Blockchains and Smart Contracts
Term Assignment

1. CryptoSOS

Καλείστε να γράψετε ένα smart contract στη γλώσσα Solidity, που να υλοποιεί το γνωστό παιχνίδι SOS στο περιβάλλον του Ethereum.

Κανόνες του Παιχνιδιού SOS

Το κλασικό αυτό παιχνίδι αυτό παίζεται από δύο παίκτες εναλλάξ. Υπάρχουν 9 τετράγωνα οργανωμένα σε ένα grid 3x3, που στα πλαίσια της άσκησης θα τα θεωρούμε αριθμημένα όπως στο σχήμα 1(a).

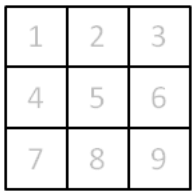

Fig 1(a)

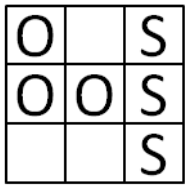

Fig 1(b)

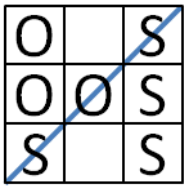

Fig 1(c)

Τα τετράγωνα είναι αρχικά άδεια. Σε κάθε κίνηση, ο παίκτης που παίζει οφείλει να βάλει ένα (και μόνο ένα) **S** ή **O** σε όποιο άδειο τετράγωνο επιθυμεί. Το παιχνίδι συνεχίζεται μέχρι να σχηματιστεί το ακρωνύμιο SOS οριζοντίως, καθέτως, ή διαγωνίως, ή μέχρι να μην υπάρχουν άλλα άδεια τετράγωνα. Αν κάποιος παίκτης σχηματίσει SOS κερδίζει την παρτίδα. Για παράδειγμα, αν μια παρτίδα βρίσκεται στην κατάσταση του σχήματος 1(b) και ο παίκτης που έχει σειρά βάλει ένα S στο τετράγωνο 7, σχηματίζει SOS όπως στο σχήμα 1(c) και κερδίζει την παρτίδα.

CryptoSOS API

Για να παίξει κάποιος πρέπει να καλέσει τη συνάρτηση **`play()`** του CryptoSOS. Τότε μπαίνει σε κατάσταση αναμονής μέχρι και κάποιος άλλος παίκτης να καλέσει το **`play()`**, οπότε και ξεκινάει μια παρτίδα.

Προσοχή: Κάποιος παίκτης δεν έχει δικαίωμα να παίξει παρτίδα με τον εαυτό του.

Στη συνέχεια, όποτε είναι η σειρά του καθενός, μπορεί να καλεί είτε την **`placeS(uint8)`** ή την **`placeO(uint8)`**, για να τοποθετεί S ή O σε τετράγωνο της επιλογής του. Η παράμετρος της συνάρτησης αυτής θα πρέπει να είναι ένας αριθμός από το 1 έως το 9.

Ανά πάσα στιγμή, μπορεί κάποιος να καλέσει τη function **`getGameState()`** η οποία θα επιστρέψει ένα string 9 χαρακτήρων, στο οποίο ο κάθε χαρακτήρας θα είναι από το σύνολο {-,S,O} και θα συμβολίζει την κατάσταση του αντίστοιχου τετραγώνου (ο $1^{ος}$ χαρακτήρας από αριστερά για το τετράγωνο 1, κ.ο.κ.), όπου προφανώς η παύλα αντιστοιχεί στο κενό τετράγωνο. Π.χ., για την κατάσταση του Fig 1(b) θα επέστρεφε το string **`O-SOOS--S`**.

Κόστος συμμετοχής και έπαθλα

Για να συμμετάσχει ένας παίκτης, οφείλει να καταβάλει ακριβώς 1 Ether κατά την κλήση της συνάρτησης **`play()`**. Στο τέλος της παρτίδας, ο νικητής πληρώνεται 1.7 Ether, ενώ τα υπόλοιπα 0.3 παραμένουν στο αποθεματικό του CryptoSOS. Σε περίπτωση ισοπαλίας θα πρέπει να επιστρέφονται 0.8 Ether στον κάθε παίκτη και τα υπόλοιπα να παραμένουν στο αποθεματικό. Μόνο ο owner του CryptoSOS (δηλ. το account που έκανε deploy το CryptoSOS) μπορεί να πάρει χρήματα από το αποθεματικό του παιχνιδιού, με τη συνάρτηση **`sweepProfit()`**, η οποία μεταφέρει όλο το υπόλοιπο στον λογαριασμό του owner.

Events

Όταν κάποιος παίκτης καλέσει επιτυχώς (δηλ. με την απαραίτητη πληρωμή) τη συνάρτηση **`play()`**, γίνεται emitted ένα Event τύπου **`StartGame(address,address)`**, που ανακοινώνει τις διευθύνσεις των δύο παικτών που ξεκινούν την παρτίδα. Μόλις δηλώσει ο πρώτος παίκτης, στέλνεται τέτοιο event με τη διεύθυνσή του και με address μηδέν για τον δεύτερο παίκτη, ενώ όταν δηλώσει κι ο δεύτερος στέλνεται και πάλι event του ίδιου τύπου αλλά αυτή τη φορά και με τις δύο addresses. Ένα τέτοιο event με μη μηδενικές και τις δύο διευθύνσεις ουσιαστικά σηματοδοτεί την έναρξη μιας παρτίδας.

Μετά από κάθε κίνηση, στέλνεται ένα event τύπου **`Move(uint8, uint8, address)`**. Η πρώτη παράμετρος είναι το τετράγωνο που έθεσε (1..9), η δεύτερη είναι 1 για S, και 2 για O, και η τρίτη είναι η διεύθυνση του παίκτη που έπαιξε. Όταν ολοκληρωθεί η παρτίδα πρέπει να σταλεί ένα event τύπου **`Winner(address)`**, όπου αναφέρεται η διεύθυνση του νικητή, ή σε περίπτωση ισοπαλίας, ένα event τύπου **`Tie(address, address)`** όπου θα αναφέρονται οι διευθύνσεις και των δύο παικτών (με τη σειρά που έχουν κάνει register).

Δικλείδες ασφαλείας

Αν κάποιος παίκτης δηλώσει να παίξει, και μέσα σε 2 λεπτά της ώρας δεν έχει ακόμα δηλώσει κάποιος δεύτερος παίκτης, δικαιούται να πάρει όλα τα λεφτά του (1 Ether) πίσω, καλώντας τη συνάρτηση **`cancel()`**.

Αν έχει ξεκινήσει η παρτίδα, και περάσει 1 λεπτό από την κίνηση ενός παίκτη και ο άλλος δεν έχει παίξει ακόμα, τότε αυτός που έκανε την τελευταία κίνηση δικαιούται να καλέσει τη συνάρτηση **`tooslow()`** και να πάρει πίσω 1.9 Ether, αφήνοντας 0.1 Ether κέρδος για το αποθεματικό του CryptoSOS, και τερματίζοντας έτσι πρόωρα την παρτίδα. Πάλι πρέπει να γίνει emitted το event **`Winner(address)`**. Άμα κανείς από τους δύο παίκτες δεν έχει κάνει κάποια κίνηση για 5 συνεχόμενα λεπτά, θα μπορεί ο owner να καλεί την **`tooslow()`** τερματίζοντας την παρτίδα με ισοπαλία (επιστρέφοντας 0.8 Ether στον κάθε παίκτη και κάνοντας emit το Tie event όπως περιγράφεται πιο πάνω).

2. MultiSOS

Στο δεύτερο σκέλος της άσκησης θα υλοποιήσετε ένα ξεχωριστό smart contract με όνομα MultiSOS, το οποίο θα υλοποιεί το ίδιο ακριβώς παιχνίδι, με τη διαφορά ότι θα υποστηρίζει την παράλληλη εκτέλεση πολλών παρτίδων. Ο πρώτος που καλεί την **`play()`** θα μπαίνει σε αναμονή μέχρι ένας δεύτερος παίκτης να καλέσει την **`play()`**, οπότε και θα ξεκινάει μια παρτίδα μεταξύ τους. Αν τώρα ένας τρίτος και τέταρτος καλέσουν με τη σειρά τους την **`play()`** πριν τελειώσει η πρώτη παρτίδα, θα ξεκινάει άμεσα μια δεύτερη, παράλληλα με την πρώτη. Θα μπορούν να ξεκινήσουν απεριόριστες παρτίδες που θα τρέχουν παράλληλα.

Το API του MultiSOS θα περιλαμβάνει τις ίδιες συναρτήσεις με αυτό του CryptoSOS.

- **UPDATE: Το κάθε address μπορεί να συμμετέχει το πολύ σε μία παρτίδα ανά πάσα στιγμή! Αυτό θα σας επιτρέψει να έχετε ακριβώς το ίδιο API και στα δύο smart contracts.**

3. Γενικές οδηγίες

Οι μέθοδοι που αναφέρονται στην εκφώνηση επαρκούν για την υλοποίηση των smart contracts. Είστε ελεύθεροι να υλοποιήσετε όσες βοηθητικές, εσωτερικές μεθόδους θέλετε, αλλά για το public API θα πρέπει να αποφύγετε την υλοποίηση επιπλέον μεθόδων από αυτές που αναφέρονται στην εκφώνηση. Σε περίπτωση που υλοποιήσετε επιπλέον μεθόδους, θα πρέπει να περιγράψετε αναλυτικά τη λειτουργικότητά τους στο Readme αρχείο.

Όταν συμβαίνει κάποιο σφάλμα λόγω του input ενός χρήστη, θα πρέπει να του επιστρέφεται ένα περιγραφικό μήνυμα, π.χ., μέσω require() ή revert(), το οποίο να του εξηγεί γιατί το input που έβαλε είναι λάθος.

Η υλοποίησή σας θα πρέπει να στοχεύει στην ελαχιστοποίηση των gas costs, τόσο στη φάση του deployment του smart contract (δηλ. να μην είναι αχρείαστα μεγάλος ο κώδικας), όσο και κατά τη μεμονωμένη κλήση της κάθε μεθόδου (δηλ. προσοχή στο storage και στο computation).

Θα πρέπει να δοθεί ιδιαίτερη προσοχή στην αποτροπή των επιθέσεων που είδαμε στο μάθημα. Ενδεχομένως, μπορεί να βρείτε και να χρησιμοποιήσετε καλές πρακτικές που να κάνουν τα smart contract σας ανθεκτικά σε ευρέως γνωστές επιθέσεις. Θα πρέπει να αναφερθείτε στις επιθέσεις αυτές αναλυτικά στο παραδοτέο Readme αρχείο.

Ο κώδικάς σας πρέπει να είναι σαφής σχετικά με το *τι* κάνει και το *γιατί* σε κάθε σημείο του. Για να δικαιολογήσετε τμήματα κώδικα που δεν είναι ξεκάθαρο το τι κάνουν, χρησιμοποιήστε ευανάγνωστα σχόλια μέσα στον κώδικα ή/και εξηγήστε τα στο παραδοτέο Readme αρχείο. Ενθαρρύνουμε πολύ την προσθήκη σχολίων στον κώδικα, ιδίως σε σημεία που θεωρείτε ότι είναι σημαντικά.

4. Βαθμολόγηση και Συνεργασία

Η άσκηση αντιστοιχεί σε **2.5 από τις 10 μονάδες** του τελικού βαθμού του μαθήματος.

Οι υπόλοιπες **7.5 μονάδες** προκύπτουν από το τελικό διαγώνισμα.

Για τη βαθμολόγηση της άσκησης, με άριστα το 10, οι βαθμοί θα υπολογιστούν ως εξής:

- **CryptoSOS:** 7 μονάδες
- **MultiSOS:** 3 μονάδες

Η άσκηση είναι **ατομική**. Οποιαδήποτε συνεργασία, λύσεις που έχουν ύποπτες ομοιότητες, ή λύσεις βασισμένες σε «δανεικό» κώδικα, θα μηδενίζονται πλήρως. Αυτό αφορά όλους, όχι μόνο όσους έλαβαν βοήθεια αλλά και όσους έδωσαν.

5. Οδηγίες υποβολής

Στείλτε στο voulgaris+sos@aueb.gr (απαραιτήτως σ' αυτό το email!) τα smart contracts που έχετε φτιάξει, ως συνημμένα αρχεία (καθαρά, σκέτα, ασυμπίεστα) με filenames **`CryptoSOS.sol`** και **`MultiSOS.sol`**, καθώς και ένα σύντομο **`Readme.txt`** ή **`Readme.pdf`** με μια σύντομη περιγραφή σημείων του κώδικα που θέλετε να επισημάνετε, δυσκολιών που αντιμετωπίσατε και των λύσεων που δώσατε.

Η προθεσμία υποβολής είναι μέχρι την **Παρασκευή 12 Ιανουαρίου 2024** το βράδυ.