Dr. Annette Bieniusa
Albert Schimpf, M. Sc.

TU Kaiserslautern
Fachbereich Informatik
AG Programmiersprachen

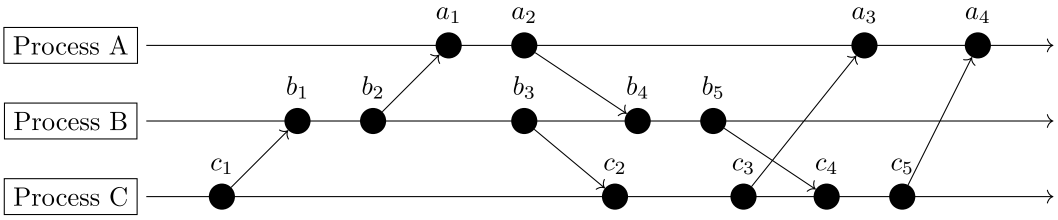
Exercise 3: Programming Distributed Systems (Summer 2020)

Submission

- You need a team and a Gitlab repository for this exercise sheet.
- In your Git repository, create a branch for this exercise sheet (for example with `git checkout -b ex3`)
- Create a folder named “ex3” in your repository and add your solutions to this folder.
- Create a merge request in Gitlab and assign Albert Schimpf as assignee. If you do not want to get feedback on your solution, you can merge it by yourself.
- Test your submission with the provided test cases. Feel free to add more tests, but do not change the existing test cases.

1 Logical clocks

Calculate the Lamport clock and vector clock timestamps $t(e)$ for all events in the following execution.



2 Time and causality

Give an example execution that shows that for two timestamps from Lamport clocks, $C(e_1) < C(e_2)$ does not imply that $e_1 \to e_2$.

Prove that the other direction is valid: If $e_1 \to e_2$, then $t(e_1) < t(e_2)$.

3 Implementing Vectorclocks

A vector clock is a mapping from processes to positive integers[1]. Implement a module named `vectorclock` with the following functions:

- `new()` creates a new vector clock, where all processes have value 0.
- `increment(VC, P)` increments the entry of process `P` by 1.
- `get(VC, P)` returns the value for process `P`.
- `leq(VC1, VC2)` checks, whether `VC1` is less than or equal to `VC2`. This is the case, iff $\forall P.\, get(VC_1, P) \leq get(VC_2, P)$.

[1] In the literature it is often assumed that processes are numbered which allows to write down clocks like $[4, 7, 3]$ or $\begin{pmatrix}4\\7\\3\end{pmatrix}$ instead of the longer $\{p_1 \mapsto 4, p_2 \mapsto 7, p_3 \mapsto 3\}$. However, in this exercise we do not assume that the number of processes is known and arbitrary terms can be used as process names.

- `merge(VC1, VC2)` merges two vector clocks by computing their least upper bound (the smallest vector clock `V`, such that $VC_1 \leq V$ and $VC_2 \leq V$).