Τελική Αναφορά

Στοιχεία Μελών

- Θεόδωρος Τζίμας - 1115202000190 - 3ο έτος φοίτησης – sdi2000190@di.uoa.gr
- Ζαφείριος Γεωργιάδης - 1115202100019 - 2ο έτος φοίτησης – sdi2100019@di.uoa.gr

Καταμερισμός εργασίας

Αποφασίσαμε να γράψουμε μαζί τον κώδικα ταυτόχρονα, για όλα τα μέρη της εργασίας, χρησιμοποιώντας την δυνατότητα Live Share που προσφέρεται στα Visual Studio και Visual Studio Code.

Περιεκτική περιγραφή προγράμματος/κώδικα

Θα εξηγήσουμε συνοπτικά τη χρήση των κυριότερων συναρτήσεων των κλάσεων:

Ξεκινώντας από την κλάση Map, ο constructor της είναι υπεύθυνος για την δυναμική δέσμευση μνήμης της μεταβλητής Grid (private μέλος), η οποία αντιπροσωπεύει τον χάρτη (δισδιάστατο πίνακα) στον οποίο θα πραγματοποιηθεί μία παρτίδα του παιχνιδιού και την αρχικοποίησή του με την προσθήκη των φυσικών εμποδίων (γη, νερό, δέντρα) και του φίλτρου.

Η συνάρτηση UpdateAvatar παίρνει ως ορίσματα τις συντεταγμένες της προηγούμενης θέσης του Avatar και τον ίδιο, ώστε να ενημερώσει τον χάρτη με την καινούρια θέση του Avatar και να κάνει κενή (να δηλώσει τον χώρο ως «γη») την προηγούμενή του.

Η συνάρτηση SetCoordinates, που παίρνει ως όρισμα ένα αντικείμενο της κλάσης Entity, είναι αναδρομική, και χρησιμοποιείται για να αρχικοποιήσει τις συντεταγμένες των οντοτήτων με τυχαίο τρόπο, μέσω της SetYX της κλάσης Entity και τις τοποθετεί στον χάρτη (Grid) . Η αναδρομική της λειτουργία αξιοποιείται στις περιπτώσεις όπου η τυχαία θέση που της απονέμεται αρχικά είναι ήδη κατειλημμένη ή όταν ανιχνευθεί (από την ίδια συνάρτηση) ότι η θέση αυτή αποκλείει την οντότητα, με αποτέλεσμα να μην μπορεί να μετακινηθεί.

Η κλάση Map διαθέτει στα public μέλη της δύο vectors. Αυτά χρησιμοποιούνται για την αποθήκευση δεικτών που δείχνουν σε κάθε μέλος της κάθε ομάδας (Werewolves ή Vampires). Η συνάρτηση SpawnEntities επιλέγει έναν τυχαίο, μη μηδενικό, αριθμό που αντιπροσωπεύει τον αριθμό των NPCs που θα έχει η κάθε ομάδα. Έπειτα, δημιουργεί το κάθε NPC της κάθε ομάδας και, μέσω της SetCoordinates (που εξηγήσαμε από πάνω) και των constructor της κλάσης κάθε οντότητας, τις αρχικοποιεί και τις τοποθετεί στον χάρτη.

Η συνάρτηση Scan, μία από τις πιο σημαντικές συναρτήσεις, που παίρνει ως όρισμα συντεταγμένες y και x, δεσμεύει δυναμικά στην μνήμη και επιστρέφει έναν μονοδιάστατο πίνακα (με όνομα NearPositions) από χαρακτήρες, τους οποίους παίρνει κατευθείαν από την μεταβλητή Grid. Ο πίνακας NearPositions αντιπροσωπεύει το 3x3 τετράγωνο των θέσεων που σχηματίζεται γύρω από τις δεδομένες συντεταγμένες, χωρίς να περιέχει την θέση των συντεταγμένων, με σειρά από τα αριστερά προς τα δεξιά και από πάνω προς τα κάτω. Για παράδειγμα, αν δίναμε στην Scan τις συντεταγμένες ενός werewolf για να ελέγξουμε που μπορεί να κινηθεί, η θέση από πάνω του θα ήταν η NearPositions[1] και η από κάτω του η NearPositions[6] (διότι, όπως προαναφέρθηκε, ο πίνακας δεν περιέχει την θέση των δεδομένων συντεταγμένων). Εάν τις δοθούν συντεταγμένες στις άκρες του Grid, οι περιοχές εκτός αυτού

συμβολίζονται με ‘E’. Η Scan χρησιμοποιείται για τον έλεγχο του περιβάλλοντος ενός αντικειμένου, με σκοπό να αποφασιστεί η ενέργεια την οποία αυτό θα εκτελέσει.

Η συνάρτηση Iterate απλά επιστρέφει τον αριθμό της θέσης κάποιου δεδομένου NPC μέσα από το vector της ομάδας του.

Η συνάρτηση GetNPC δέχεται τις συντεταγμένες ενός NPC, το vector στο οποίο ανήκει και το index ενός πίνακα που προκύπτει από την κλήση της Scan. Η λειτουργία της είναι να επιστρέφει το NPC που βρίσκεται κοντά στις δεδομένες συντεταγμένες (στο ίδιο 3x3 τετράγωνο υποπίνακα του Grid με κέντρο τις δεδομένες συντεταγμένες) χρησιμοποιώντας τη σχετική τους θέση που λαμβάνει από το index του πίνακα της Scan, η οποία καλείται με τις ίδιες συντεταγμένες. Πρώτα, η GetNPC υπολογίζει τις συντεταγμένες του ζητούμενου NPC χρησιμοποιώντας το index (με όνομα μεταβλητής block), μετά αναζητεί για το NPC μέσα στο δεδομένο vector με κριτήριο τις υπολογισμένες συντεταγμένες και το επιστρέφει.

Η Action, που παίρνει ως όρισμα δείκτη σε NPC, αποφασίζει τυχαία για την ενέργεια που θα εκτελέσει το NPC για έναν γύρο. Αρχικά, η Action καλεί την Scan, με τις συντεταγμένες του NPC, και παίρνει το array με τις θέσεις, και τα περιεχόμενά τους, γύρω από το NPC. Έπειτα, η Action επιλέγει τυχαία μια από τις θέσεις του και, ανάλογα το περιεχόμενο της θέσης, αποφασίζει μια ενέργεια για το NPC. Αν βρει εμπόδιο τότε μένει ακίνητο, αν βρει σύμμαχο και έχει έστω ένα γιατρικό τότε τον επουλώνει κατά μία μονάδα, αν βρει εχθρό και το επίπεδο δύναμης του είναι μεγαλύτερο ή ίσο από εκείνο του εχθρού τότε του επιτίθεται, αλλιώς προσπαθεί να ξεφύγει. Τέλος, αν βρει γη (ελεύθερη θέση) τότε καλεί την συνάρτηση Move με όρισμα την τυχαία θέση και μετακινείται. Εδώ αναφέρουμε ότι όλες οι Move (των Avatar, Vampire και Werewolf) λαμβάνουν και αυτές το index του πίνακα μιας Scan και με βάση αυτό εκτελούν την κίνηση της αντίστοιχης οντότητας.

Η print απλά εκτυπώνει τον χάρτη (Grid) στο τερματικό.

Συνεχίζοντας με την κλάση Game, η οποία είναι και το κύριο μέσο με το οποίο μια παρτίδα οργανώνεται και χειρίζεται, ο constructor της αρχικοποιεί τον Avatar μέσω της SetCoordinates και τα NPC μέσω της SpawnEntities πάνω στον χάρτη, που υπάρχει, μαζί με τον Avatar, μέσα στο private μέλος της κλάσης. Σε αυτό το σημείο το παιχνίδι είναι έτοιμο να ξεκινήσει.

Η Update εκτελεί μια ενέργεια για κάθε ζωντανή οντότητα μέσω της συνάρτησης Action του Map.

Η PauseGame εκτυπώνει των αριθμό των ζωντανών Vampires και Werewolves και τον αριθμό των διαθέσιμων φίλτρων επούλωσης και σταματάει προσωρινά το παιχνίδι.

Η συνάρτηση ExitGame τερματίζει το παιχνίδι και συνεπώς το πρόγραμμα.

Η συνάρτηση Input ελέγχει την είσοδο του πληκτρολογίου και ανάλογα με το πατημένο κουμπί εκτελεί την ανάλογη εντολή. Αν πατηθεί το ‘P’ τότε καλείται η PauseGame, αν πατηθεί το ‘E’ τότε καλείται η συνάρτηση ExitGame, αν πατηθεί το ‘H’ και ανάλογα με το ποια ομάδα υποστηρίζει ο παίκτης καθώς και την φάση της ημέρας (μέρα/νύχτα) τότε καλείται η Heal του Avatar και εφόσον υπάρχουν φίλτρα, επουλώνονται όλα τα μέλη της ομάδας που υποστηρίζει. Τέλος, αν πατηθούν τα πλήκτρα των βελών, τότε καλείται η Move του Avatar και κινείται αναλόγως και σύμφωνα πάντα με ένα array για τις θέσεις γύρω του (από την Scan).

Η ChangeDayTime χειρίζεται την εναλλαγή από μέρα σε νύχτα και το αντίθετο.

Στο τέλος, η MainLoop ευθύνεται για τις κύριες εντολές διεξαγωγής του παιχνιδιού. Κάθε loop που εκτελεί σηματοδοτεί ένα γύρο του παιχνιδιού, στον οποίο όλα τα NPC εκτελούν μία ενέργεια (κλήση της Update), γίνεται έλεγχος για πατημένα πλήκτρα και οι ανάλογες ενέργειες (κλήση της Input) και η κατάσταση του χάρτη εκείνη τη στιγμή εμφανίζεται στο τερματικό (κλήση της Map::print) μαζί με το αν είναι ημέρα ή νύχτα. Κάθε δέκα γύρους γίνεται η εναλλαγή των τελευταίων.

Παραδοχές

1. Το νερό συμβολίζεται με 'S', τα δέντρα με 'T', το γιατρικό με 'P', η γη (κενές θέσεις) με '-' , τα Vampires με 'v', τα Werewolves με 'w' και ο Avatar είτε με 'W' είτε με 'V', ανάλογα με την ομάδα που υποστηρίζει
2. Το 'γιατρικό' που έχουν τα NPCs καθώς και το 'φίλτρο μαγικής επούλωσης' του Avatar τα ορίσαμε με το ίδιο όνομα μεταβλητής (Potions) στην κλάση Entity. Παρ' όλα αυτά, το Potions των NPC επουλώνει κατά μια μονάδα το επίπεδο υγείας κάποιου συμμάχου, ενώ το αντίστοιχο του Avatar επουλώνει στο μέγιστο επίπεδο υγείας (10) όλα τα ζωντανά NPC της ομάδας που υποστηρίζει
3. Σε κάθε επίθεση το επίπεδο άμυνας του αμυνόμενου NPC πέφτει κατά μία μονάδα μέχρι να μηδενιστεί
4. Η τυχαιότητα στην ενέργεια κάθε NPC αποφασίζεται στην συνάρτηση Map::Action μέσω της επιλογής ενός τυχαίου block από αυτά που περιβάλλουν το NPC. Ανάλογα με το τι βρίσκεται στο επιλεγόμενο block και τις επιπλέον συνθήκες (για Heal, Attack, Move, Ακινησία) εκτελείται και η αντίστοιχη ενέργεια. Ειδικότερα, για το Werewolf, μπορεί να ελέγξει και τις διαγώνιες θέσεις (ακόμη και αν δεν μπορεί να κινηθεί σε αυτές) και αν επιλεγεί μια από αυτές, αποφασίσαμε να ξαναδοκιμάσει να εκτελέσει ενέργεια ώστε να μην βρίσκεται σε μειονεκτική θέση σε σχέση με το Vampire.
5. Ορίσαμε το αρχικό επίπεδο υγείας για κάθε οντότητα στις 10 μονάδες
6. Η ημέρα εναλλάσσεται σε νύχτα, και το αντίθετο, ανά 10 γύρους

IDE/ Compiler / Λειτουργικό

Κατά την συγγραφή χρησιμοποιήσαμε κυρίως το Visual Studio Code και για το compile χρησιμοποιήσαμε τόσο τον compiler του Dev C++ όσο και τον compiler MinGW μέσω του VS Code. Τέλος, κατά τον διαμερισμό στα επιμέρους κομμάτια χρησιμοποιήσαμε το Visual Studio 2022 και τον ενσωματωμένο compiler του. Ως λειτουργικό σύστημα χρησιμοποιήθηκαν τα Windows 11 για να υπάρχει η δυνατότητα χρήσης της βιβλιοθήκης <windows.h>.

Προβλήματα που αντιμετωπίσαμε

Δεν αντιμετωπίσαμε πάρα πολλά προβλήματα κατά την συγγραφή του κώδικα. Ωστόσο, θα πρέπει να αναφερθούν οι δυσκολίες που αντιμετωπίσαμε με την δεδομένη εξίσωση για την συνάρτηση attack. Όπως γράψαμε και παραπάνω κάναμε την παραδοχή το Επίπεδο Άμυνας να λειτουργεί σαν μια "ασπίδα" που καταστρέφεται καθώς σε κάθε επίθεση μειώνεται κατά 1 επίπεδο μέχρι να μηδενιστεί. Αυτό είναι ενάντια στην εκφώνηση, ωστόσο με το να το κρατάγαμε στάσιμο υπήρχε η πιθανότητα, ενώ το επίπεδο δύναμης του επιτιθέμενου να είναι ίσο με εκείνο του αμυνόμενου NPC, το επίπεδο άμυνας του αμυνόμενου να είναι μεγαλύτερο από το επίπεδο δύναμης του επιτιθέμενου. Σε αυτή την περίπτωση, η επίθεση έχει ως αποτέλεσμα να αυξήσει το τρέχων επίπεδο υγείας του αμυνόμενου, αντί να το μειώσει.

Απαιτήσεις της εργασίας που δεν υλοποιήθηκαν

Υλοποιήθηκαν όλες επιτυχώς.

Βαθμός Δυσκολίας

Από την μια πλευρά, το project δεν είχε ιδιαίτερα υψηλό βαθμό περιπλοκότητας στην υλοποίηση του. Από την άλλη, το σχετικά μικρό χρονικό περιθώριο και η μεγάλη έκτασή του σε συνδυασμό με τις υποχρεώσεις των υπόλοιπων μαθημάτων κατέστησαν την υλοποίηση του πιο πιεστική απ' όσο αρμόζει σε ένα τέτοιο project.

Ιδιωτικό αποθετήριο στο Github

https://github.com/sdi2000190/OOPVampsVWere

Σύνδεσμος με το βίντεο στο YouTube

https://youtu.be/n2xvxGZDPUA