Exercise: Multi-Party Approval

Exercise Description

- Tom needs to submit a template for a new marketing campaign which needs to be approved from multiple sources. For example
  - Proof reading by Sarah
  - Check for copyrighted material by Anne
  - Layout and user experience related comments by Eddy
  - …
  - …
- Each person should be able to either directly approve his/her part or reply with some comments that need to be addressed before getting the final approval.
- An approver can only submit his/her own approval. Approval delegation is not allowed.
- The list of approvers is not fix. Approvers can be added and removed up to the point where the approval process begins. The initial approver's feedback is "undefined".
- Once the approval process starts, no more approvers can be added. All currently added approvers should comment the process with a status ("approved", "needs work") and a text comment.
- If any approver replies with a "needs work" status, the process temporarily stops. Tom signals that approver to re-review his work as soon as he updates it (by setting the approver's comment back to "undefined").
- An approver can update his/her review at any time, but only upwards (this means, once an approver has approved, it cannot be changed).
- When all approvers have approved, the process ends.
- **Bonus**: Correlation of approvals: Eddie needs to also submit feedback but cannot do so before Sarah has reviewed and commented. Tom should be able to define such correlations when adding new approvers to the process.

Technical description / Implementation hints:

Initialize a process where a multi-approval workflow is needed. The initiator gives a name to the process (optional - this can be done via the constructor of the contract).

Each one of the approvers needs to react to the invitation by replying either with a status "needs work" or "approved". An optional comment can be submitted.

Method Calls:

There is a method to add an approver (name and address-hash). Alternatively, a list of approvers can be defined and submitted at once. The list of approvers can be updated up to a specific point (defined by a separate method call). Only the initiator can alter the approver list.

Another method is needed to define the time where the input submission ends. Then, the list of approvers can no longer be updated. The initiator only should be able to call it.

One more method should be available to an approver only, to give feedback. This method can only be called by an approver himself. Parameters here are a status and a comment. This method should check the following:

- Is the caller in the approvers list? If not, exit.
- Has the caller approved already? If yes exit.

Then the method should update the status and the comment for the current caller.

A Get-Feedback method should be available to the initiator only, to check if there are any "needs work" statuses and by whom. This method should end the process, if all approvers have submitted a status "approved".

A Request-For-Feedback method should be available to the initiator only, to update the status message of an approver from "needs work" back to "undefined", thus informing him/her to submit feedback again. No other feedback can be altered.

Bonus: A list of addresses can be passed to the approver to define correlations between them. A method should check if the current approver can submit feedback by querying if those related approvers have already submitted feedback of their own.