TTM4195 Smart contracts in Solidity

1 Requirements

This assignment requires you to implement and demonstrate a smart contract in Solidity. You will not have to install any additional software, since we recommend the use of an online compiler. The assignment will help you understand how smart contracts work and how to deploy them. You should work in groups of 4 (3 if necessary).

In order to facilitate your meeting with Solidity programming, you will find on Blackboard a short introduction to Ethereum, Solidity and Smart Contracts. The notes will be uploaded together with this assignment. Please, refer to the Solidity website for the complete documentation of the latest version v0.8.26 (warning: the version keeps being updated).

In this assignment you will need two tools: an Ethereum wallet and Remix. A **digital wallet** (or **cryptocurrency wallet**) allows you to safely store cryptocurrencies (Ether, for this assignment). Nowadays, we have two families of wallets: **hardware wallets**, resembling an hard drive and **software wallets**. For Ethereum, the most popular (*and the one we strongly recommend you to use*) online wallet is Metamask, a browser extension that guarantees instant access to the Ethereum network (and to the testnet Goerli or Sepolia, which you will be using). Another valid option is Eidoo, from a non-custodian company, which comes in the form of app for iOs, Android and desktops.

You will make use of the **Remix IDE**, an open source software which allows you to test, debug and deploy smart contracts written in Solidity or Vyper for the Ethereum blockchain. It is available both offline and online, and we strongly recommend you to go for the online version. The interface is pretty simple and it even allows you to decide the compiler version; for any queries see the online documentation.

2 Deadlines

There are 15 marks available for this assignment:

1. **13 points for the code**, deadline on 14 November, 23:59:59. You will have to submit the code containing your implemented smart contract by Sunday 14 November, 23:59:59. The code will be evaluated within 2 weeks. As reference for the structure of your smart contract, see section 4. Uncommented/poorly commented code will result in a penalty of 2 points.

2. **7 points for the presentation/demonstration**. Students presentations (20 minutes per group) will take place Monday 18 November and Wednesday 20 November, during usual class hours. Each team will have to briefly explain how the smart contract works, and illustrate/motivate its approach (splitting of the workload, obstacles and difficulties, main design choices). Then each team will demonstrate a smart contract run. This can be done with a live demonstration OR by showing a recorded demonstration OR using screenshots from a test run.

Time-slot allocation will be performed on a first-come-first-served basis until Monday 11 November, 23:59:59. Reserve a time slot for your group using the link (will be given later).

If a group can demonstrate that no member can attend any of the two presentation sessions, a recorded video can be sent to me (jeongeun.park@ntnu.no), deadline on Wednesday 18 November, 08:00. In this unfortunate case, the group must inform me via email no later than Monday 11 November. Late notifications will result in 0 points for the presentation part.

Deadline for...	**on...**
booking a presentation slot communicating impossibility to give demo in person	11 Nov, 23:59:59
code delivery	14 November, 23:59:59
recorded demo (video) submission	17 November

3 Presentation

See Item 2 for details on how the presentation should be run and what it must contain. The presentation should last 15 minutes in total, followed by 5 minutes for questions. You will use your own laptop connected to the projector.

During your presentation you will be evaluated on

1. (2 point) clarity of the presentation, including keeping to time
2. (3 points) understanding of the concepts
3. (2 points) successful demonstration

4 The smart contract

For this assignment, we ask you to write a Solidity smart contract (SC) and an nonfungible token (NFT) for a car leasing system. Here is the scenario. Alice wants to drive a new and shiny electric car, but living on a PhD salary, she cannot afford to buy one. She therefore decides to sign a car lease with the famous BilBoyd dealership, leader in both automotive and blockchain markets.

1. (3 points): BilBoyd has several cars available for leasing. Implement each car using an NFT, containing the model, the colour, the year of matriculation and the original value.

2. (2 points): Each NFT comes with (in the sense that it determines) a monthly quota that depends on

   - the original car value;
   - the current car mileage;
   - the driver's experience (years of possession of a driving license, which affects the insurance cost);
   - a mileage cap (among a set of fixed values);
   - the contract duration (among a set of fixed values).

   Implement a function to compute this value, which should not cost any gas.

3. (2 points):After evaluating all the options, Alice chooses one and registers a deal on the blockchain. After BilBoyd has confirmed, this results in the transfer of a down payment (equivalent to 3 monthly quotas) and the first monthly quota. Write some functions to implement this phase as a fair exchange (the amount is locked in the SC, and it is unlocked only when BilBoyd signs too).

4. (2 points):This event (it is not required of you to implement events in Solidity for this assignment) also implies an expected monthly payment. Bilboyd must protect itself against insolvent customers. Implement some functionality that guarantees this.

5. At the end of the lease, Alice has three options:

   (a) (1 points) to simply terminate the contract;

   (b) (2 points) to extend the lease by one year. In this case, the monthly amount will be recomputed (in his favour, because of the change in the above parameters);

   (c) (1 points) to sign a lease for a new vehicle.