Code Test Generic Rules

Please read these instructions carefully, particularly those under **Submitting your code**.

We will accept submissions in any of the following languages. Submitted code will be compiled/run under Ubuntu

GNU/Linux using the language versions shown below.

Language	Compiler/Interpreter Version
Java	JDK 1.8 (Java 8) – see note 1
JavaScript	Node.js 6.2
Python	2.7

Note 1: Java code will be assumed to be platform independent and to compile under JDK 1.8.

These languages represent a selection of those in most widespread use today. All of them are in use within BT, but so are many others – a number of which are more widely used than some of those that appear in the table! If you have a compelling reason to want to use a different language – such as having no experience with any of those listed – then please contact us to determine whether we have the capability to accept such a submission.

Command-line ("console") applications

Unless otherwise stated in the description of the problem, programs must be command-line (“console” or “terminal”) applications. Under Windows this means that they will be run from the command shell, `cmd.exe`. Under Linux, MacOS or other UNIX-like systems this means that that will be run from a `bash` shell within a terminal window. This applies even if the more normal mode of using the language would be with another kind of interface (such as a web application for JavaScript or PHP).

Unless otherwise stated programs will take a number of command-line arguments and write their results to standard output (the console/terminal). So they will be run something like this:

```
MyProgram input.txt foo bar
```

This means that MyProgram will be invoked with three command-line arguments, “input.txt”, “foo” and “bar”.

Your program should print only the specified output (or an error message where appropriate), and no additional text, blank lines etc. To be completely clear, A Java program with its main class compiled into MyProgram.class will be run as: java MyProgram input.txt foo bar, JavaScript in MyProgram.js will be run

as: node MyProgram.js input.txt foo bar and Python in MyProgram.py will be run as: python MyProgram.py input.txt foo bar

Note: *MyProgram* is just an example – you should give your program a more meaningful name!

Submitting your code

Your submitted program code may consist of multiple source files. In general it is fairly obvious how programs should be built or run, but you should include a prominent "README.txt" file that gives instructions if it may be unclear how to do so. This is particularly important if you are using different versions of the languages than those specified above.

You can also include with your program code any other data files, unit tests and so on to demonstrate that it works – but do not include any pre-compiled files (e.g. .jar, .class, .obj, .exe, .dll etc.). You may either send a link to files in a publicly-accessible code repository (e.g. GitHub or BitBucket) or e-mail your code as a zipped attachment. Because of problems in the past with zipped files being blocked by BT's and other organisation's filewalls, we recommend that any zip file should be encrypted with the password btcodetest. On Windows you can use the free, open-source, 7-Zip [http://www.7-zip.org] tool to do this. Instructions on how to create password-protected zip files on a number of operating systems can be found easily on the Internet, for example at howtogeek.com [http://www.howtogeek.com/203590/how-to-create-secure-encrypted-zip-or-7z-archives-on-any-operating-system/].

Other requirements

Your program must not rely on any libraries that do not form part of a normal base installation of the language, but you may use anything from within the language's standard libraries. So for example, unless explicitly asked to do so, we wouldn't expect you to implement your own sort routine if there is already one available in the standard libraries. Please be aware that your program should, where possible, avoid operating system-specific assumptions such line ending characters or file-name path delimiters.

The restriction on additional libraries does not apply to any tests or other collateral that you may wish to submit. So, for example, it's fine to use JUnit for Java, RSpec for Ruby and so on. We will test your code, and expect you to have done so prior to submitting it. You will **not** automatically "fail" if we uncover bugs or limitations in your program. However, please be prepared to be challenged over improvements and possible fixes when we discuss your code with you. We will test your code with input sets other than those supplied in the problem description, so please expect that.

You may find that the specification of the problem seems ambiguous, or not as complete as you would like. That is, of course, part of the reality of software development! If you find that you have to make certain assumptions in order to implement your program then please do so, but make it clear what those assumptions are. We will not provide you with further examples or test cases – although if you

find something in the problem specification that seems contradictory or just plain wrong then please let us know. It should go without saying that the program must be your own work. You are free to take inspiration and even small code snippets from other sources, but you must acknowledge if you do so with appropriate comments. In any event you will be expected to be able to explain in detail how every part of your program works.