What is the primary purpose of the smart contract in the lab manual?
To create a bank account that allows deposit, withdrawal, and balance-checking functions.

What is Metamask, and why is it used in this assignment?
Metamask is a digital wallet for managing cryptocurrency and interacting with Ethereum blockchain networks, facilitating transactions in the smart contract.

What knowledge prerequisites are needed for this lab?
Basic understanding of cryptocurrency, distributed computing concepts, and blockchain mechanics.

What is 'msg.sender' in Solidity, and how is it used?
`msg.sender` is the address of the account initiating a transaction; it is used for authentication in smart contracts.

Why is it necessary to check the balance before allowing a withdrawal?
To ensure the sender has sufficient funds to prevent unauthorized overdrafts.

What is the role of the 'require' statement in Solidity?
`require` enforces conditions; if conditions aren't met, it reverts transactions and aborts changes.

Explain the concept of 'mapping' in the context of this smart contract.
`mapping` is used to store key-value pairs, associating account addresses with balances.

Why is it important to update the balance before transferring funds in the withdraw function?
Updating first prevents reentrancy attacks by ensuring accurate balance before any further interactions.

What happens if a withdrawal amount exceeds the balance in this contract?
The transaction fails because the `require` statement will halt execution without altering the state.

How does the contract manage ether deposits?
It uses a function that checks the amount, adds it to the sender's balance, and updates the contract's storage.

Why is Solidity's version specified in the contract?
To ensure compatibility and avoid errors due to syntax or functional changes in different Solidity versions.

What is a 'reentrancy bug' and how does it affect smart contracts?
A reentrancy bug allows repeated calls to a contract before the first call is completed, potentially draining funds.

Describe the use of `payable` in Solidity functions.
`payable` functions can accept ether; without it, a function cannot receive funds.

What would happen if the `withdraw` function did not adjust the balance before the transfer?
It could create a vulnerability where a malicious contract could exploit reentrancy to withdraw repeatedly.

Why is 'balanceOf' mapping used as 'public' in this contract?
To allow anyone to view the balance of accounts stored in the contract.

How does the contract enforce that only the owner can withdraw funds?
It compares `msg.sender` to the owner’s address before allowing withdrawals.

What value does `balanceOf` mapping return for addresses that haven't made any deposits?
It returns `0` by default for any undeclared mapping entries.

How can a user check their balance using this contract?
By calling the `getBalance` function, which retrieves the balance for the account.

What might happen if an incorrect address format is used in a function call?
The contract will fail to execute properly due to an invalid address input.

What could be a real-world application of this smart contract example?
It could serve as a basic structure for decentralized banking or wallet services on the blockchain.