Melissa Jones, Nicholas Benyo, Scott Moser
Seattle University – Professor McKee
CPSC5210-01
June 6, 2019

Milestone 2: Stress Test Script

I. stressTestApp.sh (source)

```
#!/bin/bash
#================================================================================
# Team D'Buggers (Team 7)
# Scott Moser, Nicholas Benyo, Melissa Jones
# Professor McKee
# CPSC 5200-01
# 6 June 2019
#                               Milestone #2
#
# File: stressTestApp.sh
#
# Description:
# This shell script kicks of a specified number of background instances of the
# test suite each executing a specified number of times. Results are displayed
# to the user.
#
#   DEPENDENCIES, LIMITATIONS, & DESIGN NOTES:
#       Dependencies :
#       Design Notes :
#           1. Each test suite instance is kicked off as a background task.
#           2. The PID of each instance is saved, then a join is performed.
#           3. Once all instances are complete, their logs are combined.
#           4. The combined log is then parsed to determine overall PASS/FAIL.
#           5. Results and statistics are calculated and displayed to the user.
#           6. Console output is logged to $LOG_FILE
#       Limitations :
#           1. Due to memory limitations on SU's CS1 server, errors may be seen
#              if too many background instances are requested (e.g. >= 5).
#
#   Example Usage:
#   "./stressTestApp.sh 2 5"  # Launches two instances of the test suite in the
#                               background, each executing 5 loops.
#================================================================================

#set -o errexit
set -o pipefail
set -o nounset
#set -o xtrace

#================================================================================
# Constants
#================================================================================
NUM_ARGS=2

LOG_PREFIX="stressTestLog"
LOG_EXT="txt"
LOG_FILE="${LOG_PREFIX}_main.${LOG_EXT}"
```

```
#================================================================================
# Script
#================================================================================

# Check number of arguments
if [ "$#" != "$NUM_ARGS" ]; then
    echo "ERROR: Invalid number of command-line arguments!"
    echo "Usage:"
    echo "   ./stressTestApp <numInstances> <numIter>"
    echo "     - <numInstances> - How many instances of runTestSuite (background
process) to run"
    echo "     - <numIter> - Number of iterations for each instance to run"
    exit 1
fi

# Basic input validation for number of iterations
if [ "$1" -le "0" ]; then
    echo "ERROR: Caller must enter numInstances of 1 or more!"
    exit 2
fi

# Basic input validation for number of iterations
if [ "$2" -le "0" ]; then
    echo "ERROR: Caller must enter numIter of 1 or more!"
    exit 3
fi

# Remove previous log files
echo Cleaning up stale log files...
rm -f -v $LOG_PREFIX*.$LOG_EXT

echo -e "
================================================================================
Stress Test Execution
================================================================================
" | tee -a $LOG_FILE

numInstances=$1
numIter=$2

# Capture START of test section
echo "Start date/time:" $(date) | tee -a $LOG_FILE
start=$SECONDS

# Kick off the background processes
for i in `seq 1 $numInstances`; do
    echo "Launching test suite instance $i of $numInstances..." | tee -a $LOG_FILE

    # This block gets spun off as a non-blocking background thread.
    # Avoid resource sharing (e.g. writing to same log file) between threads
    # to avoid performance bottlenecks inside this block.
    (
        # Obtain a unique index for this block
        fileIdx=$i

        # Execute the test suite
        ./runTestSuite.sh $numIter >> ${LOG_PREFIX}_$fileIdx.${LOG_EXT}
    ) &
```

```
    # Add block PID to list
    pidList[$i]=$!
done

# Block until all background tasks are complete - essentially performing a join
for i in `seq 1 $numInstances`; do
    echo "Waiting for instance $i (pid=${pidList[$i]}) to complete..." | tee -a
$LOG_FILE
    wait ${pidList[$i]}
done

# Capture END of test section
stop=$SECONDS
echo "Stop date/time:" $(date) | tee -a $LOG_FILE

# Combine all output to one file for easier parsing
sleep 1  # Allow time for previous file writes to complete
for i in `seq 1 $numInstances`; do
    cat ${LOG_PREFIX}_$i.${LOG_EXT} >> ${LOG_PREFIX}_combined.${LOG_EXT}
done

# Parse results and report PASS/FAIL
expectedNumPass=$numInstances
actualNumPass=$(grep PASS -c ${LOG_PREFIX}_combined.${LOG_EXT})

if [ "$actualNumPass" -eq "$expectedNumPass" ]; then
    result="PASS"
    rval=0
else
    result="FAIL"
    rval=4
fi

# Perform calculations
duration=$(( $stop - $start ))
passRate=$(bc -l <<< "scale=2; $actualNumPass/$numInstances*100")

echo ""
| tee -a $LOG_FILE
echo
"================================================================================"
| tee -a $LOG_FILE
echo "Results & Statistics"
| tee -a $LOG_FILE
echo
"================================================================================"
| tee -a $LOG_FILE
echo "Overall stress test result:       $result"
| tee -a $LOG_FILE
echo "Execution time:                   $duration [seconds]"
| tee -a $LOG_FILE
echo ""
| tee -a $LOG_FILE
echo "Expected # of passing instances:  $expectedNumPass"
| tee -a $LOG_FILE
echo "Actual # of passing instances:    $actualNumPass"
| tee -a $LOG_FILE
```

```
echo "Stress test passing rate:         $passRate%"
| tee -a $LOG_FILE
echo
"================================================================================"
| tee -a $LOG_FILE
echo "Please see the log file for the full console output: $LOG_FILE"
echo ""

# Return status code
exit $rval
```

II. stressTestApp.sh (output)

```
[mosers1@cs1 scripts]$ pwd
/home/st/mosers1/cpsc5210/buildSystem/Java-Battleship/scripts
[mosers1@cs1 scripts]$
[mosers1@cs1 scripts]$ ./stressTestApp.sh 5 10
Cleaning up stale log files...
removed ‘stressTestLog_1.txt’
removed ‘stressTestLog_2.txt’
removed ‘stressTestLog_3.txt’
removed ‘stressTestLog_4.txt’
removed ‘stressTestLog_5.txt’
removed ‘stressTestLog_combined.txt’
removed ‘stressTestLog_main.txt’

================================================================================
Stress Test Execution
================================================================================

Start date/time: Thu Jun 6 21:22:36 UTC 2019
Launching test suite instance 1 of 5...
Launching test suite instance 2 of 5...
Launching test suite instance 3 of 5...
Launching test suite instance 4 of 5...
Launching test suite instance 5 of 5...
Waiting for instance 1 (pid=21851) to complete...
Waiting for instance 2 (pid=21854) to complete...
Waiting for instance 3 (pid=21859) to complete...
Waiting for instance 4 (pid=21866) to complete...
Waiting for instance 5 (pid=21874) to complete...
Stop date/time: Thu Jun 6 21:27:49 UTC 2019

================================================================================
Results & Statistics
================================================================================
```

```
Overall stress test result:       PASS
Execution time:                   313 [seconds]

Expected # of passing instances:  5
Actual # of passing instances:    5
Stress test passing rate:         100.00%
================================================================================
Please see the log file for the full console output: stressTestLog_main.txt
```