2017 Atlantic Canadian Preliminary Programming Contest

2017-10-13

1 Background

The Atlantic Canadian Preliminary Programming Contest is a sub-regional contest for the ACM International Collegiate Programming Contest (ICPC), an international, multi-tier, team-based competition [1]. Teams are a maximum of three students with at most 5 years of university computer science education. The best two or three teams from the Atlantic Canadian Preliminary contest are invited to the North American Northeast Regional Contest. Participation of Atlantic Canadian teams in the Regional contest is usually remotely via a Satellite contest at an Atlantic Canadian University.

2 Schedule, Locations

In the following HH is Head Hall, accessible from Dineen Drive (or via Gillin Hall from Windsor Street), and IT is the Information Technology Center, accessible from Windsor Street. 'C' is ground level and 'D' is one floor up.

Briefing 11:45 Friday, HH C9

Contest 12:30 - 17:30 Friday, ITD 414 and ITD415

Coaches' Lounge 12:30 - 17:30 Friday, ITC307. We'll have the scores, and probably some coffee for coaches.

Banquet 18:00 Friday, Student Union Building (follow the crowd).

Closing / Announcement of Winners 17:00 Saturday HH C13 (Dineen Auditorium)

3 Contest Computing Environment.

Each team will be provided with one Linux workstation running CentOS 7.3.

- The workstations will be in rooms ID414 and ID415 on the second floor of the information technology center.
- A username and password for the workstation will be provided at the contest briefing.

[1] `http://icpc.baylor.edu`

- You are not allowed to download any software or configuration not on the provided Linux workstations, other than test data from Kattis.

3.1 Language Runtimes

The main version gap to be aware of is C/C++ where the local version of `gcc/g++` is about 1 year older than the one on Kattis. In particular:

C Gcc is version 4.8.5. It includes "substantially complete" support for c99.

C++ G++ is also version 4.85. It includes what is described as *experimental* support for `C++11`. Use of `C++11` features is thus at your own risk.

Java Java is version `1.8.0_144`; this is slightly ahead of Kattis.

python2 python 2.7.5 is available as `python`. This is slightly behind Kattis.

python3 Python 3.5.2 is available as `python3`. This is the same as Kattis.

3.2 Editing

You may use any of the following text editors; no other text editor or IDE is permitted.

- emacs
- gedit
- nano
- vim

3.3 Compiling and Running

The following command lines assume you are working in a terminal, with the source code and test data in the current directory. For `C` and `C++` we suggest using roughly the same flags as used on Kattis (in particular the `-std` flag makes a big difference). For other languages this is less important, but in case of doubt the flags used by the judging software are documented on Kattis.

3.3.1 C

Omit the `-static` flag suggested on kattis.

```
$ gcc -g -O2 -std=gnu99 -o solution solution.c -lm
$ solution < test.in > test.out
```

3.3.2 C++

Omit the `-static` flag suggested on kattis.

```
$ g++ -g -O2 -std=gnu++11 -o solution solution.cc -lm
$ solution < test.in > test.out
```

3.3.3 Java

The following simple invocations should be sufficient for Java; in case of difficulty see the Kattis docs for the flags used on Kattis.

```
$ javac Solution.java
$ java Solution < test.in > test.out
```

3.3.4 Python2

```
$ python solution.py < test.in > test.out
```

3.3.5 Python3

```
$ python3 solution.py < test.in > test.out
```

3.4 Browsers

You are only allowed to use web sites linked from the Kattis language specific help, and Kattis itself. The following browsers are available

chrome Version 60.0.3112.113 (Official Build) unknown (64-bit)

firefox 52.2.0 (64-bit)

4 Refreshments

- Water and light snacks will be provided outside the labs.
- Please do not consume food in the labs.
- Students are welcome to bring their own snacks to consume outside the labs.

5 Kattis

Submission and judging of problems will be via a web based system run by `kattis.com`. Teams are strongly encouraged to become familiar with the Kattis interface at `https://open.kattis.com`. The contest URL is `https://atlantic-canada17.kattis.com`

5.1 Accounts

All contestants must have registered via

```
https://icpc.baylor.edu/regionals/finder/na-acpc-2017
```

A Kattis account will be created based on the email used to register with ICPC. If you end up with multiple Kattis Accounts, make sure you use the one whose email address matches your ICPC registration.

5.2 Submission

Submit your solutions using the Submit link on Kattis (after logging in and joining the contest). Only submit source code.

5.3 Solving problems

You can find language specific help linked from Kattis[2]. A key point is that you should take all input from `stdin`, and write to `stdout`. Debugging output on `stderr` is *ignored*. Submissions should exit with code 0, or they will be rejected. You can test the return code of your program with the following

```
$ ./hello < /dev/null > hello.out || echo "bad exit code"
```

6 Documentation and reference material

- The only online documentation permitted is that linked from Kattis, in particular language API reference for C++, Java, and Python. You can also use `https://cppreference.com` for C language reference.
- Per ACM rules, you are permitted 25 pages of printed reference material

7 Scoring

Kattis follows the ACM rules for scoring. In particular there is a 20 minute penalty for rejected runs. The full rules are as follows[3].

> Teams are ranked according to the most problems solved. For the purposes of awards, or in determining qualifier(s) for the World Finals, teams who solve the same number of problems are ranked by least total time. The total time is the sum of the time consumed for each problem solved. The time consumed for a solved problem is the time elapsed from the beginning of the contest to the submittal of the accepted run plus 20 penalty minutes for every rejected run for that problem regardless of submittal time. There is no time consumed for a problem that is not solved.

8 Printing

There is a print queue `fcs_local` available in both contest labs. You can print to it with, e.g.

```
$ lpr -Pfcs_local Solution.java
```

Slightly prettier output is possible with

```
$ print_local Solution.java
```

This is just a script that calls `a2ps` with appropriate parameters. You can also select the destination `fcs_local` from various printing dialogs. Please be reasonable in your use of printing; there a soft limit of 30 pages of printing per team.

[2] `https://open.kattis.com/help`
[3] `https://icpc.baylor.edu/regionals/rules`