Hwk 8b - TM Design

100/100 Points

4/24/2022

Attempt 1

REVIEW FEEDBACK Attempt 1 Score: **100/100**
4/24/2022

View Feedback

Unlimited Attempts Allowed
5/20/2022

Details

You and up to two other students are challenged with solving one of the three problems below via the **STEM** java software (i.e., jar file) (posted under **Modules** in Canvas). One member of your group (or yourself if you are working alone) must submit the appropriate **txt** file (generated by the applet) along with a **narrative summary** (as a docx, txt , or pdf file) of the following: (1) names of collaborators and the roles for each in the design of the TM, (2) a problem description/summary for the method of solution used, (3) a functional description of the TM in terms of what each state means (e.g., an annotated state diagram formatted as a pdf, jpg, or png can be included), (4) specification of how to start the TM (i.e., what is the start state) and what the input (on the tape) should be (your GTA will be executing your machine so documentation of how to start it is important), and (5) examples of tape outputs) that represent accepted and rejected inputs (if appropriate). The **STEM** software loads/stores TM's from/to files using a special ASCII file format.

Important: This Canvas assignment has been designated as a **group assignment**. Use the Piazza posting "Group Membership for Hwk 8b (TM Design)" to indicate your group's membership. **Only one member** of your group should submit the **documentation file** and appropriate (STEM.jar generated) input **txt** file to Canvas.

Java issues: The code in the **STEM**.jar requires **JavaFx** libraries which may not be available in recent Java Runtime Environments (JREs). A procedure to run the jar file on any EECS lab machine (**tesla** or **hydra**) is available below. The procedure below uses JRE 14. Warning: you should use a remote login such as **ssh -X [-Y] hydra1.eecs.utk.edu** to enable **X11** forwarding to your local machine. The **-Y** option enables *trusted* X11 forwarding. You only need to use the procedure below if you are unable to run Java code on your local machine.

VNC: A faster connection is now available on the VNC cluster in the EECS Department. Using the RealVNC® VNC Viewer app, a secure (encrypted) desktop connection can be made between one of the machines in the cluster (say **vnc1.eecs.utk.edu:99**) to your local machine. The link below will provide information on how to download the VNC app for your local machine and establish a connection:

https://help.eecs.utk.edu/knowledge-base/general/remote-access/vnc
(https://help.eecs.utk.edu/knowledge-base/general/remote-access/vnc)

RealVNC access is now enabled on the VNC lab systems in the EECS Department, the legacy Ada systems, and the new VLSI systems (see **https://help.eecs.utk.edu/systems/start#virtual-labs (https://help.eecs.utk.edu/systems/start#virtual-labs)**)

Simply execute the following command: `javafx STEM.jar`

To execute the latest (Version 2.0) STEM.jar file, you'll need Java 9.0 or higher JRE which can be downloaded using the link below:

(https://www.oracle.com/java/technologies/javase/javase9-archive-downloads.html) https://www.oracle.com/java/technologies/javase/javase9-archive-downloads.html (https://www.oracle.com/java/technologies/javase/javase9-archive-downloads.html)

The **STEM.jar** file and source code along with install/execute instructions are available through the **Turing Machine Software, Articles, and Examples (https://utk.instructure.com/courses/139839/modules/389210)** link under the **Modules** tab on the Canvas menu (on the left). Students are encouraged to install the **STEM.jar** on their own computer and run the simulator locally.

New **Windows** bundle for STEM.jar

John McElroy has graciously provided a self-extracting bundle that runs STEM.jar on Windows platforms with no hassle. The **exe** file below is a WinRAR self-extracting RAR archive that contains STEM.jar and the Adoptium Java runtime (version jdk-17.0.2+8-jre). It extracts to a temp folder and runs "javaw -jar STEM.jar"

STEM-2.0.1-selfextract.exe (https://utk.instructure.com/courses/139839/files/12225692?wrap=1) (https://utk.instructure.com/courses/139839/files/12225692/download?download_frd=1)

New Mac M1 jar file for STEM

The STEM software makes use of the **JavaFx** library, which has somewhat recently been modified for m1 Macs (built with AMD rather than Intel processors). John Malloy has graciously re-compiled the java source with a more recent JavaFx library that is compatible for m1 Macs:

STEM_v2.0.2.jar (https://utk.instructure.com/courses/139839/files/12303305?wrap=1) (https://utk.instructure.com/courses/139839/files/12303305/download?download_frd=1)

Grading Rubric:

Points	Criteria
20	Problem description/summary

20	TM functional design
20	Input specification
30	Correct execution (halting)
10	Sample outputs
100	**Max points**

Examples of past COSC 312 TMs for this assignment:

1. Binary to unary conversion.
2. Integer division for an input unary number (provide quotient and remainder).
3. Find the longest shared prefix of two input binary numbers of the same length.
4. Compute the 2's complement of an input binary number.
5. Compute the square of an input unary number.
6. Produce the "mirror" of an input string - given any string of x's, y's, and z's overwrite the input with the reverse of the string.
7. Compute the bitwise AND of two input binary numbers having the same number of binary digits.
8. Convert military 24-hour time (in binary) to regular 12-hour time (in binary).
9. Compute the XOR parity string for two input binary strings of the same length (e.g. the XOR parity string for inputs 001 and 101 is 100).
10. Compute the fibonacci sequence up to a specific number of elements.
11. Bubble sort binary integers.
12. Recognition and categorization of emoticons.
13. String reversal parser.
14. Logarithm parser and solver.
15. Substring matching, counting, and printing for strings of equal length.
16. Depth-First-Search for finding cycles in a graph.
17. Sudoku solver on a 4-by-4 grid.
18. Arithmetic expression checker.
19. Conversion of phrases from the UK to US (English) language.
20. Random number generator.
21. DNA translation to polypeptides.
22. Prime number checker.
23. Character frequency checker for a shift cipher.
24. Selection sort for arbitrary count of binary numbers.
25. Determine if a 2 by 2 matrix (containing integers in binary form) is invertible.
26. Two-player Nim subtraction game.
27. Interpreter for Spanish phrases based on pronoun-verb agreement.

28. Tic-Tac-Toe game with TM responding to each play made by user.
29. Morse code encoder and decoder.
30. Selection sort for unary numbers.
31. C-style post- and pre-increment parser with error codes.
32. Baseball scoring decoder (up to the next out).
33. Calculator with addition/subtraction/multiplication/division for two unary numbers.

View Rubric

Select Grader

Michael W Berry

TM Design (Group Assignment) (1)			
Criteria	**Ratings**		**Pts**
Problem Description/Summary	**20 pts** **Full Marks**	**0 pts** **No Marks**	20 / 20 pts
TM Functional Design (State Diagram)	**20 pts** **Full Marks**	**0 pts** **No Marks**	20 / 20 pts
Input Specification	**20 pts** **Full Marks**	**0 pts** **No Marks**	20 / 20 pts
Correct Execution (Halting)	**30 pts** **Full Marks**	**0 pts** **No Marks**	30 / 30 pts
Creativity/Novelty	**10 pts** **Full Marks**	**0 pts** **No Marks**	10 / 10 pts
	Comments This is an excellent demo of the SAT problem embedded within video games. I really like this machine!		
			Total Points: 100