ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάθημα Προπτυχιακών Σπουδών:
Τεχνολογίες Blockchain και Εφαρμογές

Ακαδημαϊκό έτος: 2023 - 2024

Εξάμηνο: 8ο

Εργασία Εξαμήνου

Ομάδα Εργασίας:	**Υπεύθυνος Καθηγητής:**
Θεόδωρος Κοξάνογλου Π20094, Απόστολος Σιαμπάνης Π20173	Αριστέα Κοντογιάννη

Περιεχόμενα

1 Εκφώνηση Εργασίας

Στην παρούσα εργασία, θα δημιουργήσετε ένα έξυπνο συμβόλαιο ψηφοφορίας στο Remix IDE. Η εργασία αυτή στοχεύει στην εξοικείωσή σας με τη δημιουργία, ανάπτυξη και αλληλεπίδραση με ένα smart contract στο Ethereum blockchain χρησιμοποιώντας το Remix IDE. Το συμβόλαιο θα επιτρέπει τη δημιουργία νέων ψηφοφοριών, τη συμμετοχή χρηστών σε ψηφοφορίες, και την προβολή των αποτελεσμάτων. Κάθε χρήστης θα μπορεί να ψηφίσει μία φορά σε κάθε ψηφοφορία και η ψηφοφορία θα μπορεί να κλείσει χειροκίνητα.

Λειτουργίες

- **Δημιουργία Νέας Ψηφοφορίας:** Δημιουργεί μια νέα ψηφοφορία με ένα ερώτημα.
- **Ψήφος σε Ψηφοφορία:** Καταγραφή ψήφων χρηστών (Ναι/Όχι, true/false) σε μια καθορισμένη ψηφοφορία.
- **Κλείσιμο Ψηφοφορίας:** Χειροκίνητο «κλείσιμο» της ψηφοφορίας ώστε να μην δέχεται πλέον ψήφους.
- **Προβολή Αποτελεσμάτων:** Ανάκτηση των αποτελεσμάτων μιας ψηφοφορίας.

Προσέξτε ότι όλες οι λειτουργίες της εφαρμογής θα παρέχονται μέσω της κλήσης των κατάλληλων συναρτήσεων που θα υλοποιηθούν από το smart contract σας. Δεν απαιτείται η ανάπτυξη γραφικής διεπαφής (UI).

Πραγματικό Σενάριο Χρήσης

Ένα πραγματικό σενάριο χρήσης του smart contract μπορούσε να είναι η διεξαγωγή δημοσκοπήσεων σε μια κοινότητα ή έναν οργανισμό. Τα μέλη της κοινότητας μπορούν να δημιουργούν δημοσκοπήσεις για διάφορα θέματα, να ψηφίζουν και να βλέπουν τα αποτελέσματα σε πραγματικό χρόνο. Με αυτόν τον τρόπο, εξασφαλίζεται η διαφάνεια και η αξιοπιστία στη διαδικασία της ψηφοφορίας, καθώς όλες οι ψήφοι καταγράφονται στο blockchain. Ένα άλλο παράδειγμα είναι οι φοιτητικές εκλογές και η δυνατότητα των φοιτητών να δημιουργούν ψηφοφορίες για θέματα που αφορούν τη σχολή τους.

Σημειώσεις

1. Ο βαθμός της εργασίας θα συμμετέχει στον τελικό βαθμό του μαθήματος με συντελεστή 40%.
2. Η εργασία μπορεί να εκπονηθεί σε ομάδες έως δύο ατόμων.
3. Bonus μιας μονάδας για την ομάδα που επιλέξει να χρησιμοποιήσει το LaTeX για τη τεκμηρίωση (documentation) της εργασίας.
4. Για απορίες, διευκρινήσεις και εν γένει για ζητήματα που σχετίζονται με την παρούσα εργασία, μπορείτε να επικοινωνήσετε με την διδάσκουσα Δρ. Αριστέα Κοντογιάννη (**aristeakontogianni@gmail.com**).

Παραδοτέα

Τα παραδοτέα που πρέπει να υποβάλλει κάθε ομάδα είναι τα εξής:

1. **Αναφορά που θα περιέχει τα παρακάτω (documentation):**
    - **Ανάλυση Κώδικα:**
    - **Στιγμιότυπα Οθόνης:** Σχετικά στιγμιότυπα οθόνης (screenshots) που δείχνουν την ορθή δημιουργία (deployment) των συμβολαίων στο Remix IDE.
    - **Παράδειγμα Εκτέλεσης:** Ένα παράδειγμα εκτέλεσης για κάθε μία από τις ζητούμενες λειτουργίες του συμβολαίου (δημιουργία δημοσκόπησης, προβολή αποτελεσμάτων δημοσκόπησης, ψηφοφορία, κλείσιμο δημοσκόπησης), συνοδευόμενο από τα σχετικά screenshots.
2. **Αρχείο κώδικα**

Τα παραπάνω παραδοτέα θα πρέπει να αποσταλούν σε ένα αρχείο zip που θα έχει ως όνομα τον αριθμό μητρώου του φοιτητή/φοιτητών που την εκπόνησε. Σημειώνεται ότι η αναφορά θα πρέπει να υποβληθεί σε μορφή PDF (μαζί με το αντίστοιχο tex αρχείο αν υπάρχει).

Τρόπος και προθεσμία παράδοσης

Η παράδοση των εργασιών θα γίνεται ηλεκτρονικά μέσω της ασύγχρονης πλατφόρμας τηλεκπαίδευσης, με ημερομηνία που θα ανακοινωθεί.
Καλή επιτυχία!

2 Ανάλυση Κώδικα

2.1 Αρχική Ανάλυση

Για την ανάπτυξη του έξυπνου συμβολαίου (Smart Contract) **"VotingSystem"** χρησιμοποιήθηκε η γλώσσα προγραμματισμού Solidity στο περιβάλλον ανάπτυξης λογισμικού Remix. Το περιβάλλον του Remix παρέχει έναν εύκολο τρόπο στον χρήστη να εκτελέσει ένα έξυπνο συμβόλαιο με την χρήση ενός εικονικού δικτύου Ethereum, καθώς διαθέτει μερικούς δοκιμαστικούς λογαριασμούς με Ether για την δοκιμαστική εκτέλεση του συμβολαίου και των συναρτήσεων αυτού.

Αρχικά αναγράφουμε τον αναγνωριστικό κωδικό άδειας, όπου καθορίζει την άδεια που διαθέτει ο κώδικας μας καθώς και την έκδοση του compiler της γλώσσας Solidity που θέλουμε να χρησιμοποιήσουμε, όπως φαίνεται στην Εικόνα 1.

```
1  // SPDX-License-Identifier: Apache-2.0
2
3  pragma solidity >=0.8.2 <0.9.0;
```

Εικόνα 1: Ορισμός άδειας και έκδοσης του Compiler της Solidity.

2.2 Structure ψηφοφορίας

Η δομή της ψηφοφορίας ορίζεται με τη χρήση **struct** (βλέπε Εικόνα 2) με τις ακόλουθες μεταβλητές κατάστασης (state variables):

- **string question**: για την αποθήκευση του ερωτήματος της εκάστοτε ψηφοφορίας. Είναι σημαντικό να γνωρίζουμε σε αυτό το σημείο πως μια ψηφοφορία μπορεί να απαντηθεί μόνο με ναι ή όχι.
- **bool isClosed**: ορίζει την κατάσταση της ψηφοφορίας, για το αν αυτή είναι ανοιχτή ή κλειστή.
- **mapping(address => bool) hasVoted**: για την αποθήκευση των διευθύνσεων που έχουν ψηφίσει. Το κλειδί (key) του map είναι τύπου διεύθυνσης (address) και η τιμή (value) τύπου boolean.
- **uint yesVotes**: για την αποθήκευση του πλήθους των θετικών ψήφων.
- **uint noVotes**: για την αποθήκευση του πλήθους των αρνητικών ψήφων.

```
7    // Struct to represent a poll
8    struct Poll {
9        // Poll question
10       string question;
11       // Poll status
12       bool isClosed;
13       // Voters, users who have voted
14       mapping(address => bool) hasVoted;
15       // Poll results
16       uint yesVotes;
17       uint noVotes;
18   }
```

Εικόνα 2: Struct της ψηφοφορίας.

Για την αποθήκευση των ψηφοφοριών που δημιουργούνται χρησιμοποιείτε μια μεταβλητή κατάστασης (state variable) με τη μορφή πίνακα (array) όπου θα περιέχει τιμές τύπου του **struct Poll**, όπως φαίνεται και στην Εικόνα 3.

```
20   // State Variable
21   Poll[] private polls; // Array to store the polls
```

Εικόνα 3: Array αποθήκευσης ψηφοφοριών.

2.3 Function δημιουργίας νέας ψηφοφορίας

Για τη δημιουργία μιας νέας ψηφοφορίας χρησιμοποιείται η συνάρτηση **createNewPoll**. Αυτή η συνάρτηση δέχεται ως όρισμα την ερώτηση της νέας ψηφοφορίας από τον χρήστη. Συγκεκριμένα, η συνάρτηση δημιουργεί την ψηφοφορία και την αποθηκεύει στον πίνακα **polls**. Στη συνέχεια, αρχικοποιεί τις μεταβλητές που περιέχουν τα ακόλουθα:

- **question**: Το ερώτημα της ψηφοφορίας που δίνεται ως όρισμα από τον χρήστη.
- **isClosed**: Η αρχική κατάσταση της ψηφοφορίας, η οποία είναι ανοιχτή.
- **yesVotes**: Το αρχικό πλήθος των θετικών ψήφων, που αρχικά είναι μηδενικό.
- **noVotes**: Το αρχικό πλήθος των αρνητικών ψήφων, που επίσης αρχικά είναι μηδενικό.

```
23      // Function to create a new poll
24      function createNewPoll(string calldata _question) public {
25          // Create a new poll
26          Poll storage newPoll = polls.push();
27          // Initialize poll's question
28          newPoll.question = _question;
29          // Initialize poll's status
30          newPoll.isClosed = false;
31          // Initialize poll's results
32          newPoll.yesVotes = 0;
33          newPoll.noVotes = 0;
34      }
```

Εικόνα 4: Δημιουργία μίας νέας φηφοφορίας.

2.4 Function δημιουργίας ψήφου ψηφοφορίας

Για τη δημιουργία μιας ψήφου σε μια ψηφοφορία χρησιμοποιείται η συνάρτηση **castVote**, η οποία δέχεται ως όρισμα τον δείκτη (index) της ψηφοφορίας. Αρχικά, με τη χρήση της εντολής **require**, ελέγχεται εάν ο δείκτης της ψηφοφορίας αυτής υπάρχει στον πίνακα **polls**. Στη συνέχεια, η ψηφοφορία αυτή αποθηκεύεται προσωρινά σε μια μεταβλητή τύπου **storage**. Σε αυτό το σημείο γίνεται έλεγχος για την κατάσταση της ψηφοφορίας, εάν είναι ανοιχτή ή κλειστή, και ελέγχεται επίσης εάν ο χρήστης έχει ήδη ψηφίσει σε αυτήν. Για την απόκτηση της διεύθυνσης του χρήστη χρησιμοποιείται το αντικείμενο **msg**, το οποίο περιέχει πληροφορίες σχετικά με τη συναλλαγή. Είναι σημαντικό να σημειωθεί ότι για να αποθηκευτεί μια ψήφος στο blockchain, πρέπει να πληρούνται όλοι οι παραπάνω έλεγχοι. Εάν πληρούνται, η εκτέλεση του κώδικα του έξυπνου συμβολαίου συνεχίζεται με την καταχώρηση της ψήφου, αυξάνοντας την αντίστοιχη μεταβλητή είτε για τις θετικές είτε για τις αρνητικές ψήφους.

```
36      // Function to vote on a poll
37      function castVote(uint _poll_index, bool _vote) public {
38          // Check if the poll index is valid
39          require(_poll_index < polls.length, "Invalid poll index");
40
41          // Get the poll
42          Poll storage poll = polls[_poll_index];
43
44          // Check if the poll is closed
45          require(!poll.isClosed, "Poll is closed");
46
47          // Check if the voter has already voted
48          require(!poll.hasVoted[msg.sender], "The user has already voted");
49
50          // Mark the voter as voted
51          poll.hasVoted[msg.sender] = true;
52
53          // Update Poll results based on the vote
54          if (_vote) {
55              poll.yesVotes++;
56          } else {
57              poll.noVotes++;
58          }
59      }
```

Εικόνα 5: Δημιουργία ψήφου.

2.5 Function κλείσιμο ψηφοφορίας

Για το κλείσιμο μιας ψηφοφορίας χρησιμοποιείται η συνάρτηση **”closePoll”** η οποία δέχεται ως όρισμα τον δείκτη (index) της ψηφοφορίας. Αρχικά με την χρήση μιας εντολής **require** ελέγχει εάν ο δείκτης της συγκεκριμένης ψηφοφορίας υπάρχει στον πίνακα **”polls”**. Στην συνέχεια βρίσκει και αποθηκεύει προσωρινά σε μια μεταβλητή τύπου **storage** την ψηφοφορία αυτή και στο σημείο αυτό ελέγει την κατάσταση της ψηφοφορίας αν είναι ήδη κλειστή. Στην περίπτωση που τηρούνται τα παραπάνω αλλάζει η κατάσταση (status) της ψηφοφορίας σε ”κλειστή” μεταβάλοντας την τιμή στην μεταβλητή κατάστασης (state variable) **”isClosed”**.

```
61      // Function to close a poll
62      function closePoll(uint _poll_index) public {
63          // Check if the poll index is valid
64          require(_poll_index < polls.length, "Invalid poll index");
65
66          // Get the poll
67          Poll storage poll = polls[_poll_index];
68
69          // Check if the poll is already closed
70          require(!poll.isClosed, "Poll is already closed");
71
72          // Close the poll
73          poll.isClosed = true;
74      }
```

Εικόνα 6: Κλείσιμο μίας ψηφοφορίας.

2.6 Function προβολή αποτελεσμάτων ψηφοφορίας

Για την προβολή των αποτελεσμάτων μιας ψηφοφορίας χρησιμοποιείται η συνάρτηση **”showPollResults”** η οποία δέχεται ως όρισμα τον δείκτη (index) της ψηφοφορίας. Αρχικά με την χρήση μιας εντολής **require** ελέγχει εάν ο δείκτης της συγκεκριμένης ψηφοφορίας υπάρχει στον πίνακα **”polls”**. Στην περίπτωση που τηρείται η παραπάνω συνθήκη βρίσκει και αποθηκεύει προσωρινά σε μια μεταβλητή τύπου **storage** την ψηφοφορία αυτή. Τέλος, η συνάρτηση επιστρέφει με δύο ακέραιους αριθμούς το πλήθος των θετικών **”yesVotes”** και αρνητικών **”noVotes”** ψήφων.

```
76      // Function to show poll results
77      function showPollResults(uint _poll_index) public view returns (uint, uint) {
78          // Check if the poll index is valid
79          require(_poll_index < polls.length, "Invalid poll index");
80
81          // Get the poll
82          Poll storage poll = polls[_poll_index];
83
84          // Return the poll results
85          return (poll.yesVotes, poll.noVotes);
86      }
87
88  }
```

Εικόνα 7: Εμφάνιση αποτελεσμάτων μίας ψηφοφορίας.

3 Παράδειγμα Εκτέλεσης

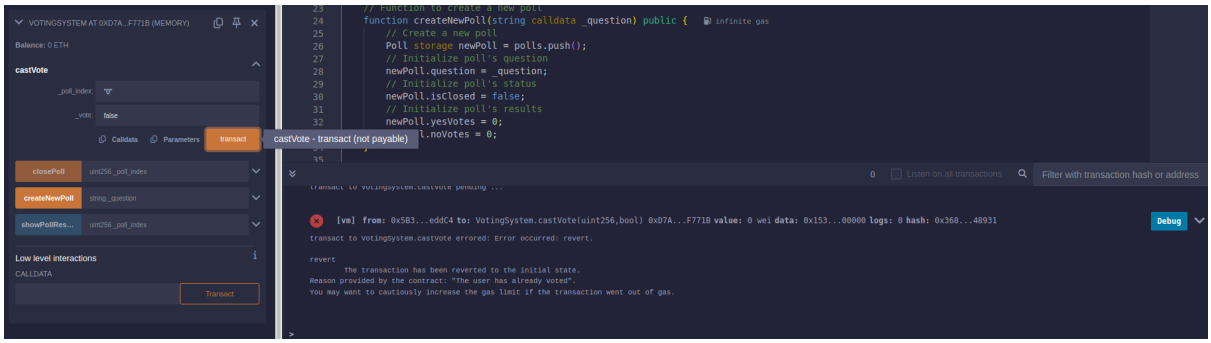
Για την εκτέλεση του συμβολαίου χρησιμοποιήθηκε το **Remix IDE** λόγω φιλικού χρηστικού περιβάλλοντος.

3.1 Δημιουργία Νέας Ψηφοφορίας

Εικόνα 8: Δημιουργία μίας ψηφοφορίας.

3.2 Ψήφος σε Ψηφοφορία

Εικόνα 9: Δημιουργία Ψήφου.



Εικόνα 10: Άρνηση αποδοχής ψήφου από χρήστη που έχει ήδη ψηφίσει.

3.3 Κλείσιμο Ψηφοφορίας

Εικόνα 11: Κλείσιμο Ψηφοφορίας.

3.4 Προβολή Αποτελεσμάτων

Εικόνα 12: Εμφάνιση Αποτελεσμάτων Ψηφοφορίας.

4 Πηγές

1. **Παρουσίαση Solidity:**
   Εισαγωγή στη Solidity
   Available online: https://thales.cs.unipi.gr/modules/document/file.php/TMD140/%CE%94%CE%B9%CE%B1%CE%BB%CE%AD%CE%BE%CE%B5%CE%B9%CF%82%20%CE%9C%CE%B1%CE%B8%CE%AE%CE%BC%CE%B1%CF%84%CE%BF%CF%82%202024/solidity.pdf
   Accessed on: 11/07/2024

2. **Παρουσίαση Solidity:**
   Remix Μεταβλητές κατάστασης & ορατότητα
   Available online: https://thales.cs.unipi.gr/modules/document/file.php/TMD140/%CE%94%CE%B9%CE%B1%CE%BB%CE%AD%CE%BE%CE%B5%CE%B9%CF%82%20%CE%9C%CE%B1%CE%B8%CE%AE%CE%BC%CE%B1%CF%84%CE%BF%CF%82%202024/solidity_2.pdf
   Accessed on: 11/07/2024