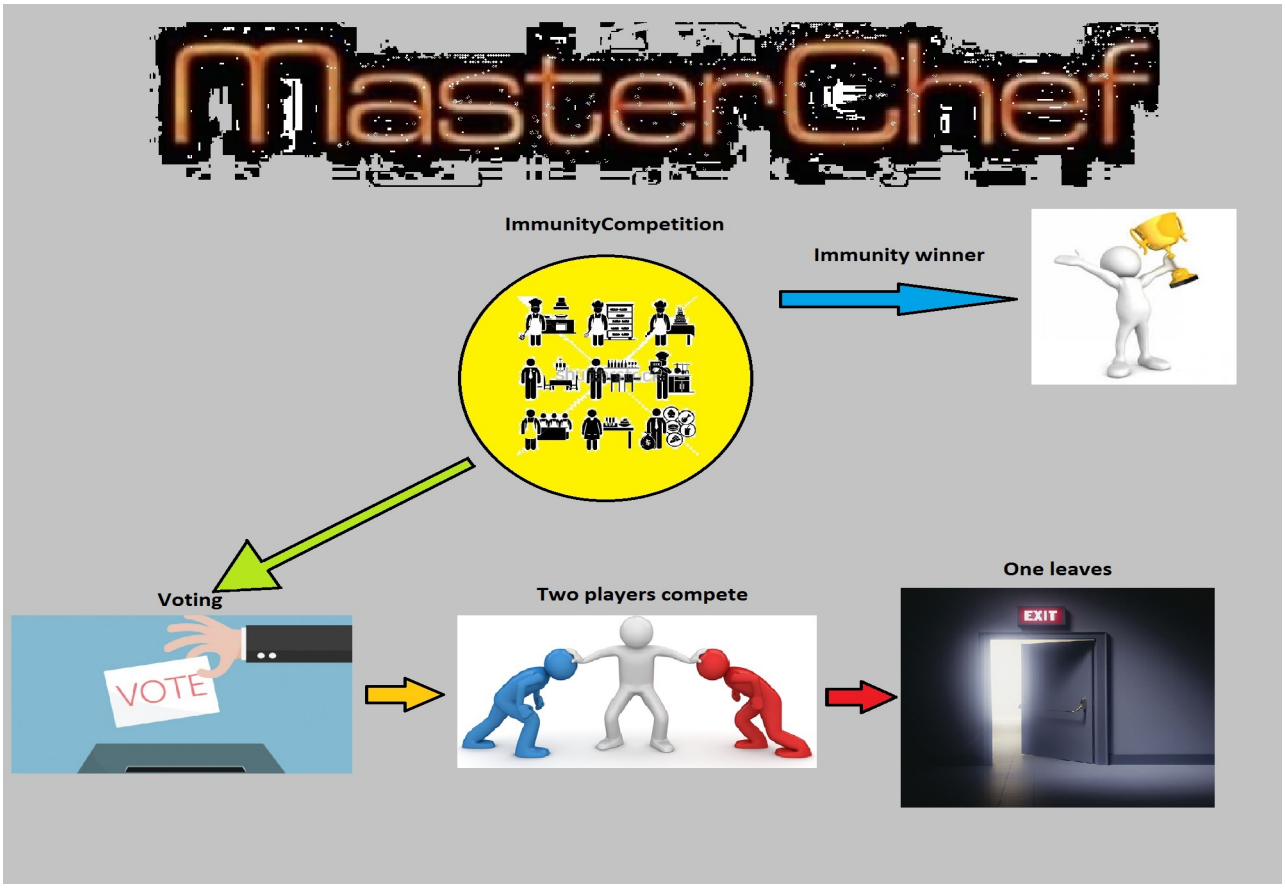
Εργασία 4
Αντικειμενοστραφής προγραμματισμός
Master Chef Ψηφοφορία

Φοιτητές :
Ελένη Παπαχριστοδούλου ΑΕΜ:9580
Μπεκιάρης Θεοφάνης ΑΕΜ:8200



Σκοπός:
Στόχος της εργασίας είναι η κατασκευή των κατάλληλων κλάσεων και συναρτήσεων για την αναπαράσταση της διαδικασίας της ψηφοφορίας του παιχνιδιού. Συγκεκριμένα πρέπει να κατασκευάσουμε την κλάση **Vote** που είναι υπεύθυνη για την δημιουργία αντικείμενων ψήφου και την κλάση **Voting** η οποία περιέχει static μεταβλητές και συναρτήσεις μέλη και δεν δημιουργεί αντικείμενα η οποία περιέχει την συνάρτηση **static void votingProcess(Team &team)** η οποία προσομοιώνει την διαδικασία ψηφοφορίας. Αρχικά εκτελείται ο διαγωνισμός ασυλίας από τον οποίο προκύπτει ένας νικητής με επιπλέον προνόμια στην διαδικασία ψηφοφορίας. Συγκριμένα έχει περισσότερους ψήφους από τους υπόλοιπους παίκτες για να δώσει και επιπλέον δεν μπορεί να ψηφιστεί. Στην συνέχεια εκτελούμε τη διαδικασία ψηφοφορίας από την οποία προκύπτουν δύο υποψήφιοι από τους οποίους ο παίκτης με την μικρότερη τεχνική κατάρτιση καλείται να αποχωρήσει από το παιχνίδι Master Chef.

Ο κώδικας για την κατασκευή των κλάσεων που ζητείται είναι απλός,αφού έχουμε απλά δημιουργία δύο κλάσεων με μεταβλητές και συναρτήσεις όπως ζητούνται στην εκφώνηση,επομένως δεν απαιτείται περαιτέρω ανάλυση. Παρακάτω θα αναλυθεί μόνο ο κώδικας της συνάρτησης **static void votingProcess(Team &team)** η οποία αντιστοιχεί στην διαδικασία με την οποία ψηφίζονται οι παίκτες και επομένως ίσως να μην είναι κατανοητός ο τρόπος λειτουργίας απλά από την ανάγνωση του κώδικα.

Συνάρτηση static void votingProcess(Team &team):
Η συνάρτηση προσομοιώνει την διαδικασία της ψηφοφορίας,καταμέτρησης των ψήφων,εύρεσης των δύο παικτών με τις περισσότερους ψήφους,και επιλογής εκείνου του παίκτης με την χειρότερη τεχνική κατάρτιση ο οποίος τελικά διαγράφεται από την λίστα της ομάδας του και αποχωρεί από το παιχνίδι. Στην συνάρτηση έχουμε σπάσει τις παραπάνω ενέργειες και τις έχει κατανείμει σε συναρτήσεις από της οποίες η κάθε μία αναλαμβάνει να διεκπεραιώσει μία από τις παραπάνω λειτουργίες,επομένως έχουμε.

void takeVotes(Player *players,vector<Vote> &votes): Η συνάρτηση προσομοιώνει την διαδικασία ψηφοφορία στην οποία κάθε παίκτης ψηφίζει κάποιον από τους συμπαίκτες του. Ο παίκτης με ασυλία δεν μπορεί να ψηφιστεί και επιπλέον έχει δικαίωμα να ψηφίζει 2 ή 3 συμπαίκτες του. Η συνάρτηση τοποθετεί σε ένα διάνυσμα αντικείμενα Vote τα οποία περιέχουν το όνομα του παίκτη και τον λόγο για τον οποίο ψηφίζεται ο παίκτης. Κατά την διαδικασία, ο κάθε παίκτης ανάλογα με τον αριθμό ψήφων που έχει(1 για όλους εκτός από αυτόν με ασυλία που έχει 2 ή 3) ο οποίος περνάει στην μεταβλητή votesNum θα πρέπει να επιλέξει να ψηφίσει κάποιον ο οποίος δεν θα είναι προφανώς ο εαυτός του,θα υπάρχει στην λίστα της ομάδας,δεν θα είναι ο παίκτης με ασυλία και επιπλέον δεν θα έχει ξαναψηφιστεί στην ίδια ψηφοφορία από τον ίδιο παίκτης(πίνακας prevVotes ο οποίος απομνημονεύει τις ψήφους ενός παίκτη),δηλαδή δεν θα έχουμε πολλαπλές ψήφους στον ίδιο παίκτη. Για τους παραπάνω περιορισμούς με την σειρά που αναφέρθηκαν έχουμε στον κώδικα:

```
if(index!=i && players[index].getAge()!=0 && players[index].getImmunity() == false
    && index != prevVotes[0] && index != prevVotes[1] && index != prevVotes[2])
```

Αν λοιπόν η ικανοποιούνται οι προδιαγραφές μια ψήφος (αντικείμενο Vote) με το όνομα του παίκτη που ψηφίστηκε μπαίνει στο διάνυσμα των ψήφων.

int voteSomeone(): Η συνάρτηση είναι υπεύθυνη για τον τρόπο με τον οποίο επιλέγεται ένα παίκτης να ψηφιστεί. Εδώ έχουμε ορίζει απλά να επιλέγεται τυχαία ένα παίκτης.

void votesCounting(Player *players,vector<Vote> &votes,map<string,int> &results):
Η συνάρτηση λαμβάνει αναφορές στο διάνυσμα του περιέχει τις ψήφους και ενός χάρτη. Η συνάρτηση κάνει καταμέτρηση των ψήφων που πείρε ο κάθε παίκτης από τους συμπαίκτες του και τοποθετεί στον χάρτη κλειδί με το όνομα του κάθε παίκτη και με μεταβλητή τον αριθμό ψήφων που πήρε.

void findCandidates(Player *players,map<string,int> &results): Η συνάρτηση λαμβάνει όρισμα τον προηγούμενο χάρτη από τον οποίο βρίσκει τους δύο παίκτες με τις περισσότερες ψήφους. Στην περίπτωση που πολλοί παίκτες έχουν ισοψηφία ψήφων επιλέγεται τυχαία ένας από αυτούς ως υποψήφιος για αποχώρηση. Στο τέλος η συνάρτηση δηλώνει τους δύο υποψήφιους για αποχώρηση θέτοντας τις μεταβλητές Candidate των αντικειμένων των παικτών στην λογική τιμή

true. Τέλος αρχικοποιούμε τις μεταβλητές του παίκτη ασυλίας,αρχικοποιούμε δηλαδή τον αριθμό των ψήφων του στην τιμή 1 και την κατάσταση της μεταβλητής Immunity στην τιμή false;

void removePlayer(Team &team): Η συνάρτηση αρχικά ψάχνει στην λίστα παικτών της ομάδας τους δύο υποψήφιους παίκτες (μέσω της μεταβλητής candidate = true) και διαγράφει από την λίστα τον παίκτη με την χειρότερη τεχνική κατάρτιση. Επιπλέον αρχικοποιεί την μεταβλητή candidate του παίκτη που μένει στην λογική τιμή false αφού πλέον δεν είναι υποψήφιος.

Επομένως η δομή της συνάρτησης **static void votingProcess(Team &team)** καλώντας τις παραπάνω συναρτήσεις είναι:

- Κατάθεση ψήφων από τους παίκτες
- Εκτύπωση ψήφων και λόγου ψήφου
- Καταμέτρηση ψήφων και αντιστοίχηση παίκτη - αριθμού ψήφων
- Εκτύπωση όνομα παίκτη με αριθμό ψήφων που πείρε
- Εύρεση δύο υποψήφιων για αποχώρηση
- Εκτύπωση υποψηφίων
- Εύρεση παίκτη με την χειρότερη τεχνική κατάρτιση για διαγραφή από το παιχνίδι
- Εκτύπωση παίκτη που αποχώρησε
- Αρχικοποίηση δυναμικών δομών(διάνυσμα,χάρτης)