Constructor -
In constructor we define total supply of the token , token symbol, decimal value and name of the token
All the initial value of the token are put in the constructor

This is simple version of constructor

```
constructor(uint256 initialSupply) ERC20("Airdrop Token", "ATH") Ownable(msg.sender){
    _mint(msg.sender, initialSupply);
    emit TokenMinted(msg.sender, initialSupply);
}
```

This is Advance version of constructor

```
constructor(address defaultAdmin, address pauser, address minter)
    ERC20("Pandemic coin", "PDC")
    ERC20Permit("Pandemic coin")
{
    _grantRole(DEFAULT_ADMIN_ROLE, defaultAdmin);
    _grantRole(PAUSER_ROLE, pauser);
    _mint(msg.sender, 10000000000 * 10 ** decimals());
    _grantRole(MINTER_ROLE, minter);
}
```

Functions -

Mint-
Mint function is used to mint tokens on chain, only owner of the contract can mint these tokens

Here is the simple example of mint

```
function mint(uint256 _tokens) public onlyOwner {
    emit TokenMinted(msg.sender, _tokens);
    _mint(msg.sender, _tokens);
}
```

Here is advance example of mint

```
function mint(address to, uint256 amount) public onlyRole(MINTER_ROLE) {
    _mint(to, amount);
}

// The following functions are overrides required by Solidity.
```

Burn-
Burn is basically process of sending tokens to unrecoverable address on chain (usually 0x00..0)

Transfer-

It does what it says, it simply transfers tokens to destination address, either to another contract or wallet

Transferfrom is used via another smart contracts (DeFi, exchange etc) in order to make swaps and transactions

Here's an example of simple transfer function-

```
function transfer(address to, uint256 value) public override whenNotPaused returns (bool) {
    super.transfer(to, value);
    return true;
}

function transferFrom(address from, address to, uint256 value) public override  whenNotPaused returns (bool){
    super.transferFrom(from, to, value);
    return true;
}
```

Transfer ownership -
Transfer ownership is used to transferring ownership of the function from one address to another

```
function transferOwnership(address newOwner) public override onlyOwner {
    emit OwnershipTransfered(msg.sender, newOwner);
    transferOwnership(newOwner);
}
```

Pause-Unpause-

These function are used to Pause and Unpause the smart contract
They take boolean value as a parameter and works based on that