1. Solidity의 기본 문법
    1) 솔리디티의 주요 자료형
        a. int : 정수형
        b. uint : 부호가 없는 정수형
        c. bool : 논리 자료형
        d. string : 문자열
        e. bytes : 바이트
        f. address : 이더리움 주소값
    2) 솔리디티는 자동 형 변환을 허용하지 않는다!! 따라서 형을 명시할 것
    3) payable를 통해 외부에서 이더를 송금 받을 수 있음
    4) 메시지 프로퍼티를 사용해 계약을 호출한 사람이 보낸 메시지를 확인할 수 있음
    5) 송금은 다음과 같은 흐름으로 이루어짐(A가 B에게 5이더를 준다)
        a. A가 계약 계정에게 5이더를 보냄
        b. 계약 계정이 B에게 5이더를 보냄
        c. transfer 함수를 이용
    6) 스마트 계약은 EVM으로 컴파일된 후, 이더리움 네트워크에 배포됨. 이 계약 코드는 Gas 수수료를 지불하여 사용 가능(Gas라는 수수료는 데이터를 검증하고 기록하는 노드들에 대한 대가로 지불)
    7) 이더의 송금과 같은 스마트 계약을 실행하게 되면 이더리움 블록체인에 트랜잭션으로 남게 된다.
        *트랜잭션은 다음과 같은 정보를 가진다.
        a. blockHash : 현재 트랜잭션이 추가된 블록의 해시값
        b. blockNumber : 현재 트랜잭션이 추가된 블록의 번호값
        c. transactionHash : 현재 트랜잭션의 해시값
        d. transactionIndex : 현재 트랜잭션의 인덱스 값
        e. gasUsed : 현재 트랜잭션 호출에 소비한 가스의 양
        f. contractAddress : 계약의 주소
        g. cumulativeGasUsed : 누적 가스 사용량

2. DApp 개발하기
    1) 솔리디티 버젼을 명시해주어야 함 ex) pragma solidity ^0.4.18;
    2) 접근 제어자
        a. public : 외부에서 호출이 가능하고 상태변수 선언시 Getter 함수를 생성
        b. internal : 이 상태 변수 또는 함수를 선언한 계약과 그 계약을 상속받은 계약에서만 호출
        c. private : 해당 계약에서만 호출 가능
        d. external : 인터페이스의 함수를 의미
    3) 데이터를 읽기만 하는 작업은 가스를 소모하지 않도록 view 키워드를 통해 함수 선언
        => 블록체인 네트워크 상의 데이터를 읽기만 할 수 있고 데이터를 수정할 수 없음
    4) 블록체인 네트워크에 기록된 데이터에 아예 접근하지 못하도록 하기 위해서 pure 키워드를 통해 함수 선언
        => 파라미터로 주어지지 않은 상태변수는 읽거나 쓸 수 없음