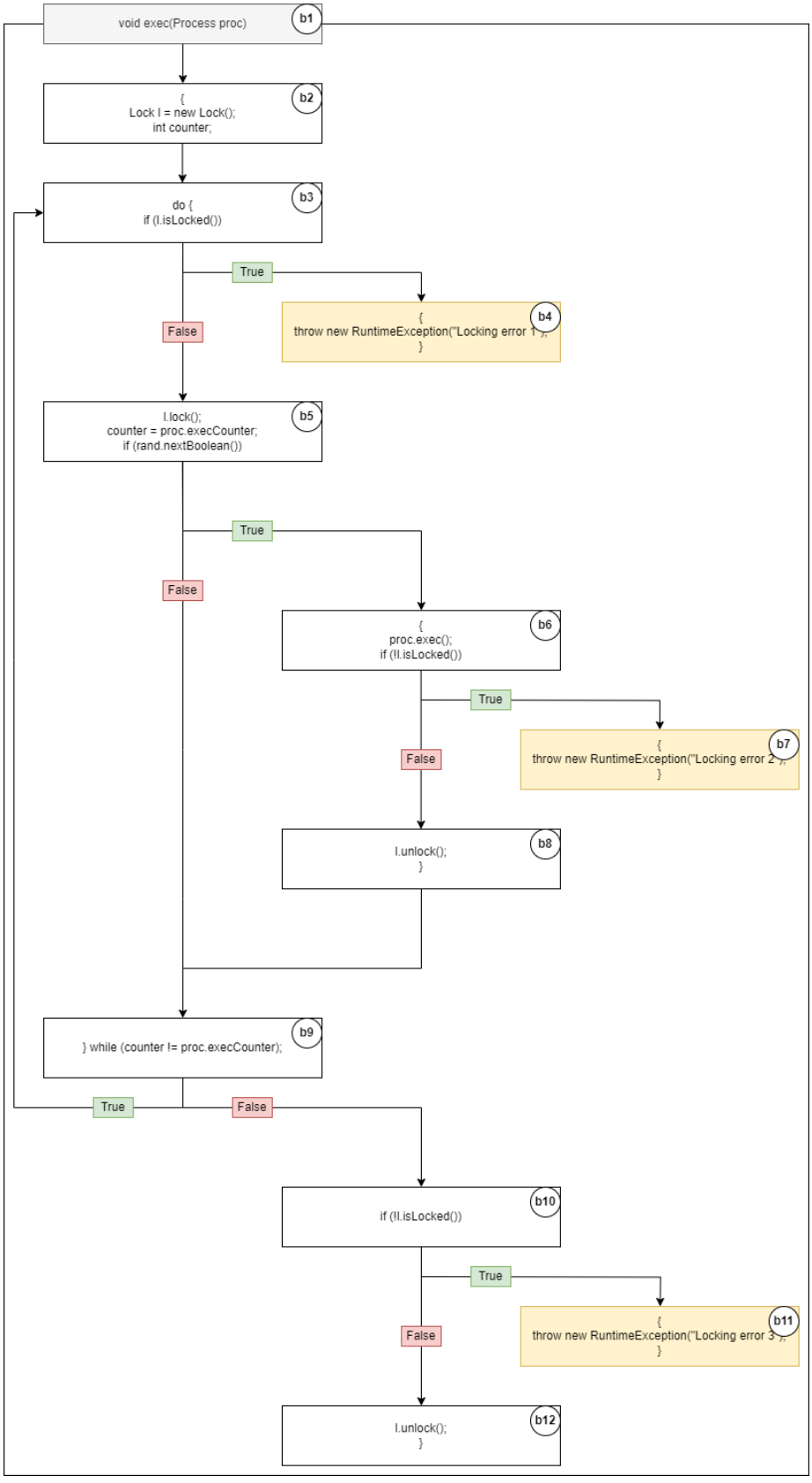
Piacente Cristian 866020

Homework H1 – CFG

Software Quality, Academic Year 2023-2024, University of Milan – Bicocca

Task 1) Draw the control flow graph (CFG) of the method CriticalProcessExecManager.exec(Process).



Question 1) Is there any CFG path that leads to throwing the exceptions "Locking error 1", "Locking error 2" and "Locking error 3"? (Show a CFG path for each of the three exceptions).

Yes, here is an example of a path (not necessarily feasible) for each of the three exceptions:

- "Locking error 1": `b1-b2-b3-b4`
- "Locking error 2": `b1-b2-b3-b5-b6-b7`
- "Locking error 3": `b1-b2-b3-b5-b9-b10-b11`

Question 2) Considering the CFG paths that you indicated as answers to the previous question, do they correspond to some program execution in which we can truly observe those exceptions, or are they false alarms?

To answer this question, we must analyze the program execution state carefully.

First of all, from the given class Lock, we can see that when a Lock object gets instantiated it is **not locked**.
This means that the first "True" branch (which would execute the basic block b4) is **infeasible** (there is no possible input that can make the execution cover it), and so the exception with message "Locking error 1" will never be thrown: **the first CFG path leads to a false alarm**.

Let's continue analyzing the paths.
If we look at the code that is executed in the second CFG path, **the lock gets locked**, then the variable counter gets set to proc.execCounter and there are two if statements.
The first if statement checks if a random boolean equals to true: this can be possible.
The second one checks **if**, after invoking the exec() method on the proc object (passed as an argument), which **increments execCounter**, the lock is **not locked**: we must remember the lock was locked and also the lock has a local scope and cannot be accessed by the proc object. This means that the condition !l.isLocked() always evaluates to false, so the "True" branch, which would execute the basic block b7, is **infeasible** and the exception with message "Locking error 2" cannot be thrown for any input: **the second CFG path leads to a false alarm**.

Finally, if we have a look at the third CFG path, after locking the lock (basic block b5) and setting the counter variable to proc.execCounter, a false boolean must be randomly generated (which is possible), then the loop condition "counter != proc.execCounter" must evaluate to false, which is correct since in this CFG path the exec() method doesn't get invoked, so the value of execCounter doesn't change and it equals to the counter variable.
The problem occurs when considering the last if statement: in order to throw the exception with message "Locking error 3", the lock must not be locked, but the lock is locked (since the basic block b8 wasn't executed in this path): the "True" branch which would execute the basic block b11 is **infeasible** and so **the third CFG path also leads to a false alarm**.

EXTRA: about the third exception, if instead of considering b5-b9 in the path we used b5-b6-b8-b9 it would still produce a false alarm because in this case proc.exec() would increment execCounter, it follows that the counter variable would not be equal to proc.execCounter, which would result in another loop iteration (counter != proc.execCounter evaluates to true) and so we would have b9-b3 instead of b9-b10.

All the three exceptions are false alarms: there's no feasible CFG path that can lead to them.