Εργασία 1 – Λαχειοφόρος αγορά (30%)

Πρόβλημα:

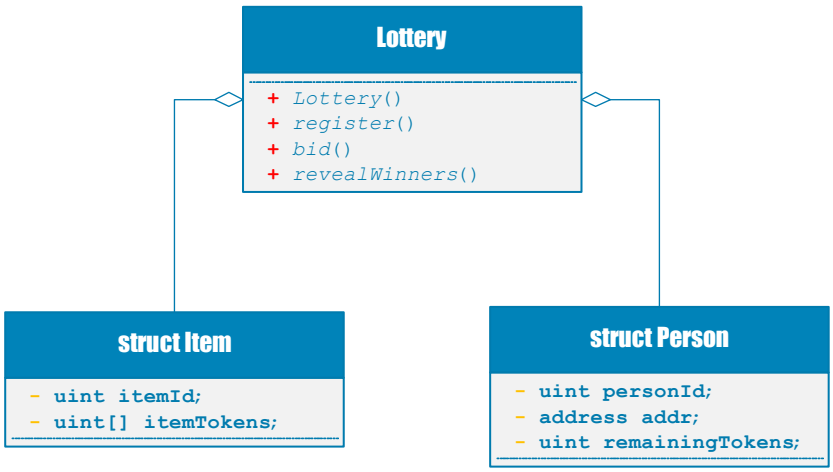
Για τη συγκέντρωση χρημάτων για έναν κοινωφελή σκοπό, ένας σύλλογος συλλέγει αντικείμενα προς κλήρωση. Πριν από την κλήρωση, τα αντικείμενα τακτοποιούνται στη σειρά, τοποθετώντας μια κάλπη/κληρωτίδα μπροστά από το καθένα.

Κάποιοι παίκτες αγοράζουν λαχεία. Τα λαχεία πωλούνται σε 5άδες. Κάθε λαχείο λειτουργεί σαν ένα token ή μάρκα που θα τοποθετηθεί σαν κλήρος στην παραπάνω κληρωτίδα. Τη στιγμή που ο παίκτης αγοράζει κάποιο λαχείο, αυτό «σφραγίζεται» με τα στοιχεία του (με τον ταυτοτικό αριθμό του). Οι παίκτες τοποθετούν ένα ή περισσότερα από τα λαχεία τους στην κάλπη όποιου ή όποιων αντικειμένων θέλουν να κερδίσουν. Μπορούν να το κάνουν μέχρι να ξοδέψουν όλα τους τα λαχεία. Αφού οι παίκτες έχουν ξοδέψει τα λαχεία τους, θα επιλεγεί τυχαία ένα λαχείο από την κάθε κληρωτίδα, για να βρεθεί ο νικητής (νικητής του αντίστοιχου αντικειμένου). Όσα πιο πολλά λαχεία/κλήρους έχει κάποιος σε μια κληρωτίδα, τόσο μεγαλύτερη η πιθανότητα να επιλεγεί ως ο νικητής της.
Την κλήρωση για κάθε αντικείμενο την κάνει ο πρόεδρος του συλλόγου και επικεφαλής της δράσης (μόνο αυτός έχει δικαίωμα). Όμως, κληρωτίδα που δεν περιέχει λαχείο, ΔΕΝ ανακηρύσσει νικητή (δεν επιτρέπει την κλήρωση).

Υπόδειγμα – Μερική υλοποίηση:

Κατασκευάστε πρόγραμμα solidity που υλοποιεί την παραπάνω λαχειοφόρο αγορά ως smart contract.

Παρακάτω ακολουθεί τμήμα προτεινόμενης κλάσης του συμβολαίου. Προσπαθήστε να ακολουθήσετε τα ονόματα (Δεν είναι υποχρεωτικό αλλά θα βοηθήσει στην κατανόηση του προγράμματός σας από τον αξιολογητή).



Ακολουθεί ένα μεγάλο τμήμα μιας ενδεχόμενης υλοποίησης για ΑΚΡΙΒΩΣ 4 παίκτες και 3 αντικείμενα. **(Το δικό σας πρόγραμμα ΠΡΕΠΕΙ να υποστηρίζει μεταβλητό πλήθος παικτών και αντικειμένων)**

```
mapping(address => Person) tokenDetails; // διεύθυνση παίκτη
Person [4] bidders; // πίνακας 4 παικτών

Item [3] public items; // πίνακας 3 αντικειμένων
```

```
address[3] public winners; // πίνακας νικητών - η τιμή 0 δηλώνει πως δεν υπάρχει νικητής
address public beneficiary; // ο πρόεδρος του συλλόγου και ιδιοκτήτης του smart contract

uint bidderCount = 0; // πλήθος των εγγεγραμένων παικτών

constructor() public payable{    //constructor

    // Αρχικοποίηση του προέδρου με τη διεύθυνση του κατόχου του έξυπνου συμβολαίου
    beneficiary = msg.sender;
    uint[] memory emptyArray;
    items[0] = Item({itemId:0, itemTokens:emptyArray}); // το πρώτο αντικείμενο
    items[1] = Item({itemId:1, itemTokens:emptyArray});
    items[2] = Item({itemId:2, itemTokens:emptyArray});
}

function register() public payable{ // εγγραφή παίκτη

    bidders[bidderCount].personId = bidderCount;

    // Αρχικοποίηση της διεύθυνσης του παίκτη
    bidders[bidderCount].addr = msg.sender;

    bidders[bidderCount].remainingTokens = 5; // μόνο 5 λαχεία
    tokenDetails[msg.sender] = bidders[bidderCount];
    bidderCount++;
}

function bid(uint _itemId, uint _count) public payable { // Ποντάρει _count λαχεία στο
αντικείμενο _itemId

    /*
    Δύο έλεγχοι:
        ο παίκτης που καλεί έχει επαρκές πλήθος λαχείων;
        το αντικείμενο υπάρχει;
    */

    /*
    Ενημέρωση του υπολοίπου λαχείων του παίκτη
    */

    /*
    Ενημέρωση της κληρωτίδας του _itemId με εισαγωγή των _count λαχείων που ποντάρει ο παίκτης
    */
}

function revealWinners() public onlyOwner { // θα υλοποιήσετε modifier με το όνομα onlyOwner

    /*
        Για κάθε αντικείμενο που έχει περισσότερα από 0 λαχεία στην κάλπη του
        επιλέξτε τυχαία έναν νικητή από όσους έχουν τοποθετήσει το λαχείο τους
    */

    for (uint id = 0; id < 3; id++) { // Εδώ για 3 μόνο αντικείμενα

        // παραγωγή τυχαίου αριθμού

        // ανάκτηση του αριθμού παίκτη που είχε αυτό το λαχείο

        // ενημέρωση του πίνακα winners με τη διεύθυνση του νικητή

    }
}
```

Ζητήματα:

1. Το συμβόλαιο υποστηρίζει μεταβλητό πλήθος παικτών και αντικειμένων. **(10%)**

2. Η κλήση της συνάρτησης **revealWinners** επιτρέπεται ΜΟΝΟ από τον ιδιοκτήτη του συμβολαίου (με χρήση modifier). Επίσης, επιτρέπεται ΜΟΝΟ αν για κάποιο αντικείμενο δεν υπάρχει ακόμη νικητής. Η συνάρτηση εκτελεί κλήρωση μόνο σε όσα αντικείμενα έχουν τοποθετήσει λαχεία/κλήρους οι παίκτες (δεν χρειάζεται απόλυτα ασφαλής γεννήτρια τυχαίων αριθμών), και μόνο αν δεν υπάρχει ήδη νικητής για αυτά. Οι αριθμοί των νικητών

καταχωρούνται σε πίνακα. Όταν δεν είναι δυνατή η κλήρωση (όταν δεν υπάρχουν καθόλου λαχεία/κλήροι στην κληρωτίδα), στον πίνακα καταχωρείται ο αριθμός 0 (Οι παίκτες αριθμούνται από το 1). **(20%)**

3. Η κλήση της συνάρτησης **bid** επιτρέπεται ΜΟΝΟ από κάποιον που έχει επαρκές πλήθος λαχείων (μεγαλύτερο ή ίσο με το πλήθος που ποντάρει) και τα τοποθετεί σε «υπαρκτό» αντικείμενο (με χρήση modifiers). **(20%)**

4. Η κλήση της συνάρτησης **register** επιτρέπεται ΜΟΝΟ από κάποιον που μεταφέρει στο συμβόλαιο (πληρώνει) ποσό μεγαλύτερο ή ίσο από 0,005 (δοκιμαστικά) Ether (γίνεται χρήση modifier ώστε αν το υπόλοιπο του πορτοφολιού του δεν επαρκεί, να γίνεται revert). Η κλήση επιτρέπεται ΜΟΝΟ από κάποιον που δεν είναι ήδη καταχωρημένος. Η κλήση δεν επιτρέπεται στον ιδιοκτήτη του συμβολαίου. Αυτός που την καλεί με επιτυχία, «κατέχει» πλέον 5 λαχεία. **(20%)**

5. Δημιουργήστε μια επιπλέον συνάρτηση **withdraw** με την οποία ο ιδιοκτήτης του συμβολαίου μεταφέρει όλα τα δοκιμαστικά Ether από το συμβόλαιο (όσα δώσαν οι παίκτες) στο πορτοφόλι του. Η εκτέλεση επιτρέπεται μόνο από τον ιδιοκτήτη του συμβολαίου (με χρήση modifier). **(5%)**

6. Δημιουργήστε μια επιπλέον συνάρτηση **reset** με την οποία ο ιδιοκτήτης του συμβολαίου μπορεί να κάνει «επανεκκίνηση» του συμβολαίου με νέο πλήθος αντικειμένων. Τα αντικείμενα εκκινούν με άδειες κληρωτίδες και δεν υπάρχει πλέον κανείς καταχωρημένος παίκτης (αν υπήρχαν διαγράφονται) και κανείς νικητής. **(5%)**

7. Κάντε χρήση της μεταβλητής **stage** τύπου **Stage** όπου ο enum τύπος Stage ορίζεται ως

```
enum Stage {Init, Reg, Bid, Done}
Stage public stage;
```

Τροποποιήστε τις συναρτήσεις ώστε εγγραφές παικτών να επιτρέπονται μόνο όταν η stage είναι Reg, οι τοποθετήσεις λαχείων σε κληρωτίδες μόνο όταν η stage είναι Bid και οι κληρώσεις να επιτρέπονται μόνο όταν η stage είναι Done. Δημιουργήστε μια επιπλέον συνάρτηση **advanceState** με την οποία ο ιδιοκτήτης του συμβολαίου μπορεί να μετακινήσει την εκτέλεση σε επόμενο στάδιο. Μόνο η **reset** πλέον θα επαναφέρει την stage σε Reg. **(5%)**

8. Κάθε φορά που προκύπτει νικητής για κάποιο αντικείμενο, να πυροδοτείται (emit) το συμβάν **Winner** το οποίο θα καταγράφει τη διεύθυνση του παίκτη νικητή, τον αριθμό του αντικειμένου που κέρδισε και τον αριθμό της συγκεκριμένης λαχειοφόρου. Η πρώτη λαχειοφόρος είναι αμέσως μετά τη δημοσίευση του συμβολαίου. Κάθε φορά που γίνεται **reset**, ο αριθμός της λαχειοφόρου αυξάνεται κατά ένα. **(5%)**

9. Γράψτε το smart contract σε solidity και με τη βοήθεια του Remix δημοσιεύστε το σε κάποιο δημόσιο δοκιμαστικό δίκτυο (π.χ. Sepolia). Ετοιμάστε ένα βίντεο (screencast) ανάλυσης καλύτερης από 720p και διάρκειας 2 -5 λεπτών. Στο video θα παρουσιάζεται μια σύντομη επίδειξη της εκτέλεσης των μεθόδων του συμβολαίου με τη βοήθεια του Remix (κατά την καταγραφή, για πιο γρήγορη απόκριση, το συμβόλαιο μπορεί να είναι δημοσιευμένο τοπικά αν θέλετε). Για την καταγραφή της οθόνης σας σε βίντεο μπορείτε να χρησιμοποιήσετε κάποιο εμπορικό λογισμικό όπως το Camtasia, ή κάποιο από τα δωρεάν active presenter, screencastify και screencast-o-matic, ή τέλος, σε υπολογιστή με Windows,

τον συνδυασμό πλήκτρων Win+G. Φροντίστε να υπάρχει φωνητική περιγραφή της κάθε ενέργειας.

Παραδώστε στο openeclass ένα αρχείο *.zip που θα περιέχει **α)** το αρχείο ***.sol** με τον ολοκληρωμένο πηγαίο κώδικα και **β)** ένα **pdf** με **1)** τα στοιχεία σας **2)** τη διεύθυνση URL όπου δημοσιεύσατε το screencast (κατά προτίμηση στο YouTube), **3)** τη διεύθυνση της δοκιμαστικής blockchain (με το όνομά της) όπου δημοσιεύσατε το συμβόλαιο **4)** λίστα με όλα τα ζητήματα αριθμημένα και την αυτοαξιολόγηση του καθενός. Θα αναφέρετε ποιες είναι οι όποιες ελλείψεις π.χ. «δεν λειτουργεί σωστά η κλήρωση» κι όχι αόριστα «κατάφερα περίπου το 70% του ζητήματος».

Το αρχείο zip ΠΡΕΠΕΙ να έχει όνομα ίδιο με τον αριθμό μητρώου σας π.χ. **xxx32999**.zip

Το ζήτημα 9 είναι απολύτως απαραίτητο. Αποτελεί το παραδοτέο. Χωρίς το ζήτημα 9 δεν θα βαθμολογηθείτε. Ωστόσο, η ποιότητά του και συνεπώς η συμβολή του στην αξιολόγηση θα εκτιμηθεί με **(10%)**