Name: Darshan Bele
Roll No: 14110
Class: BE – A – A1

Practical 3

Program:

```
// SPDX-License-Identifier: MIT
// Specifies the license under which the code is released.  pragma
solidity ^0.8.18;

/**
* @title BankAccount
* @dev A simple smart contract that allows a user to deposit, withdraw,  * and check their
  balance in Ether.
 */
contract BankAccount {

    // State variable to store the balance of the account in Wei.     //
'public' keyword automatically creates a getter function for it.
uint256 public balance;

    // The address of the person who deploys the contract.
address public owner;

    // Event to log deposits for a transparent transaction history.
event Deposit(address indexed account, uint amount);     // Event
to log withdrawals for a transparent transaction history.
event Withdraw(address indexed account, uint amount);

    /**
* @dev The constructor is called only once when the contract is deployed.      * It sets the
  deployer of the contract as the 'owner'.
     */
constructor() {
owner = msg.sender;
    }

    /**
* @dev Allows anyone to deposit Ether into this contract.
* 'payable' is a special keyword that allows a function to receive Ether.
     */
    function deposit() public payable {
        // 'msg.value' holds the amount of Ether sent with the transaction.
balance += msg.value;
        // Log the deposit event to the blockchain.
emit Deposit(msg.sender, msg.value);
    }

    /**
```

```
* @dev Allows only the owner to withdraw a specific amount of Ether.
* @param _amount The amount of Ether (in Wei) to withdraw.
     */
    function withdraw(uint256 _amount) public {
        // 1. Check if the person calling the function is the owner.
        require(msg.sender == owner, "Only the owner can withdraw.");
        // 2. Check if the requested withdrawal amount is available in the balance.
        require(_amount <= balance, "Insu icient balance for withdrawal.");

        // 3. Subtract the amount from the balance *before* sending Ether.
        // This is a crucial security step to prevent re-entrancy attacks.
balance -= _amount;

        // 4. Transfer the Ether to the owner's address.
        payable(msg.sender).transfer(_amount);

        // 5. Log the withdrawal event.
        emit Withdraw(msg.sender, _amount);
    }

    /**
* @dev A function to view the contract's current balance.
* Note: While the 'balance' variable is public, this function provides a       * clear, explicit way for
  users or other contracts to query it.
* 'view' means it doesn't change the state of the blockchain and doesn't cost gas to call.       *
  @return The current balance in Wei.
     */
    function getBalance() public view returns (uint256) {
return balance;
    }
}
```