System Validation 2022
Homework Part 2 – Runtime Monitoring and Software Analysis

Deadline: **23:59 CET, Friday Oct. 14, 2022**

- It is recommended to make the assignment in pairs.
- While working together, be very careful with public repositories (if you decide to collaborate via e.g. Git), as we will check for fraud!
- A total of 100 points can be earned.

All solutions should be uploaded via Canvas. You should hand in a single ZIP file with:

- The report in PDF format. Do not forget to include your names and student numbers in the report. The report should describe all your solutions.
- Your Larva models, the corresponding program code and any scripts used for execution/compilation of Larva.
- Your modified cartesianTrees.c. Clearly indicate which modifications were introduced for which (sub-)question.

Good luck!

Runtime Monitoring (50 pts)

This exercise comes with a Java implementation of a grade administration system. Students, courses and grades can be added to the system through use of the textual interface. Before a student's grade can be registered for a course, (s)he is expected to be enrolled in this course first. The amount of credits that a student has earned is automatically calculated based on whether (s)he got a passing grade (6 or higher) in the courses for which (s)he has been enrolled.

Instructions on how to run this implementation are included in the README. The Java implementation may or may not be correct.

Exercises

Construct a LARVA runtime monitor that captures the following behaviours. Run the system and your LARVA monitor to check if the application behaves as it should. If not, point out where it violates a specific property, and give the event traces produced by LARVA that show these violations.

i. All registered grades are between 1 and 10 (including 1 and 10 themselves). **(15 pts)**

ii. A student cannot have less than zero credits. **(15 pts)**

iii. When data is saved to the database, the network can cause delays. Check that when a new grade is added to the database, the network delay is less than 5 seconds. *Use the built-in Larva constructs such as Clocks to construct this monitor.* **(20 pts)**

Bounded Software Analysis (50 pts)

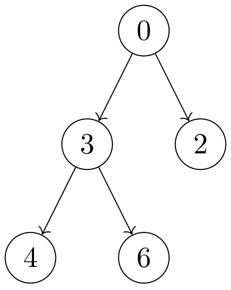

Figure 1: Tree deduced from the array [4, 3, 6, 0, 2]

You are given the program CARTESIANTREES.C which constructs a tree from the numbers in a given array (which represent the inorder traversal of the tree). See Figure 1 for an example.

The following questions are all regarding the cartesianTrees.c program:

i. Verify that the method MINELEMENTINDEX returns the index of the minimal element between the indices START and END using assertions. If you need any assumptions to be able to prove this, explain why these assumptions are necessary. **(15 pts)**

ii. Verify that the tree constructed by the method CONSTRUCTTREE is well-formed using assertions, i.e. the parent element is always strictly less than its child elements. If you need any **(15 pts)**

assumptions to be able to prove this, explain why these assumptions are necessary. *Hint: you only need to verify that the parent element is less if there exists a child element.*

iii. Currently the array size for which the program is verified is set to 4. Modify the program (using CIVL constructs) such that the program is verified for arrays of size 0 up until a variable MAX. **(10 pts)**

iv. Up until what bound MAX for the array size are you able to verify the program within 1 minute? **(10 pts)**