



2007 ACM SOUTH PACIFIC REGIONAL PROGRAMMING CONTEST

Welcome to the 2007 South Pacific ACM Regional Programming Contest! Every year this competition gathers university teams from the South Pacific Region to compete solving problems posed by a team of judges. The winning teams will be invited to participate in the 2008 ACM Programming Contest World Finals next year. In order to make the day as enjoyable as possible, please keep to the rules and follow the guidelines carefully.

GENERAL DETAILS

The competition will run for 5 hours. There are a number of problems to solve, one computer for each team of three, and shared printing facilities.

1 General instructions

Please note these important issues relating to the actual running of the competition.

- ⇒ Your contest pack should identify your team by number and name.
- ⇒ We have 10 problems for the 2007 Contest. Although the classification as easier or harder is very subjective, the problems considered easier by the judges are placed at beginning of the set and they have been sorted in ascending order of difficulty.
- ⇒ Your solutions are judged by comparing your program's output to our correct output. Therefore, you *must stick exactly to the output format* described in the problem statement, including spaces, new lines and letter cases.
- ⇒ From time to time teams will find a problem statement difficult to understand or confusing, and will request a clarification. If the issue is deemed by the judges to be clearly specified in the problem statement, a feedback message to “read the statement carefully” or to “read the sample input-output thoroughly” will be returned only to the requesting team. If the clarification is deemed to be due to an ambiguity or an imprecision in the problem statement, a clarification message will be sent to all the competing teams in all sites.

2 Conduct during the competition: do's and don'ts

- ⇒ All accesses from a team's workstation are monitored; any attempt at unauthorised access such as email will result in instant disqualification. Using the internet in any way is not permitted.
- ⇒ Teams are expected to work quietly and avoid being disruptive. The judges reserve the right to disqualify a team in any way disrupting the running of the competition.
- ⇒ Calculators, mobile phones and electronic Personal Assistant devices are not allowed. These should be left with your coach or a contest official.
- ⇒ Contestants may use any printed material, including books, but must not bring any machine-readable information into the lab. Please leave all disks with your coach or a contest official.

3 Submissions and feedback

- ⇒ Your solutions may be written in one of the available languages. See Site Details for a list of languages available at your site.
- ⇒ For all languages, all code to solve a problem must be included in a single source file. The file name should be based on the problem number and with the proper language specific extension, such as **P1.java**, **P1.c**, **P1.cpp**. Failure to follow this rule will lead to a rejection as "Compilation error".
- ⇒ For Java, the name of the source file and the name of the main class must be identical, e.g., the main class in **P1.java** must be named **P1**. Failure to follow this rule will lead to a rejection as "Compilation error".
- ⇒ Submissions are sent to the judges through PC². A separate set of instructions for using this competition software is provided.
- ⇒ You will be informed by PC² of the result of your submission:
 - A successful submission will return a 'Submission Accepted' message from the PC² software.
 - A failed submission will be returned with an error message (see details below)
- ⇒ Solutions will be tested by compiling them using the command line compilers with the program's name as the only argument. It is possible that a conceptually correct solution will exceed the limits of the machine used for judging, for example will run out of the default stack space, and generate a run-time exception. However, all problems have solutions that work with the default configurations. You may wish to take this into account if you keep getting inexplicable run-time errors.

- ⇒ Many problems have an obvious solution that may take a long time to complete and is therefore wrong in this context. It is also possible that you submit an incorrect program that goes into an endless loop on the test data. Thus, no submission will be allowed to run for more than 2 minutes on the judge's machine, (adjusted if necessary to allow for differences between centres). Submissions that 'hang' and are stopped for this reason will be returned with 'Time limit exceeded'.
- ⇒ The following messages will be issued as a result of an incorrect submission. The feedback will be provided only on the first of the errors encountered by the judges.

1. *Compilation Error*

Any error that prevents the program from compiling and linking successfully.

2. *Run-Time Error*

Issued as a result of any error which occurs at execution time.

3. *Time Limit Exceeded*

Programs exceeding two minutes of execution time will be issued this message. It will also be issued when a program enters an "obvious" infinite loop.

4. *Wrong Answer*

Issued when some or all of the output is incorrect, excluding errors in formatting (see below).

5. *Output Format Error*

This result is issued when all of the cases in the team's submission are basically correct (all of the technical parts of the problem work correctly) but the output is not in the format requested in the problem statement. If one or more of the test cases fail, the message should be Wrong Answer. White space at the end of a line, or at the end of the output, is ignored.

6. *Other -Contact Staff*

This will be used in exceptional circumstances only. It allows for human intervention in a mechanical system!

Basic Acceptance Rejection Rules

If the program executes and produces output within the time limit, that output will be captured in a file. The file is then judged as follows:

1. Trailing white space (tab and space characters) is stripped from your output.
2. Trailing blank lines are stripped from your output.
3. The trimmed output file is then compared with a similarly treated model answer.
4. If the files are identical, your solution is **accepted**.
5. Otherwise, all white space is stripped from every line of your output, leading and embedded blank lines are removed, and all characters converted to lower case.
6. The treated output is compared with a similarly treated model answer.
7. If the files are now identical, it is an **output Format Error**.
8. Otherwise it is a **wrong answer**.

TECHNICAL DETAILS

4 Input/Output File Format

- ⇒ All input and output files use the ASCII encoding, 1 character per byte.
- ⇒ All files use only 7-bit ASCII characters in the printable range space (' ') to tilde (~), plus end-of-lines terminators (more details in the END-OF-LINES section) – tabs ('\t') should not be used.
- ⇒ Unless otherwise stated, all input lines contain up to 1000 characters (not counting line terminators). This is a rounded upper-bound and the actual figure could be less or much less. Exceptions with longer lines are explicitly mentioned by footnotes in the problem specs.
- ⇒ Ignore any trailing spaces at the end of lines and trailing blank lines at the end of files (if any). Correspondingly, the judging system will ignore any such trailing spaces or blank lines (as explained previously in the **Basic Acceptance Rejection Rules** section).

Source File Format And Naming Conventions

- ⇒ As a general rule, source files must also use the ASCII encoding, 1 character per byte. Other encodings (such as UTF-8 with byte order marks) will be accepted if used by one of the editors officially recommended by your local site.
- ⇒ All your program sources must use only 7-bit ASCII characters in the printable range space (' ') to tilde('~'), plus tabs ('\t'), and end-of-lines terminators (more details in the END-OF-LINES section).
- ⇒ All source lines must only contain up to 250 characters (not counting line terminators).

End-Of-Lines

Valid end-of-line terminators are CR-LF ('\r\n'), LF ('\n'), and CR ('\r'). These terminators are typically platform-specific and your local site will advise you which end-of-line terminators are actually used in your data files.

5 Reading From Input and Writing To Output

According to the local rules of your site, you will be required to either:

- ⇒ Read directly from a given file and write directly to another file, or
- ⇒ Read from a redirected ('<') standard input stream (i.e., use **System.in**, **stdin**, **cin**, **Console.in**, or equivalent) and write to a redirected ('>') standard output stream (i.e., use **System.out**, **stdout**, **cout**, **Console.out**, or equivalent – never write your output to the standard error stream). Hypothetical example:

```
P1.EXE < P1_in_1.txt > MyOutput.txt
```

Your local site will advise you more specific and accurate details on the correct procedure for your site. Please follow it literally!

Site Specific Information

Each site will provide contestants with information required to login and use the local system.