3η Εργασία OOP
MasterChef PART III:
The Return of the Chef

Βέλλιος Γεώργιος-Σεραφείμ

ΑΕΜ: 9471

velliosg@ece.auth.gr

Μανούσος Διαγόρας

ΑΕΜ: 9554

dmanouso@ece.auth.gr

Περιγραφή του προβλήματος:

Καλούμαστε να υλοποιήσουμε την βασική κλάση του διαγωνισμού (*Competition*) και τις παραγόμενες κλάσεις *TeamCompetition*, *ImmunityCompetition* και *CreativityCompetition* που κληρονομούν τα γνωρίσματα της βασικής κλάσης. Στην συνέχεια, χρησιμοποιώντας αυτές τις κλάσεις και τις μεθόδους και μεταβλητές τους, όπως και τις κλάσεις που μας δίνονται έτοιμες, καλούμαστε να υλοποιήσουμε τις 4 συναρτήσεις στο αρχείο *main.cpp* που αντιστοιχούν στις τέσσερις μέρες του παιχνιδιού. Αυτές είναι οι :

-Κανονική μέρα, όπου δεν γίνεται κάποιος διαγωνισμός

-Ημέρα Ομαδικού Διαγωνισμού, (γίνεται ένας TeamCompetition)

-Ημέρα διαγωνισμού ασυλίας (γίνεται ένας ImmunityCompetition)

-Ημέρα διαγωνισμού δημιουργικότητας, όπου γίνεται ένας CreativityCompetition.

Αλγόριθμος και διαδικασίες υλοποίησης του :

Αρχικά δημιουργήθηκαν οι κλάσεις *Competition*, *TeamCompetition*, *ImmunityCompetition* και *CreativityCompetition.*

Πληροφορίες για τις κλάσεις αναφέρονται στα αντίστοιχα αρχεία **.h** και πληροφορίες για τις μεθόδους τους και τον τρόπο λειτουργίας αυτών (εκτός των μεθόδων ***compete()*** στις οποίες θα γίνει αναφορά παρακάτω) αναφέρονται στα αρχεία **.cpp**. Επιλεκτικά, αναφέρουμε πως:

- **Η κλάση Compete** είναι βασική και οι υπόλοιπες 3 παράγωγές της όποτε χρησιμοποιούν στους *constructor* τους με ορίσματα την σύνταξη: **συνάρτηση_αρχικών_συνθηκών(λίστα_ορισμάτων): Competition(ορίσματα){ λοιπές_εντολές; }.**
- Στις παράγωγες κλάσεις η μεταβλητή …….Award δεν έχει *setters* και *getters* καθώς αυτοί υπάρχουν ήδη στα αρχεία **.h** που μας έχουν δοθεί.
- **Στην κλάση CreativityCompetition** υπάρχει ο στατικός πίνακας *ingredients* που είναι κοινός για όλα τα αντικείμενα της κλάσης. Επίσης, ο πίνακας μπορεί να καλεστεί από άλλες συναρτήσεις χωρίς να γίνει αναφορά σε συγκεκριμένο αντικείμενο.

<u>Τρόπος λειτουργίας των μεθόδων compete():</u>

Στην κλάση TeamCompetition: **Int compete(Team&team1,Team&team2)**

Δέχεται ως ορίσματα αναφορές στις δύο ομάδες (Μπλε και Κόκκινη) και επιστρέφει 1 αν χάνει η ομάδα 2 και 0 αν χάνει η ομάδα 1.

Αρχικά δηλώνονται οι μεταβλητές *wins1,wins2* που αναφέρονται στις νίκες της κάθε ομάδας σε κάθε γύρο και η *numOfRounds* που δείχνει τον αριθμό του τρέχοντος γύρου. Ύστερα, ξεκινάει μία *for* που επαναλαμβάνεται 3 φορές, όσες και οι γύροι. Μετά δημιουργούνται δύο πίνακες *int mix1, mix2* που έχουν ο καθένας τους αριθμούς 0 έως 10 και με τον αλγόριθμο:

```
int mix1[11]={0, 1,2,3,4,5,6,7,8,9,10};
int swap;
for (int i = 0; i <11; i++) {
    int j = rand() % 11;
    swap = mix1[j];
    mix1[j] = mix1[i];
    mix1[i] = swap;
}
```

Εικόνα 1: Η δημιουργία του πίνακα mix1

```
   int mix2[11]={0,1,2,3,4,5,6,7,8,9,10};
 for (int i = 0; i <11; i++) {
    int j = rand() % 11;
    swap = mix2[j];
    mix2[j] = mix2[i];
    mix2[i] = swap;
}
```

Εικόνα 2: Η δημιουργία του πίνακα mix2

Αλλάζει τυχαία η σειρά των στοιχείων στον πίνακα αλλά συνεχίζει να εμφανίζεται το κάθε στοιχείο μόνο μία φορά. Αυτό χρειάζεται όταν διαλέγονται τυχαία 5 παίκτες από κάθε ομάδα. Μετά ορίζονται οι μεταβλητές *technique1* και 2 και *exhaustion1* και 2 που είναι η συνολική τεχνική κατάρτιση και κούραση αντίστοιχα των 2 ομάδων. Κατόπιν εκτελείται μία forγια 5 επαναλήψεις(όσοι και οι παίκτες της κάθε ομάδας), που , με χρήση των πινάκων *mix1, mix2* διαλέγει 5 τυχαίους παίκτες από κάθε ομάδα, καλεί την μέθοδο *<u>.compete()</u>* για τους παίκτες και αυξάνει την τεχνική κατάρτιση και τη κούραση των ομάδων. Μετά, επιστρέφοντας στην εξωτερική *for*, γίνεται έλεγχος σύμφωνα με τον ψευδοκώδικα που μας δόθηκε. Έτσι ελέγχεται ποια ομάδα νίκησε στον γύρο. Αυτής της ομάδας αυξάνεται η αντίστοιχη μεταβλητή *wins1 ή wins2* κατά ένα , δίνεται το χρώμα της ως τιμή στην μεταβλητή *winner* του *<u>rounds[]</u>*και εκτυπώνονται τα στοιχεία του γύρου. Εδώ τελειώνει η εξωτερική *for*. Τώρα γίνεται έλεγχος ποια από τις δύο ομάδες είχε τις περισσότερες νίκες στους 3

γύρους. Αυτή που έχει τουλάχιστον 2 νικάει. Αυτής της ομάδας αυξάνεται η μεταβλητή *wins* κατά 1 με την εντολή *team.setWins(team.getWins()+1);* Και αυξάνεται η μεταβλητή *supplies* κατά το ποσό που ορίζει το *Bonus* του *FoodAward* (η διαδικασία γίνεται με τους κατάλληλους *setters* και *getters* και με μια προσωρινή μεταβλητή *float temp* για να είναι πιο ευανάγνωστος ο κώδικας. Εκτυπώνεται το χρώμα της νικήτριας ομάδας και δίνεται στην μεταβλητή *winner* της *TeamCompetition* το χρώμα της νικήτριας ομάδας. Τέλος, επιστρέφεται 0 ή 1.

Στην κλάση ImmunityCompetition: **Void compete(Team &team);**

Στην ***compete()*** αυτή διαγωνίζεται μία ομάδα και νικητής είναι ο παίκτης που έχει τον καλύτερο συνδυασμό **(0.75*technique + 0.25*(100-fatigue)).** Αρχικά, δημιουργείται μία μεταβλητή *float combination* που θα κρατάει τις τιμές του παραπάνω συνδυασμού για τον κάθε παίκτη:

- οι μεταβλητές *temp1, temp2* που χρησιμοποιούνται για τον υπολογισμό του *combination,*
- *winning* που είναι η τιμή του μεγαλύτερου συνδυασμού που έχει μέχρι κάποιο σημείο της εκτέλεσης βρεθεί και
- *numWinner* που είναι η θέση του παίκτη με τον μεγαλύτερο συνδυασμό.

Στην συνέχεια, υπάρχει μία *for* που εκτελείται 11 φορές, μία για κάθε παίκτη, και υπολογίζει τον συνδυασμό του παίκτη ενώ παράλληλα συγκρίνει αυτόν τον συνδυασμό με τον μεγαλύτερο μέχρι τώρα (*winning*) συνδυασμό και, αν είναι μεγαλύτερος δίνει την τιμή του στην μεταβλητή *winning* και την θέση του παίκτη στη μεταβλητή *numWinner*. Μετά το τέλος της *for* έχει βρεθεί ο νικητής, οπότε εκτυπώνονται τα στοιχεία του, αυξάνονται οι νίκες του κατά μία και δίνεται στην μεταβλητή *winner* **της κλάσης ImmunityCompetition** το όνομα του παίκτη που νίκησε.

CreativityCompetition:

Νικητής είναι ο παίκτης από τις δύο ομάδες που έχει την μεγαλύτερη τεχνική κατάρτιση. Όπως και προηγουμένως, ορίζονται αρχικά οι μεταβλητές:

1. *technique* που θα κρατάει την τεχνική κατάρτιση του κάθε παίκτη για όσο χρόνο αυτή χρειάζεται,
2. *maxTechnique* (μέγιστη τεχνική),
3. *maxIndex* (θέση του παίκτη με την μέγιστη τεχνική),
4. *playerIndex* (θέση του κάθε παίκτη).

Δημιουργείται ένας δυναμικός πίνακας *players [22]* όπου τοποθετούνται με δύο *for* οι παίκτες και από τις δύο ομάδες. Με μία *while* που εκτελείται 22 φορές

ελέγχεται αν η τεχνική κατάρτιση του καθενός από τους 22 παίκτες **(technique=players[playerIndex].getTechnique())** είναι μεγαλύτερη από την μέγιστη τεχνική κατάρτιση (*maxTechnique*) που έχει βρεθεί μέχρι τώρα. Αν αυτό ισχύει, ο αριθμός του παίκτη αυτού δίνεται στην μεταβλητή *maxIndex* και η τεχνική του κατάρτιση στη *maxTechnique*. Μόλις τελειώσει η *while* εκτυπώνονται τα στοιχεία του νικητή. Επίσης, με τον κώδικα:

```
temp= players[maxIndex].getTechnique();
temp += excursionAward.getTechniqueBonus();
players[maxIndex].setTechnique(temp);
```

Εικόνα 3: Η αύξηση της τεχνικής κατάρτισης του νικητή

Αυξάνεται η τεχνική κατάρτιση του νικητή κατά το ποσό που ορίζεται στην *κλάση ExcursionAward (+5)* και με τον κώδικα:

```
temp= players[maxIndex].getPopularity();
temp += excursionAward.getPopularityPenalty();
players[maxIndex].setPopularity(temp);
```

Εικόνα 4: Η μείωση της δημοφιλίας του νικητή

Μειώνεται η δημοφιλία του νικητή κατά το ποσό που ορίζεται στην *κλάση ExcursionAward (-10)* . Επίσης αυξάνονται οι νίκες του παίκτη κατά μία. Τέλος, δίνεται στην μεταβλητή *winner* του *CreativityCompetition* το όνομα του παίκτη που νίκησε.

main.cpp:

Πρωταρχικά, στην main ξεκινάει μια γεννήτρια ψευδοτυχαίων αριθμών που θα χρειαστεί για πολλές από τις μεθόδους των κλάσεων. Επίσης δημιουργείται μια μεταβλητή *int xamenos* που κρατάει την τιμή της ομάδας που έχασε στον *teamCompetition*. Μετά έγιναν οι επιμέρους συναρτήσεις:

Void normalDay():

Όπως λέει το κείμενο, το πρωί οι δύο ομάδες εργάζονται και τρώνε και το βράδυ τρώνε, κοινωνικοποιούνται και κοιμούνται:

```
void normalDay()
{

    cout << endl << "This is a normal day in the Master Chef Game." << endl << endl;
    cout<<"Morning\n";
    teams[0].teamWorks();
    teams[1].teamWorks();
    teams[0].teamEats();
    teams[1].teamEats();
    cout<<"Night\n";
    teams[0].teamEats();
    teams[1].teamEats();
    teams[0].teamSocializes();
    teams[1].teamSocializes();
    teams[0].teamSleeps();
    teams[1].teamSleeps();

}
```

Εικόνα 5: Η συνάρτηση void normalDay()

Οι συναρτήσεις αυτές επηρεάζουν τις μεταβλητές των παικτών των δύο ομάδων. Τα παραπάνω τμήματα κώδικα είναι κοινά για όλες τις μέρες και για αυτό δεν θα αναφερθούν ξανά.

Void teamCompetitionDay():

Ίδια ρουτίνα το πρωί και το βράδυ. Το μεσημέρι πραγματοποιείται ένας ομαδικός διαγωνισμός. Έτσι, δημιουργείται δυναμικά ένα αντικείμενο **vraveio* τύπου *FoodAward* που θα αποτελέσει όρισμα για τον *diagonismos* (αντικείμενο της **κλάσης TeamCompetition** που είναι δυναμικό και δέχεται για ορίσματα την καθολική μεταβλητή *competitionId* που γίνεται το *id* του διαγωνισμού, το *string TeamComp* ως όνομα του διαγωνισμού και το *FoodAward*vraveio*. Μετά εκτελείται η συνάρτηση *int compete()* της κλάσης *TeamCompetition* που επιστρέφει την τιμή του *int* στην μεταβλητή *xamenos*. Τέλος, διαγράφονται τα δυναμικά στοιχεία και αυξάνεται το *competitionId* κατά 1 ώστε κάθε ένας διαγωνισμός μετά να έχει έναν μοναδικό αριθμό *id* που θα είναι αύξοντας.

Void immunityCompetitionDay():

Ίδια ρουτίνα το πρωί και το βράδυ. Το μεσημέρι πραγματοποιείται ένας διαγωνισμός ασυλίας. Έτσι, δημιουργείται δυναμικά ένα αντικείμενο **vraveio* τύπου *ImmunityAward* που θα αποτελέσει όρισμα για τον *diagonismos* (αντικείμενο της **κλάσης ImmunityCompetition** που είναι δυναμικό και δέχεται για ορίσματα την καθολική μεταβλητή *competitionId* που γίνεται το *id* του διαγωνισμού, το *string AsiliasDiag* ως όνομα του διαγωνισμού και το *ImmunityAward *vraveio*. Μετά εκτελείται η συνάρτηση ***compete()*** της **κλάσης ImmunityCompetition** για την ομάδα που έχασε σε προηγούμενο ομαδικό

διαγωνισμό . Τέλος, διαγράφονται τα δυναμικά στοιχεία και αυξάνεται το *competitionId* κατά 1.

Void creativityCompetitionDay():

Ίδια ρουτίνα το πρωί και το βράδυ. Το μεσημέρι πραγματοποιείται ένας *CreativityCompetition*. Έτσι, δημιουργείται δυναμικά ένα αντικείμενο **vraveio* τύπου *ExcursionAward* που θα αποτελέσει όρισμα για τον *diagonismos* (αντικείμενο της **κλάσης CreativityCompetition** που είναι δυναμικό και δέχεται για ορίσματα την καθολική μεταβλητή *competitionId* που γίνεται το *id* του διαγωνισμού, το *string EkdromiDia* ως όνομα του διαγωνισμού και το *ExcursionAward *vraveio*. Μετά εκτελείται η συνάρτηση **compete()** της κλάσης. Τέλος, διαγράφονται τα δυναμικά στοιχεία και αυξάνεται το *competitionId* κατά ένα.

Έλεγχος ορθότητας:

Ο έλεγχος της ορθής λειτουργίας του προγράμματος έγινε με την εκτέλεση του αρκετές φορές ώστε να βεβαιωθούμε ότι λειτουργεί ως πρέπει. Επίσης, καθώς με μόνο την εκτέλεσή του δεν μπορούσαμε να γνωρίζουμε με ακρίβεια αν εκτελούνται όλες του οι λειτουργίες καθώς τα αποτελέσματα πολλών δεν εμφανίζονται. Για αυτό το λόγο, σε αρκετά σημεία του κώδικα προστέθηκαν εντολές, κυρίως εξόδου στην κονσόλα, για να ελεγχθούν και αυτές οι μεταβλητές. Πραγματοποιήθηκαν αρκετές εκτελέσεις του προγράμματος και με αυτές τις εντολές. Ωστόσο, στο παραδοτέο πρόγραμμα, αυτές οι εντολές βρίσκονται μέσα σε σχόλια ώστε να είναι πιο ευπαρουσίαστο το υπόλοιπο πρόγραμμα.