# Answers - Homework 4

## Upgradeable contract with version number

```
// SPDX-License-Identifier: UNLICENSED
pragma solidity ^0.8.4;

import "@openzeppelin/contracts-upgradeable/token/ERC20/ERC20Upgradeable.sol";
import "@openzeppelin/contracts-upgradeable/proxy/utils/Initializable.sol";
import "@openzeppelin/contracts-upgradeable/access/OwnableUpgradeable.sol";
import "@openzeppelin/contracts-upgradeable/proxy/utils/UUPSUpgradeable.sol";
import "@openzeppelin/contracts/utils/math/SafeMath.sol";
import "hardhat/console.sol";

contract DogCoinUups is
    Initializable,
    ERC20Upgradeable,
    OwnableUpgradeable,
    UUPSUpgradeable
{
    using SafeMath for uint256;

    address[] internal holders;

    mapping(address => uint256) internal balances;
    mapping(address => uint256) internal holderLocation;
    mapping(address => mapping(address => uint256)) internal allowed;

    event HolderAdded(address indexed holder);
    event HolderRemoved(address indexed holder);

    uint256 internal totalSupply_;

    string public version;

    function initialize(uint256 total) public initializer {
        totalSupply_ = total;
        balances[msg.sender] = totalSupply_;
        holders.push(msg.sender);
        holderLocation[msg.sender] = holders.length;
        version = "1.0";
        __ERC20_init("DogCoin", "DOG");
        __Ownable_init();
        __UUPSUpgradeable_init();
    }

    function _authorizeUpgrade(address newImplementation)
        internal
        override
```

```
    onlyOwner
{}

function totalSupply() public view override returns (uint256) {
    return totalSupply_;
}

function balanceOf(address tokenOwner)
    public
    view
    override
    returns (uint256)
{
    return balances[tokenOwner];
}

function transfer(address receiver, uint256 numTokens)
    public
    override
    returns (bool)
{
    require(numTokens <= balances[msg.sender]);
    balances[msg.sender] = balances[msg.sender].sub(numTokens);
    balances[receiver] = balances[receiver].add(numTokens);
    // Add to array
    if (holderLocation[receiver] == 0) {
        holders.push(receiver);
        holderLocation[receiver] = holders.length;
    }
    // Remove from array if zero balance
    if (balances[msg.sender] == 0) {
        removeFromArray(msg.sender);
    }
    emit Transfer(msg.sender, receiver, numTokens);
    return true;
}

function holdersLength() public view returns (uint256) {
    return holders.length;
}

function holdersArrayLocation(address _holder)
    public
    view
    returns (uint256)
{
    return holderLocation[_holder];
}

function removeFromArray(address _holder) internal {
    uint256 _holderIndex = holderLocation[_holder];
    address _toShift = holders[_holderIndex - 1];
```

```
        // Update holderLocation mapping
        holderLocation[_toShift] = 0;

        if (_holderIndex != 0) {
            uint256 lastIndex = holders.length - 1;
            if (lastIndex != _holderIndex - 1) {
                // Update holders
                holders[_holderIndex - 1] = holders[lastIndex];
            }
            holders.pop();
        }
        emit HolderRemoved(_holder);
    }

    function removeFromArrayLoop(address _holder) internal {
        for (uint256 index = 0; index < holders.length - 1; index++) {
            if (holders[index] == _holder) {
                if (index != holders.length - 1) {
                    holders[index] = holders[holders.length - 1];
                }
                holders.pop();
            }
        }
        emit HolderRemoved(_holder);
    }
}
```

## Modified hardhat.config.js

```
require("@nomiclabs/hardhat-waffle");
require('@nomiclabs/hardhat-ethers');
require('@openzeppelin/hardhat-upgrades');
require('@nomiclabs/hardhat-etherscan');


// This is a sample Hardhat task. To learn how to create your own go to
// https://hardhat.org/guides/create-task.html
task("accounts", "Prints the list of accounts", async (taskArgs, hre) => {
  const accounts = await hre.ethers.getSigners();

  for (const account of accounts) {
    console.log(account.address);
  }
});

// You need to export an object to set up your config
// Go to https://hardhat.org/config/ to learn more

/**
 * @type import('hardhat/config').HardhatUserConfig
 */
module.exports = {
  solidity: "0.8.4",
};
```

## deploy_upgradeable_dog.js

```
const { ethers, upgrades } = require('hardhat');

async function main() {
  const DogCoinUups = await ethers.getContractFactory('DogCoinUups');
  console.log('Deploying DogCoinUups...');
  const dogCoinUups = await upgrades.deployProxy(DogCoinUups, [10000000000], {
    initializer: 'initialize',
  });
  await dogCoinUups.deployed();
  console.log('DogCoinUups deployed to:', dogCoinUups.address);
}

main();
```

## Running Scripts

```
[tom@tom-r7 ExpertSolidity]$ npx hardhat run --network localhost
scripts/deploy_upgradeable_dog.js
Compiled 1 Solidity file successfully
Deploying DogCoinUups...
DogCoinUups deployed to: 0x2aB3C5B5e0bcb29ca85EF719418e51822e4e9159
[tom@tom-r7 ExpertSolidity]$ npx hardhat console --network localhost
Welcome to Node.js v16.14.0.
Type ".help" for more information.
> npx hardhat console --network localhost
> const Dog = await ethers.getContractFactory('DogCoinUups');
undefined
> const dog = await Dog.attach('0x2aB3C5B5e0bcb29ca85EF719418e51822e4e9159');
undefined
> (await dog.version());
'1.0'
```