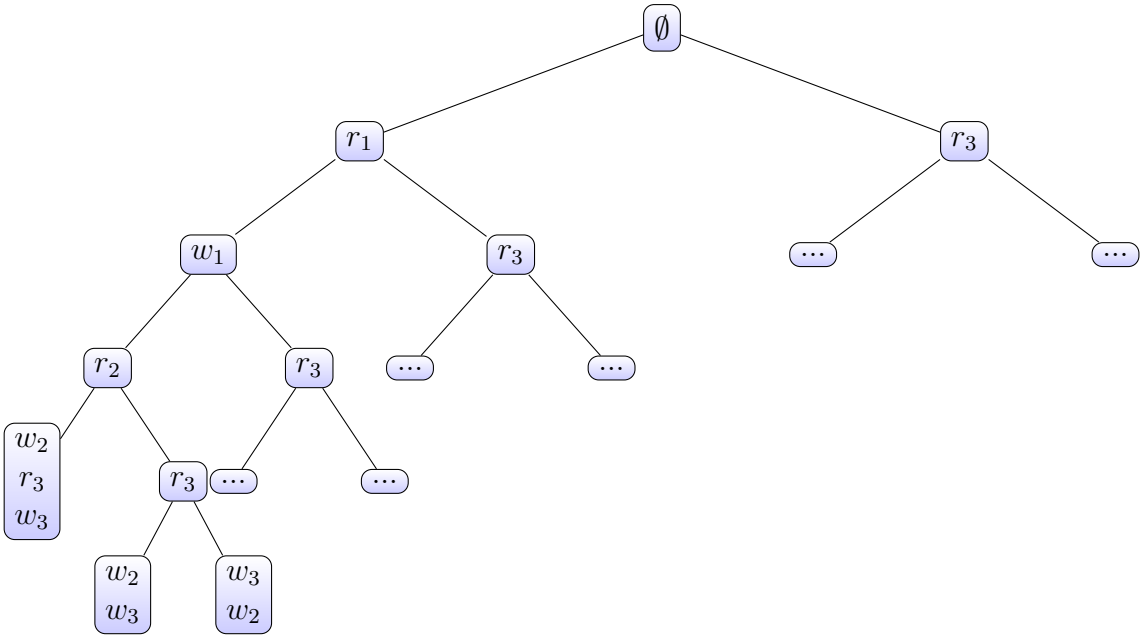
Concurrent Programming - Sheet 1

1. We want individual tabs to run in parallel. For example playing videos in multiple tabs, or receiving messages in a different tab than the one we are working in. The individual tabs do not actually interact with each other (or not frequently), so parallelisation would not be hard to achieve. Moreover writing each tab as a standalone thread might be desirable for encapsulation purposes.

2. We choose m atomic actions to be of P out of the total $m+n$, resulting in $\binom{m+n}{m}$.

3. Let p be consisted of atomic operations r_1, w_1, r_2, w_2 and q of r_3, w_3.



The possible results are 3,4,5,6,7.

4. It might but might not. Depending on the interleaving, the two values might go past each other inbetween two loop guard evaluations.

5. Based on interleaving, two processes might first evaluate the canDebit, both passing, and then they will both charge the account possibly making it go into negatives. If the methods are performed atomically, a solution would be to make canDebit a part of the debit function, possibly throwing an exception we can handle appropriately in case of insufficient funds.

6. Programming

7. Programming

8. We can create one task per element in a, the workers job is to check if the according element is contained in the array b and send a 1 to the collector if it is present, 0 otherwise (it contains no repetitions). One job can alternatively contain more than one element.
   There can either be one worker per element or we can use similar pattern to the one in 7. It is important for there to be a constant number of elements per task to prevent quadratic time complexity.