str.erase(start, 6);
            start = str.find("name=\"");
        }

        start = str.find("\" value=\"");
        while (start != string::npos) {
            str.replace(start, 9, "=");
            start = str.find("\" value=\"");
        }
        start = str.find("\"");
        while (start != string::npos) {
            str.erase(start, 1);
            start = str.find("\"");
        }
        str.erase(str.find("remember id=mod_login_remember"), 30);
        start = str.find("\n");
        while (start != string::npos) {
            str.replace(start, 1, "&");
            start = str.find("\n");
        }
        str.erase(str.find("&&"), 2);

        string usr, pass, remember;
        cout << "Input username: ";
        cin >> usr;
        cout << "Input password: ";
        cin >> pass;
        cout << "Remember? [y/n] ";
        cin >> remember;
        remember = (remember == "y" or remember == "Y" ? "yes" : "no");
        string usrpass = "username=" + usr + "&passwd=" + pass + \
                         "&remember=" + remember + "&";
        return usrpass + str;
    }

public:
    bool logout() {
        string cmd = "rm " + cookie_jar + " 2>&1";
        string str = system_exec(cmd.c_str());
        if (str == "") return true;
        return false;
    }

    bool login() {
        string data = formdata();
        if (data == "") {
            cout << "Can not connect to www.uva.onlinejudge.org\n";
            return false;
        }
        string cmd = curlfunc + " -X POST -f -L -s --compressed ";
        cmd += "--cookie-jar " + cookie_jar + " --data \"";
        cmd += data;
        cmd += "\" \"https://uva.onlinejudge.org/index.php\
?option=com_comprofiler&task=login\"";
        string str = system_exec(cmd.c_str());
        if (str.find("My Account") != string::npos
                and str.find("Logout") != string::npos) {
            return true;
        }
        return false;
    }

    void problem_submit(string pnumber, string ppath, string plang) {
        string cmd = curlfunc + \
                     " -X POST -f -L -s -w '%{url_effective}' " \
                     "--compressed --cookie " \
                     + cookie_jar + " --cookie-jar " + cookie_jar + \
                     " --form localid=" + pnumber + " --form language=" + plang + \
                     " --form \"codeupl=@" + ppath + "\" \"" + submitlink + "\"";
        string str = system_exec(cmd.c_str());
        size_t notlogin = str.find("You are not authorised to view this resource");
        size_t subid = str.find("mosmsg=Submission+received+with+ID+");
        if (str == submitlink)
            cout << "Can not connect to www.uva.onlinejudge.org\n";
        else if (notlogin != string::npos) {
            cout << "Not logged in\n";
        } else if (subid != string::npos) {
            size_t sz = str.find("\"", subid);
            cout << "Submission received with ID ";
            for (size_t i = subid + 35; i < sz; i++)
                cout << str[i];
            cout << '\n';
        }
    }
};

int main(int argc, char *argv[]) {

    try {
        if (!strcmp(argv[1], "-hunt") && !strcmp(argv[2], "-u")
                && !strcmp(argv[4], "-r") && argc == 6) {
            hunt(argv[3], argv[5]);
        } else if (!strcmp(argv[1], "-submit") && argc == 5) {
            submit sub;
            sub.problem_submit(argv[2], argv[3], argv[4]);
        } else if (!strcmp(argv[1], "-login") && argc == 2) {
            submit sub;
            if (sub.login()) {
                cout << "Logged in\n";
            } else cout << "Log in failed\n";
        } else if (!strcmp(argv[1], "-logout") && argc == 2) {
            submit sub;
            if (sub.logout()) {
                cout << "Logged out\n";
            } else cout << "Not logged in\n";
        } else cout << "Wrong option\n";
    } catch (int exception) {
        cerr << "Exeption no: " << exception << '\n';
    }
    return 0;
}
```

# install.sh

```
g++ -g -Wall -Wextra -Wshadow -Wfloat-equal -pedantic -std=c++11 -O2
-Wformat=2 -Wconversion -lm -o uva-tool uva-tool.cpp
sudo mv uva-tool /usr/bin/
sudo mkdir /usr/share/uva-tool/
sudo cp pid-to-num.cvs /usr/share/uva-tool/
mkdir ~/.cache/uva-tool/
```

# LICENSE

```
MIT License

Copyright (c) 2017 Arafat Hasan

Permission is hereby granted, free of charge, to any person obtaining a copy
of this software and associated documentation files (the "Software"), to deal
in the Software without restriction, including without limitation the rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all
copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

```