Introduction

Write a program that automates the sequence of test programs to be run.

- Resources:
- Language: Perl or Python
- Time limit: 24 – 48 hours.

Problem Statement

Intro:

We want to automate the task of running a composite system test. Such a system test consists of various steps, and each step can have one of more test tasks running concurrently. We want to build an infrastructure program that can a system test specification in an configuration file of proprietary ASCII format. A given step may require running multiple test tasks concurrently and waiting until all of them complete.

At the end of the system test, publish the following:

1. Run time of each step.
2. Run time of each test task in each step.
3. Status and output of each test task in each step.
4. Start/stop time of each task.

Additionally, to get a sense of the environment before and after the test, provide the following global information:

1. ps -ef output before running the first step and after running all steps.
2. plot of free memory measured at 1 second intervals during the run.

Test configuration:

Various different configuration file formats are valid. You can define the appropriate format for the file as part of the solution. The configuration file must be an ASCII file. One basic format for example can be as follows:

```
SchedulerTest
SqlTest1, SqlTest2, SqlTest3
IOTest
```

In this format, each line represents a step of the system test. The second line indicates that SqlTest1, SqlTest2 and SqlTest3 must be started concurrently. The next step should not be begun until all 3 are completed. This is only one possible format. You can use this (or something like this) or use XML or JSON if you wish. Any configuration file syntax that supports defining steps with multiple test tasks per step is adequate.

Test functions:

In general, the functions corresponding to test tasks will be provided by other users. However, for now, please provide some dummy test task functions. The test task functions can be stub implementations that sleep or print some message and exit. This automation system will be used by developers to write new system tests. So keep in mind the ease with which they can define new configuration files and test task functions and run them using the infrastructure you write.

Test output:

At the end of the test, certain output stats for each step and test need to be output. These are listed above. You can pick the location and format of the output e.g. output can be to stderr/stdout, or to separate files. If you print to stdout, feel free to perform any pretty printing/formatting as appropriate. The design of the output locations/format is open for design.

We want to see the process list output (i.e. ps -ef output) for before and after the full run in two files psBefore.txt and psAfter.txt. These files should be created in the same directory as the test.

Finally, we want to see a plot of memory consumption during the run. The memory usage samples can be gathered from a program like vmstat (run vmstat 1). The choice of the plotting tool/software is up to you. Assume that the test is being run from a GUI environment.

Goals:

Key areas of focus for the assignment:

1. Correctness and completeness
   - code should run correctly on submitted sample and perform required tasks.
2. Clean, extensible software design
   - easy to implement new task steps
3. Clean interface design
   - intuitive command line and configuration file syntax
4. Clean visualization of data
   - easy to read graph/reports printed at end
5. Knowledge of Python/Perl

This is an unstructured problem by design. The programmer can and should make assumptions on the requirements during implementation. These assumptions should be briefly documented in the code or a README. The programmer is supposed to make reasonable design choices along the way during implementation.

Timing guidelines:

This is a self-timed assignment. Assignment timing guidelines:
1. You can look at the problem at any time, and start thinking about it.

2. Do not spend more than 5 hours in total doing any on-paper design, or any coding.
3. You DON'T have to spend 5 hours on the assignment. That is just an outer limit.

Candidates would need at least 2 hours of design/coding to address at least few of the key focus areas of the problem. Please send us how long you spent coding and the areas you focused on so that we can take that into account.

Deliverables:

Please provide us:

1. All code files written for the infrastructure.
2. Sample configuration file for a system test.
3. Stub implementations of test functions corresponding to test tasks in the sample configuration file.
4. Command line arguments on how to run your sample tests.
5. Time taken, and any key areas of focus, known deficiencies etc.