Προαιρετική εργασία στη C++

**Α.Ε.Μ:** 2513

E-mail: [prodpoly@csd.auth.gr](mailto:prodpoly@csd.auth.gr)

Ονοματεπώνυμο: Πρόδρομος Πολυχρονιάδης

Διάγραμμα Κλάσεων:



Οι κλάσεις που δημιούργησα είναι οι εξής παραπάνω.

* Class movies: Η κλάση αυτή δημιουργήθηκε με σκοπό το χειρισμό των ταινιών καθ' όλη τη διάρκεια της εφαρμογής. Με τη δημιουργία αντικειμένων της κλάσης αυτής έγινε δυνατή η υλοποίηση όλων των λειτουργιών που απαιτούσαν την αλληλεπίδραση με τις ταινίες του Video Club. Κληροδοτείται στην κλάση series.
* Class series: H κλάση αυτή δημιουργήθηκε με σκοπό το χειρισμό των σειρών με τις οποίες χρειαζόταν να ασχοληθεί ο χρήστης. Κληρονομεί την κλάση movies καθώς έχουν πάρα πολλά κοινά (π.χ. κοινές μεταβλητές ίδιας χρησιμότητας, κοινές συναρτήσεις) και ελάχιστες διαφορές.
* Class video\_games: Η κλάση αυτή δημιουργήθηκε για το χειρισμό των παιχνιδιών του Video Club από τον χρήστη. Και αυτή έχει κάποια κοινά με τη movies αλλά όχι τόσα όσα η series και πολλές μεταβλήτες και συναρτήσεις θα ήταν άχρηστες για τη video\_games και γενικότερα, σαν ιδεά και λογική δε θα μου άρεζε να εβάζα τη video\_games να κληρονομεί τη movies ή την series διότι τα video games είναι κάτι τελείως διαφορετικό από τις ταινίες και τις σειρές που μοιάζουν μεταξύ τους.
* Class client\_info: Η κλάση αυτή δημιουργήθηκε με σκοπό τον χειρισμό των πληροφοριών σχετικά με τους πελάτες του Video Club. Κληροδοτείται στην κλάση VIP\_client\_info.
* Class VIP\_client\_info: Η κλάση αυτή αναφέρεται στους VIP πελάτες του Video Club και κληρονομεί τα στοιχεία της client\_info καθώς είναι όλα κοινά, δηλαδή δεν της είναι κανένα άχρηστο απλώς έχει και από μόνη της κάποιες επιπλέον πληροφορίες (μεταβλητές-συναρτήσεις)
* Class rent\_info: Η κλάση αυτή δημιουργήθηκε με σκοπό το χειρισμό των δανεισμών των ταινιών, σειρών και παιχνιδιών του Video Club. Περιέχει τις απαραίτητες μεταβλητές και συναρτήσεις για την ενοικιασή ταινίων. Κάθε αντικείμενο της είναι ένας ξεχωριστός δανεισμός.

Οι συναρτήσεις των κλάσεων στις οποίες αναφέρθηκα παραπάνω είναι οι Getters και οι Setters (και οι συναρτήσεις Copy που εφτίαξα που είναι ουσιαστικά Setters) των μεταβλητών. Αυτές τις μεθόδους τις υλοποίησα στο header file κάθε κλάσης και όχι στο .cpp καθώς ήταν πολύ μικρές σε γραμμές κώδικα.

Όσον αφορά τη σχεδίαση της εφαρμογής, αυτή βασίστηκε πολύ στους vectors. Με αυτούς δημιούργησα δυναμικούς πίνακες αντικειμένων της κάθε κλάσης. Για κάθε κλάση δημιούργησα και από έναν vector (π.χ. vector<movies> MoviesVector). Παρακάτω θα παρουσιάσω τις συναρτήσεις του προγράμματός μου και το πως χειρίστηκα τα αντικείμενα των κλάσεων. Θα γράφω δίπλα από κάθε συνάρτηση το σημείο στο οποίο βρίσκεται στον κώδικα για τη διευκόλυνση εύρεσης τους.

* **int main() [γραμμή 1785]**: Η main που δημιούργησα έχει το ρόλο του "οδηγού" της εφαρμογής. Αυτή παρουσιάζει τις επιλογές της εφαρμογής (μέσω της συνάρτησης menu() ) στο χρήστη ώστε αυτός να πραγματοποιήσει ό,τι επιθυμεί. Ο χρήστης επιλέγει την κατάλληλη λειτουργία και η main στη συνέχεια καλεί την κατάλληλη ή τις κατάλληλες συναρτήσεις για την υλοποίηση της λειτουργίας. Επιπλέον στη main δηλώνονται οι vectors των συναρτήσεων.
* **int menu() [γραμμή 1761]:** Παρουσιάζει το μενού επιλογών στο χρήστη και επιστρέφει την επιλογή του στη main.
* **void load\_msvg(MoviesVector, SeriesVector, GamesVector, ap) [γραμμή 476]:** Η συνάρτηση αυτή δέχεται από τη main τους vectors των ταινιών, των σειρών, των παιχνιδιών καθώς και την επιλογή του χρήστη όσων αφορά τη λειτουργία που θέλει ο χρήστης να εκτελέσει το πρόγραμμα. Όπως προδίδει και το όνομα της συνάρτησης, αυτή φορτώνει στους vectors όλες τις υπάρχουσες ταινίες, σειρές και παιχνίδια που βρίσκονται στα αρχεία movies.txt, series.txt και video\_games.txt αντίστοιχα. Αυτό το καταφέρνει με τον εξής τρόπο: Κάθε γραμμή των αρχείων αντιπροσωπεύει και μία μεταβλητή των αντικειμένων των κλάσεων. Έτσι, όσο το κάθε αρχείο δεν έχει τελειώσει, η συνάρτηση δημιουργεί ένα αντικείμενο της αντίστοιχης κλάσης, διαβάζει γραμμή-γραμμή το αρχείο και αυτά που διαβάζει τα θέτει στις μεταβλητές του αντικειμένου με τις συναρτήσεις Copy. Όταν έχουν συμπληρωθεί όλες οι μεταβλητές του αντικειμένου, αυτό εισέρχεται στον αντίστοιχο vector. Επομένως και τελειώνει η πρώτη επανάληψη. Αν όμως π.χ. υπάρχει και δεύτερη ταινία τότε ξανά φτιάχνοντας ένα νέο αντικείμενο, αυτό προστίθεται στον vector, αφού έχουν συμπληρωθεί οι μεταβλητές του (το διάβασμα του αρχείου στη δεύτερη επανάληψη συνεχίζει από εκεί που τελείωσαν οι πληροφορίες της πρώτης ταινίας και αρχίζουν της δεύτερης). Στο μεταξύ το size του vector έχει γίνει 2 αφού είναι δυναμικός. Και αυτό πραγματοποιείται και για τις τρεις κλάσεις (movies,series,video\_games) ώστε να έχουμε όλες τις ταινίες, σειρές, παιχνίδια στους αντίστοιχους vectors. Κάτι ακόμα που πραγματοποιεί αυτή η συνάρτηση είναι να καλέσει την addMovie(MoviesVector), addSeries(SeriesVector) και addGame(GamesVector) σε περίπτωση που ο χρήστης το επιθυμεί.
* **void addMovie(MoviesVector) [γραμμή 438], Void addSeries(SeriesVector) [γραμμή 51], void addGame(GamesVector) [γραμμή 16]:** Και οι τρεις αυτές συναρτήσεις έχουν ως σκοπό το να προσθέσουν μια νέα ταινία ή σειρά ή παιχνίδι αντίστοιχα. Ο τρόπος υλοποίησης είναι πολύ απλός και αυτός γίνεται μέσω των Setters που υλοποίησα. Ουσιαστικά δημιουργείται ένα νέο αντικείμενο, του δίνω τιμές με τους Setters, το βάζω στον αντίστοιχο vector και στη συνέχεια το γράφω στο αντίστοιχο αρχείο. Επίσης, έχω δημιουργήσει κάποια αρχεία, τα number\_of\_m.txt, number\_of\_s.txt, number\_of\_vg.txt, τα οποία κρατάνε των αριθμό των ταινιών, των σειρών και των παιχνιδιών, ο οποίος ενημερώνεται από αυτές τις συναρτήσεις αφού πρέπει να αυξηθεί. Γνωρίζω πως δε μας ζητήσατε να κρατιέται κάπου αυτός ο αριθμός απλώς σκέφτηκα πως σε ένα Video Club θα ήθελε ο υπάλληλος ή ο ιδιοκτήτης να γνωρίζει τον αριθμό αυτόν για διάφορους λόγους. Είναι κάτι miscellaneous αλλά ήταν έμπνευση της στιγμής και το έβαλα.
* **void deleteMovie(MoviesVector) [γραμμή 94], void deleteSeries(SeriesVector) [γραμμή 138], void deleteVGame(GamesVector) [γραμμή 182]:** Και οι τρεις συναρτήσεις υλοποιούν τη διαγραφή ταινιών, σειρών και παιχνιδιών αντίστοιχα. Στην αρχή ζητείται από τον χρήστη το ID του συγκεκριμένου προϊόντος που θέλει να διαγραφεί και αφού αυτό βρεθεί μέσα στον αντίστοιχο vector τότε αυτό αφαιρείται από τον vector και ξαναγράφεται ολόκληρος ο vector στο αντίστοιχο αρχείο όπου τώρα όμως θα λείπει το συγκεκριμένο στοιχείο που διαγράφηκε. Παράλληλα, ενημερόνωνται και τα αρχεία που κρατάνε τους αριθμούς από τις ταινίες κλπ του Video Club.
* **void modifyMovie(MoviesVector) [γραμμή 312], void modifySeries(SeriesVector) [γραμμή 237], void modifyVGame(GamesVector) [γραμμή 381]:** Και οι τρεις συναρτήσεις υλοποιούν την επεξεργασία ταινιών, σειρών και παιχνιδιών αντίστοιχα. Δέχονται τους αντίστοιχους vectors οι οποίοι είναι φορτωμένοι μέσω της load\_msvg() με τα αντικείμενα των κλάσεων και ζητείται από τον χρήστη το τι θέλει να τροποποιήσει (μέσω της modifyMenu() και modifyMenu\_vg() ) και σε ποιο αντικείμενο θέλει να το τροποποιήσει. Αφού αυτό το αντικείμενο βρεθεί μέσα στον αντίστοιχο vector τότε τροποποιείται η μεταβλητή που επιθυμεί ο χρήστης. Στη συνέχεια, γράφεται στο αντίστοιχο αρχείο ξανά ολόκληρος ο vector με το τροποποιημένο αντικείμενο.
* **void modifyMenu() [γραμμή 224], void modifyMenu\_vg() [γραμμή 370]:** Οι συναρτήσεις αυτές παρουσιάζουν στον χρήστη το μενου των επιλογών τους σχετικά με την τροποποιήση των ταινιών, των σειρών και των παιχνιδιών. Η πρώτη παρουσιάζει τις επιλογές για τις ταινίες και τις σειρές ενώ η δεύτερη για τα παιχνίδια.
* **void load\_clients(ClientsVector, VIPClientsVector, ap) [γραμμή 1405]:** Η συνάρτηση αυτή έχει ακριβώς την ίδια λειτουργία με την load\_msvg απλώς αυτή αναφέρεται στους πελάτες του Video Club. Έτσι, αυτή φορτώνει τους vectors των απλών πελατών και των VIP με τα αντικείμενα της αντίστοιχης κλάσης αφού τα δημιουργήσει με τη βοήθεια των αρχείων clients.txt και VIP\_clients.txt. Καλεί την addClient(ClientsVector) και την addVIP\_client(VIPClientsVector) αν αυτή είναι η επιλογή (ap) του χρήστη.
* **void addClient(ClientsVector) [γραμμή 1325], void addVIP\_client(VIPClientsVector) [γραμμή 1362]:** Οι συναρτήσεις αυτές πάλι, έχουν ολόιδια λειτουργία με τις addMovie, addSeries, addGame απλώς αυτές προσθέτουν πελάτες στους αντίστοιχους vectors -δημιουργώντας ένα νέο αντικείμενο για τον καινούριο πελάτη- και στην συνέχεια στο αντίστοιχο αρχείο. Και αυτές οι δύο αυξάνουν τον αριθμό των πελατών που βρίσκεται αποθηκευμένος στα αρχεία num\_of\_cl.txt και number\_of\_VIP.txt.
* **void deleteClients(ClientsVector, VIPClientsVector) [γραμμή 1513]:** Η συνάρτηση αυτή έχει ίδια λειτουργία με τις συναρτήσεις delete των ταινιών, σειρών και παιχνιδιών απλώς αυτή είναι μοναδική και εξυπηρετεί και τη διαγραφή των απλών πελατών αλλά και των VIP με τον ίδιο τρόπο που το καταφέρνουν και οι παραπάνω delete συναρτήσεις. Ζητάει, δηλαδή, το ID του πελάτη που πρόκειται να διαγραφεί και το αν είναι VIP ή όχι και μέτα αφού κάνει erase το αντικείμενο του πελάτη από τον αντίστοιχο vector τότε τον ξαναγράφει ολόκληρο στο αντίστοιχο αρχείο.
* **void modifyClients(ClientsVector, VIPClientsVector) [γραμμή 1630]:** Μία ακόμη συνάρτηση που λειτουργεί με τον ίδιο τρόπο με τις αντίστοιχες modify, αυτή την φορά, συναρτήσεις των ταινιών, σειρών παιχνιδιών. Απλώς επιτρέπει την τροποποιήση και απλών πελατών και VIP. Εμφανίζει στο χρήστη της επιλογές τροποποιήσης μέσω της συνάρτησης c\_modify\_menu() [γραμμή 1613].
* **void load\_rents(RentVector,choice) [γραμμή 632]:** Ξανά μία ακόμη load συνάρτησης φόρτωσης vector, η οποία φορτώνει τον vector που περιέχει αντικείμενα της κλάσης rent\_info. Ανάλογα με το αν ο χρήστης επιθυμεί να φορτώσει τους δανεισμούς ταινίας ή σειράς ή παιχνιδιών, δημιουργούνται τα κατάλληλα αντικείμενα , με τη βοήθεια των αρχείων rented\_m.txt, rented\_s.txt, rented\_vg.txt, και φορτώνονται στον vector. Καλείται και απο την main και από την rent\_msvg.
* **void rent\_msvg(MoviesVector,SeriesVector,GamesVector,RentVector) [γραμμή 1030]:** Συνάρτηση που επιτρέπει ή απαγορεύει το δανεισμό καποια ταινίας κλπ. Στην αρχή ζητείται από το χρήστη να δηλώσει το είδος του προιόντος που θέλει να δανειστεί και στη συνέχεια ,αφού φορτωθούν οι ήδη υπάρχοντες δανεισμού στον RentVector μέσω της load\_rents, ζητείται το ID από αυτό που θέλει να δανειστεί ο χρήστης και δημιουργείται το κατάλληλο αντικείμενο της κλάσης rent\_info το οποίο επίσης φορτώνεται στον RentVector, οποιος στη συνέχεια γράφεται ολόκληρος στο κατάλληλο αρχείο. Αν πραγματοποιηθεί ο δανεισμός τότε μειώνει κατά 1 τη μεταβλητή num\_of\_copies του αντικειμένου του αντίστοιχου vector(π.χ. MoviesVector) που "δανείστηκε" στον πελατή και έτσι ξαναγράφω τον vector στο αντίστοιχο αρχείο με ενημερωμένο num\_of\_copies.
* **void calc\_cost(RentVector, VIPClientsVector, MoviesVector, SeriesVector, GamesVector, choice) [γραμμή 733]:** Συνάρτηση υπεύθυνη για τον υπολογισμό του κόστους των δανεισμών. Στην αρχή της ζητάει το ID του πελάτη που επιστρέφει την ταινία για να εξακριβώσει αν ο πελάτης ειναι VIP ή όχι μέσω της συνάρτησης isVIP(VIPClientsVector ,id). Στη συνέχεια, με καθαρά μαθηματικό τρόπο και σύμφωνα με τα στοιχεία χρέωσης που μας δώσατε υπολογίζει και εμφανίζει τη χρέωση του πελάτη. Στο τέλος προσθέτει και +1 στη μεταβλητη num\_of\_copies του αντικειμένου, που επιστραφηκε, του αντίστοιχου vector και τον ξαναγράφει στο αντίστοιχο αρχείο.
* **bool isVIP(VIPClientsVector, id) [γραμμή 719]:** Ελέγχει αν το ID του πελάτη ανήκει σε VIP πελάτη. Αν ναι, τότε επιστρέφει true. Αν όχι, τότε ο πελάτης είναι "απλός" και επιστρέφει false.
* **void re\_copies(MoviesVector, SeriesVector, GamesVector) [γραμμή 1216]:** Η συνάρτηση αυτή ζητάει από τον χρήστη να δηλώσει το είδος του προιόντος που θέλει να αναζητήσει (π.χ. ταινία, σειρά) και στη συνέχεια ζητάει να δώσει ο χρήστης είτε τον αριθμό ID είτε το όνομα του προιόντος και αφού το προιόν βρεθεί στον αντίστοιχο vector τότε εμφανίζεται το πλήθος των αντιγράφων του προιόντους που έψαξε ο χρήστης. Δυστυχώς, μόνο όταν έφτασα σε αυτή τη λειτουργία κατάλαβα τι εννοούσατε με τον σειριακό αριθμό αντιγράφου. Εγώ πίστευα ότι και αυτός θα ήταν μοναδικός για κάθε ταινία. Όταν έφτασα σε αυτή τη λειτουργία όμως είχα γράψει ήδη 1700+ γραμμές κώδικα και θα έπρεπε να αλλάξω πολλά πράγματα για να το κάνω ακριβώς όπως το θέλετε και θεωρώ πως θα χανόμουν μέσα σε όλες αυτές τις γραμμές, οπότε θα σας περιγράψω εδώ τη λογική με την οποία θα το έκανα αν το είχα συνειδητοποιήσει από την αρχή: Αντί για μεταβλητή int serial\_number στις κλάσεις, θα έφτιαχνα πίνακα int serial\_number[10] τον οποίον θα αρχικοποιούσα με 0 σε κάθε θεση (10 θέσεων γιατί θα έβαζα λογική παραδοχή ότι το Video Club δε θα παρείχε πάνω απο 10 αντίγραφα μιας ταινιας). Όταν θα πήγαινα να κανω addΜovie για παράδειγμα, θα ζητούσα και τους 10 σειριακούς αριθμούς αντιγράφων με επανάληψη. Η διαγραφή θα παρέμενε ίδια. Στην τροποποίηση αν ήθελα να αλλάξω τον αριθμό αντιγράφων και να τον μειώσω από τα 10 τότε απλά θα έβαζα στις θέσεις του πίνακα που θα επρέπε 0 (δηλαδή δεν υπάρχει αντίγραφο)**.** Στο δανεισμό θα έβαζα 0 στη θέση του αντιγράφου που δανείστηκε και θα περνούσα τον σειριακο αριθμό της θέσης αυτής στο αρχείο rent\_m (αν επροκειτο για ταινία) και στην επιστροφή θα έπαιρνα πίσω αυτόν τον αριθμό από το αρχείο και θα το ξαναέβαζα στον πίνακα. Κατά τη διάρκεια όλων αυτών των ενεργειών θα μείωνα ή θα αύξανα τη μεταβλητη num\_of\_copies όπως κάνω και στο πρόγραμμα μου. Τέλος στην τελευταία λειτουργία απλώς θα εμφάνιζα τον πίνακα αντί για το num\_of\_copies. Δε θα ήταν κάτι σπουδαίο ή δύσκολο οι αλλαγές που θα έκανα στον κώδικα απλώς ειναι η πρώτη φορά που έγραψα τόσο μέγαλο πρόγραμμα μόνος μου και το φοβήθηκα μην το χάσω κάπου λόγω των πολλών γραμμών.

*Λογικές Παραδοχές:*

Η παραδοχή στην οποία βασίζεται κυρίως το πρόγραμμα είναι το ότι θα το χρησιμοποιεί κάποιος εργαζόμενος του Video Club και όχι κάποιος άλλος "άσχετος" χρήστης. Τα αρχεία είναι δομημένα με τέτοιο τρόπο, ώστε να είναι αποτελεσματική η ανάγνωση τους από το πρόγραμμα. Μία ακόμα παραδοχή είναι το ότι το πλήρες όνομα των πελατών δε μπορεί να ξεπερνάει τους 20 χαρακτήρες καθώς και των τραπεζών τους 10 (δημιούργησα στατικούς πίνακες για αυτά τα τρία). Επιπλέον, σε μερικά σημεία ο χρήστης μπορεί να χρειαστεί να πατήσει 2 ή 3 φορές το Enter για να συνεχίσει το πρόγραμμα και αυτό κυρίως επειδή έχω χρησιμοποιήσει συναρτήσεις .ignore() γιατί αλλιώς το πρόγραμμα προσπερνούσε κάποιες getline() και cin που έχω. Επίσης, στις τροποποιήσεις, έδωσα τη δυνατότητα στο χρήστη να τροποποιεί αυτά που θεώρησα ότι είναι λογικό να χρειάστουν τροποποίηση. Για παράδειγμα, δεν του δίνω τη δυνατότητα να τροποποιήσει το χρόνο που κυκλοφόρησε η ταινία, γιατί είναι κάτι πολύ συγκεκριμένο και δε δέχεται αλλαγή.Τέλος, στον υπολογισμό της χρέωσης θεωρώ πως οι μήνες έχουν 30 μέρες.