Bank smart contract task

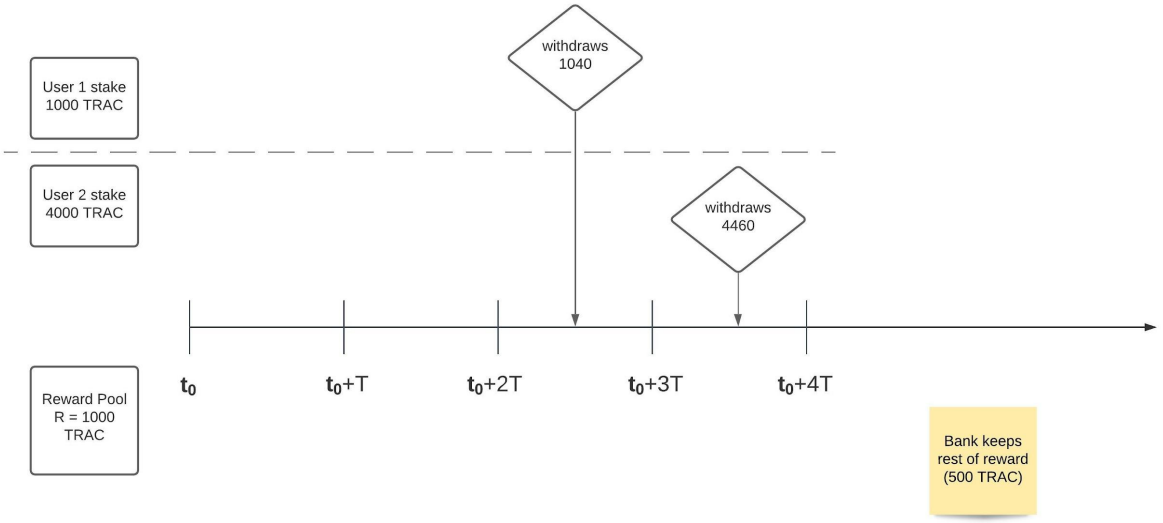
Your task is to create a **bank smart contract** which will enable anyone to deposit an amount X of XYZ ERC20 tokens to their savings (staking) account. The bank smart contract also contains an additional token reward pool of R XYZ tokens, deposited to the contract by the contract owner (bank owner) at contract deployment. At deployment the bank owner sets a time period constant T, to be used for reward calculation.

Contract dynamics (example illustrated below):

- The smart contract is deployed at $\mathbf{t_0}$
- The reward pool R is split into 3 subpools
    - **R1 = 20% of R**, available after 2T has passed since contract deployment
    - **R2 = 30% of R**, available after 3T has passed since contract deployment
    - **R3 = 50% of R**, available after 4T has passed since contract deployment
- **Deposit period:** During the first period of T time the users can deposit tokens. After T has passed, no more deposits are allowed.
- **Lock period:** From moment $\mathbf{t_0}$+T to $\mathbf{t_0}$+2T, users **cannot** withdraw their tokens (If the user tries to remove tokens before T time has elapsed since they have deposited, the transaction should fail).
- **Withdraw periods:** After T2 has passed since contract deployment, the users can withdraw their tokens. However, the longer they wait, the bigger the reward they get
    - If a user withdraws tokens during the period $\mathbf{t_0}$+2T to $\mathbf{t_0}$+3T, they collect a proportional amount of the reward pool R1, according to the ratio of the number of tokens they have staked compared to the total number of tokens staked on the contract (by all users).
    - If a user withdraws tokens during the period $\mathbf{t_0}$+3T to $\mathbf{t_0}$+4T, they collect a proportional amount of the remaining reward pool R1 and R2, according to the proportion of the number of tokens they have staked compared to the total number of tokens staked on the contract (by all users)
    - If the user withdraws tokens after 4T has passed since contract deployment, they can receive the full reward of R (R1+R2+R3) proportionally to their ratio of tokens in the total pool
    - If no user waits for the last period (for 4T to pass), the remaining tokens on the contract can be withdrawn by the bank (contract owner). In no other situation can the bank owner remove tokens from the contract.

Example:

- **User 1** stakes S1 = 1000 XYZ during deposit period
- **User 2** stakes S2 = 4000 XYZ during deposit period
- Reward pool R = 1000 XYZ (R1 = 200XYZ, R2 = 300 XYZ, R3 = 500 XYZ)
- User 1 withdraws their tokens in period t_0+2T to t_0+3T
  - User 1 should receive their initial deposit of 1000 XYZ and
  - A reward of 40 XYZ, proportional to their amount of tokens in the pool
- User 2 is impatient and withdraws their tokens in period t_0+3T to t_0+4T
  - User 2 should receive their their initial deposit of 4000 XYZ and
  - A reward of 460 XYZ, which is 100% of the remaining R1 tokens, and 100% of the remaining R2 tokens (as user 2 tokens are 100% of the remaining staked tokens in the bank)
- After 4T has passed, the bank can withdraw the remaining reward (500XYZ) since no user has any more deposits to withdraw



Your task is to

- Create a smart contract according to the specifications above
- Create a test suite for the above functionalities

Please include instructions on how to run your project together with the code.

You get bonus points for

- Reusing existing (standard) smart contracts
- Deploying the contract on Ethereum Rinkeby testnet
- Providing truffle deploy scripts for Rinkeby
- For having the ERC20 token be Rinkeby ATRAC (we can send you some)
- Providing a graphical interface for the contract

Push the project to a private Github (not public) repository and add collaborators for review - usernames: *branarakic, djordjekovac, kotlarmilos & marijakrivosic*

Have any questions? Feel free to ask us

Take your time and happy coding!