Assignment-3

1) For input x=5, the iterative factorial() function uses 2350 gas while the recursive factRec() uses 2756 gas. The iterative approach is more gas-efficient because recursion creates multiple function calls on the stack, each with its own overhead, while iteration maintains state in a single function context.

2) Modifiers :

modifiers are reusable code snippets that act as conditions or checks to control the behavior of functions. They are used to modify the execution flow of functions, ensuring certain conditions are met before or after a function runs. Modifiers can help reduce repetitive code and enhance contract security.

Visibility Modifiers:

- **public**: Accessible both internally and externally (outside the contract).
- **private**: Restricted to the contract where the variable or function is defined.
- **internal**: Accessible within the contract and derived contracts.
- **external**: Only accessible from external calls (outside the contract).

Mutability Modifiers:

- **view**: Guarantees the function will not modify the contract's state but allows reading from it.
- **pure**: Ensures the function does not read or modify the contract's state.
- **payable**: Allows the function to accept Ether transactions.

3) require : require is used to validate conditions before executing a function. It ensures that inputs or conditions meet specified criteria, such as checking if a user has enough funds for a transaction. If the condition fails, the transaction is reverted, and all changes to the state are undone, with an optional error message for clarity.

assert : assert is used to check for invariant conditions that should always be true within the contract, such as internal consistency after an operation. If assert fails, it indicates a serious error in the contract's logic, causing the transaction to revert and consuming all gas, making it unsuitable for user input validation.

revert : revert allows explicit rollback of a transaction when certain conditions aren't met. It can be used with a custom error message to provide more specific feedback when a failure occurs. This makes it useful for handling complex conditions where other error handling methods might not be as clear.

4) selfdestruct is an Ethereum opcode that allows a smart contract to delete itself from the blockchain and transfer any remaining Ether to a specified address.

Due to security risks like unauthorized destruction and forced Ether transfers, Ethereum is deprecating selfdestruct. The **Shanghai upgrade** (2023) introduced **SendAll semantics**, where contracts enter a "ghost" state but remain on-chain. The **EIP-4758** proposal seeks to fully deactivate selfdestruct. Developers are encouraged to avoid it and use safer alternatives like logical disable functions in new contracts.

5) **State-Changing Functions**: Functions that modify the blockchain state (e.g., payable functions) require a transaction to be mined and will not show output directly in Remix. The output appears in the transaction receipt or logs.

Read-Only Functions: Functions marked as view or pure do not modify the blockchain state and can return values directly, showing output immediately in Remix's function call results section.

In the case of transferEther(), the output isn't shown in Remix because it's a state-changing function that requires a transaction to be mined. The result is recorded in the transaction receipt. On the other hand, display() is a view function that doesn't change the state, so its output is displayed immediately in Remix.

6) A **Denial of Service (DoS) attack** is a malicious attempt to disrupt the normal operation of a network, service, or application by overwhelming it with an excessive amount of requests or malicious actions, making it unavailable to legitimate users. In the context of smart contracts, a DoS attack could involve strategies like exploiting contract vulnerabilities to prevent transactions from being processed or funds from being accessed. This could lead to a smart contract becoming non-functional or unresponsive, affecting its users.