CSC 343 Operating Systems, Fall 2020

Dr. Dale E. Parson, Assignment 1, Implementing and testing a first state machine simulation.
This assignment is due via **make turnitin** from the hitakeslo2020 directory by **11:59 PM on Thursday September 17**. There is a 10% penalty for each day it is late, and I will not accept solutions after I go over my solution in class.

EDIT 9/9/2020, From README.txt:
Q2: (7% of assignment): The standard game of high-takes-low, also
know as "war", treats ties differently than our hitakeslo2020.stm.
The standard game has each player draw 3 more cards, and compare again, whenever there is a tie.
What transition or transitions in the current solution's state
machine would you modify to implement this standard approach to a tie?
You do NOT have to write the code. You DO have to copy&paste only
the transition or transitions requiring some kind of modification
or replacement, and you MUST explain why that transition or those
transitions are the one(s) requiring modification.

I did not have that red line in there, but should have. Answer README.txt Q2 after reading the above.

The goal of this assignment is to learn how to write an introductory state machine in this semester's STM language. We will simulate a card game of high-takes-low with 2 players. There is a README.txt file with questions for you to answer after you have completed the code. **Answers in README.txt are worth 15% of this assignment, so remember that working code is not the end of the requirements**.

Perform the following steps to get my handout. You will code and test on acad.

```
cd  $HOME               # or start out in your login directory
mkdir  OpSys  # All of this semester's work goes under here, skip if you did it before.
cd  ./OpSys
cp  ~parson/OpSys/hitakeslo2020.problem.zip  hitakeslo2020.problem.zip
unzip hitakeslo2020.problem.zip
cd ./hitakeslo2020
make clean test
```

Testing fails within the handout directory as follows by hanging, **requiring a control-C to abort**.

```
$ make clean test
/bin/rm -f *.o *.class .jar core *.exe *.obj *.pyc __pycache__/*.pyc
/bin/bash -c 'chmod 666 ~parson/tmp/parson_STM*'
chmod: cannot access '/home/kutztown.edu/parson/tmp/parson_STM*': No such file or directory
make: [clean] Error 1 (ignored)
/bin/bash -c '/bin/rm -f *.out *.dif *.pyc junk parsetab.py *.vmlf hitakeslo2020_crunch.png'
/bin/bash -c '/bin/rm -f *.dot *.gif *.jpg testmachine.ck junk.* *.tmp *.log hitakeslo2020.py'
/bin/bash -c '/bin/rm -f *.crunch ~parson/tmp/parson_STM_*.log parson_STM_*.log Unsafe*.log'
/bin/bash -c '/bin/rm -f rr*.py sjf*.py fcfs*.py plotcrunch.csv *.crunch *_crunch.py *.crunch *_crunch.csv
./__pycache__/*.pyc'
COMPILING hitakeslo2020
/bin/bash -c "PYTHONPATH=/home/kutztown.edu/parson/OpSys:.:.. /usr/local/bin/python3.7
```

```
/home/kutztown.edu/parson/OpSys/state2codeV17/State2CodeParser.py hitakeslo2020.stm
hitakeslo2020.dot hitakeslo2020.py CSC343Compile CSC343Compile"

WARNING, singleton transition for scoreTwocards -> nextRound cpu(...)[@mycard > othercard@] will
discard events not matched by the guard.


WARNING, singleton transition for nextRound -> endGame cpu(...)[@len(pcb.myhand[tid]) == 0@] will
discard events not matched by the guard.


WARNING, singleton transition for waitForOtherPlay -> scoreTwocards
cpu(...)[@len(pcb.inplay[othertid]) > 0@] will discard events not matched by the guard.

INFO: Blocking function spawn is in mid-transition from thread.dealCards -> startNextTurn, so its
completion event will not trigger a state change.
COMPILING COMPLETED
SIMULATING (TESTING) hitakeslo2020
/bin/rm -f ~parson/tmp/parson_STM_*.log parson_STM_*.log hitakeslo2020.log
/bin/bash -c "PYTHONPATH=/home/kutztown.edu/parson/OpSys:.:.. STMLOGDIR=~parson/tmp time
/usr/local/bin/python3.7 hitakeslo2020.py 2 4 500 12345 3"
MSG cmd line: ['hitakeslo2020.py', '2', '4', '500', '12345', '3'], usage USAGE: python THISFILE.py
NUMCONTEXTS NUMFASTIO SIMTIME SEED|None LOGLEVEL
^C
Traceback (most recent call last):
  File "hitakeslo2020.py", line 711, in <module>
    main()
  File "hitakeslo2020.py", line 656, in main
    scheduler.__run__()
  File "/home/kutztown.edu/parson/OpSys/state2codeV17/CSC343Sim.py", line 145, in __run__
    waitingObject.__generator__.__next__()  # run() the model
  File "hitakeslo2020.py", line 302, in run
    elif len(locals.keys()) != self.__localcount__:
KeyboardInterrupt
15.89user 0.02system 0:15.99elapsed 99%CPU (0avgtext+0avgdata 10172maxresident)k
0inputs+24outputs (0major+5130minor)pagefaults 0swaps
make: *** [test_hitakeslo2020] Error 1
```

All of the detailed instructions for your code additions appear in **STUDENT** comments in source file **hitakeslo2020.stm**, which is one of the two files you will change. The other file is README.txt. We will go over your project requirements in class on 9/2.

If you get an error message at run-time that gives an index into __codeTable__ like this:

```
Traceback (most recent call last):
  File "hitakeslo2020.py", line 758, in <module>
    main()
  File "hitakeslo2020.py", line 694, in main
    scheduler.__run__()
  File "/home/kutztown.edu/parson/OpSys/state2codeV17/CSC343Sim.py", line 145, in __run__
    waitingObject.__generator__.__next__()  # run() the model
  File "hitakeslo2020.py", line 457, in run
    if eval(__codeTable__[39],globals,locals):
  File "nofile", line 1, in <module>
NameError: name 'ln' is not defined
```

Just run this decode.py command with that index to see the original source code.

```
$ python decode.py hitakeslo2020.py 39

__codeTable__[39] = compile('ln(pcb.inplay[othertid]) == 0','nofile','eval'),
```

A successful test run appears as follows.

```
$ make clean test
/bin/rm -f *.o *.class .jar core *.exe *.obj *.pyc __pycache__/*.pyc
/bin/bash -c 'chmod 666 ~parson/tmp/parson_STM*'
/bin/bash -c '/bin/rm -f *.out *.dif *.pyc junk parsetab.py *.vmlf hitakeslo2020_crunch.png'
/bin/bash -c '/bin/rm -f *.dot *.gif *.jpg testmachine.ck junk.* *.tmp *.log hitakeslo2020.py'
/bin/bash -c '/bin/rm -f *.crunch ~parson/tmp/parson_STM_*.log parson_STM_*.log Unsafe*.log'
/bin/bash -c '/bin/rm -f rr*.py sjf*.py fcfs*.py plotcrunch.csv *.crunch *_crunch.py *.crunch *_crunch.csv
./__pycache__/*.pyc'
COMPILING hitakeslo2020
/bin/bash -c "PYTHONPATH=/home/kutztown.edu/parson/OpSys:.:.. /usr/local/bin/python3.7
/home/kutztown.edu/parson/OpSys/state2codeV17/State2CodeParser.py hitakeslo2020.stm
hitakeslo2020.dot hitakeslo2020.py CSC343Compile CSC343Compile"
INFO: Blocking function spawn is in mid-transition from thread.dealCards -> startNextTurn, so its
completion event will not trigger a state change.
COMPILING COMPLETED
SIMULATING (TESTING) hitakeslo2020
/bin/rm -f ~parson/tmp/parson_STM_*.log parson_STM_*.log hitakeslo2020.log
/bin/bash -c "PYTHONPATH=/home/kutztown.edu/parson/OpSys:.:.. STMLOGDIR=~parson/tmp time
/usr/local/bin/python3.7 hitakeslo2020.py 2 4 500 12345 3"
MSG cmd line: ['hitakeslo2020.py', '2', '4', '500', '12345', '3'], usage USAGE: python THISFILE.py
NUMCONTEXTS NUMFASTIO SIMTIME SEED|None LOGLEVEL

Scheduler exiting at time 161 within time limit 500, simulation has finished.
0.06user 0.02system 0:00.16elapsed 53%CPU (0avgtext+0avgdata 10196maxresident)k
0inputs+120outputs (0major+5183minor)pagefaults 0swaps
/bin/bash -c 'chmod 666 ~parson/tmp/parson_STM*'
/bin/bash -c "PYTHONPATH=/home/kutztown.edu/parson/OpSys:.:.. /usr/local/bin/python3.7
crunchlog.py hitakeslo2020.log"

DIFFing hitakeslo2020_crunch.py hitakeslo2020_crunch.ref
OK: SUM_dealCards_thread_0_process_0 at 20.0% tolerance.
OK: SUM_dealCards_thread_1_process_0 at 20.0% tolerance.
OK: SUM_startNextTurn_thread_0_process_0 at 20.0% tolerance.
OK: SUM_startNextTurn_thread_1_process_0 at 20.0% tolerance.

# STUDENT, COMMENT OUT NEXT LINE TO SEE THE LOG FILE.
# bash -c '/bin/rm -f ~parson/tmp/parson_STM_*.log parson_STM_*.log hitakeslo2020*.log'
```

```
grep "GAME OVER, THREAD ID" hitakeslo2020.log | sort > hitakeslo2020.out
diff hitakeslo2020.out hitakeslo2020.ref
COMPLETED (OK) SIMULATING (TESTING) hitakeslo2020
```

You can see the successful extracted messages from your program by doing this.

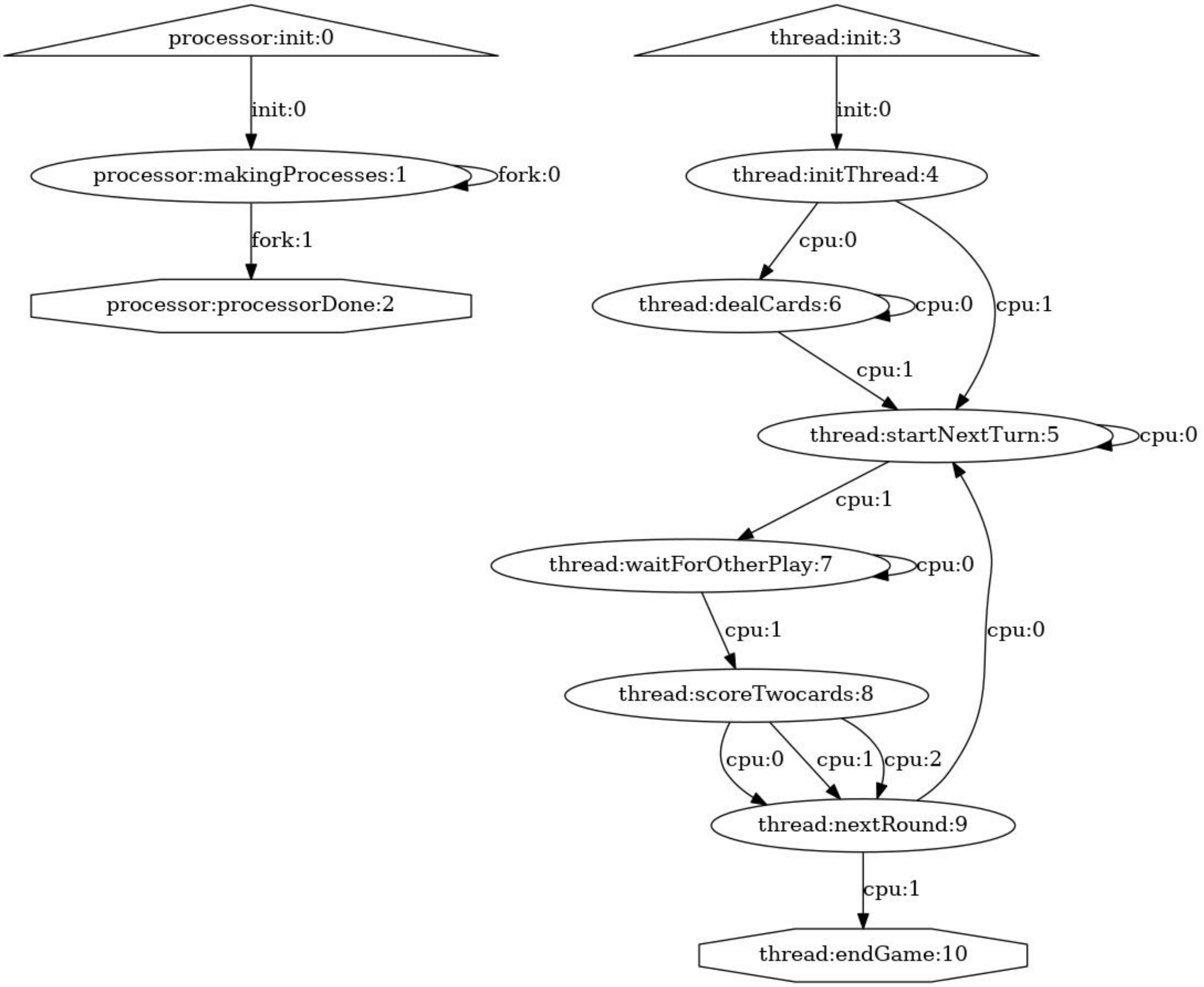
```
$ cat hitakeslo2020.out
000000000160,MSG,thread 0 process 0,GAME OVER, THREAD ID 0 has 35 CARDS, I AM WINNER
000000000161,MSG,thread 1 process 0,GAME OVER, THREAD ID 1 has 17 CARDS, I AM LOSER
```

Any time a COMPILE succeeds, you can look at the graph for your state machine by running **make graphs** and then inspecting https://kuvapcsitrd01.kutztown.edu/~STUDENT/hitakeslo2020.jpg, where STUDENT is your login ID. If you can't get at it with a browser this way, use WinSCP or FileZilla to copy the JPEG file from your project directory to your local machine. Below is the final, correct graph.

Once **make clean test** passes, **ANSWER THE QUESTIONS IN FILE README.txt** included in this project directory. Follow all instructions in README.txt.

Finally, turn it in by entering **make turnitin** and following the prompt. We do not use the turnin script in this course; instead **make turnitin** turns in the project; it prompts you for a carriage return (Enter) to complete its work.

I will distribute grades via email before the next class after the due date.


processor:init:0
init:0
processor:makingProcesses:1
fork:0
fork:1
processor:processorDone:2
thread:init:3
init:0
thread:initThread:4
cpu:0
cpu:1
thread:dealCards:6
cpu:0
cpu:1
thread:startNextTurn:5
cpu:0
cpu:1
thread:waitForOtherPlay:7
cpu:0
cpu:1
cpu:0
thread:scoreTwocards:8
cpu:0
cpu:1
cpu:2
thread:nextRound:9
cpu:1
thread:endGame:10