Προγραμματισμός Υπολογιστών με C++

Εργασία 2020/2021

FlappyBird ++

Ονοματεπώνυμο & ΑΜ :

Βασίλειος – Ηλίας Δρούζας (**3180051**)

Καλησπέρα σας!

Με χαρά και όρεξη δημιούργησα μια παραλλαγή του κλασικού παιχνιδιού ‘FlappyBird’, το δικό μου ‘**FlappyBird++**’. Όπως προμηνύει το όνομα της δικής μου εκδοχής, το FlappyBird++ διαθέτει κάποια επιπλέον χαρακτηριστικά σε σχέση με το κλασικό FlappyBird.

Γενικά χαρακτηριστικά του παιχνιδιού:

Ο παίχτης μας είναι το flappybird, ένα πουλί που πετάει και σκοπός είναι να αποφύγει τα εμπόδια που έχει μπροστά του. Τα εμπόδια αυτά είναι οι αγωγοί (ή κοινώς,τα λεγόμενα pipes). Το παιχνίδι δεν τελειώνει παρά μόνο όταν χάσει ο παίχτης μας, επομένως ο στόχος του είναι ο παίχτης να επιτύχει το μεγαλύτερο δυνατό σκορ. Ξεκινάει στην αρχή με μηδενικό σκορ και κάθε φορά που περνάει ανάμεσα απο δύο pipes,το σκορ του αυξάνεται κατά 1, -όπως ακριβώς και στην εκδοχή του γνωστού FlappyBird’-. Ο παίχτης πρέπει να πατάει ένα κουμπί για να σηκώνει το flappybird στη προσπάθεια του να αποφύγει τα εμπόδια. Όσο δεν πατάει το κουμπί, το πουλί μας πέφτει και αν πέσει στο ύψος του εδάφους,πεθαίνει και το παιχνίδι τελειώνει.

Κάπου εδώ αρχίζουν οι διαφορές μας σε σχέση με το κλασικό παιχνίδι. Έχουμε δύο τύπους από εμπόδια, τα *κόκκινα* και *πράσινα* pipes. Τα κόκκινα είναι θανατηφόρα -αν συγκρουστεί το πουλί μας με αυτά πεθαίνει επιτόπου- και το παιχνίδι τελειώνει. Τα πράσινα ναι μεν δεν είναι θανατηφόρα επιτόπου, αφαιρούν όμως ένα κομμάτι της ζωής του flappybird (εντάξει, αυτό ίσως να μην βγάζει πρακτικά πολύ νόημα, ας θεωρήσουμε ότι τα πράσινα pipes είναι άυλα αλλά δηλητηριώδη και για αυτό ο παίχτης απλά χάνει ζωή αντί να πεθαίνει επιτόπου. Ο λόγος που έβαλα αυτό το χαρακτηριστικό είναι για μεγαλύτερη ποικιλία εμποδίων στο παιχνίδι). Με το αρχικό default ποσό ζωής του παίχτη, αρκούν 3 χτυπήματα για να χάσει το πουλί μας τη ζωή του. Για να ρεφάρει τα χτυπήματα από πράσινα pipes, το παιχνίδι μας κάνει spawn σε τυχαίο σημείο καρδιές, οι οποίες αυξάνουν τη ζωή του παίχτη,εφόσον ο παίχτης ‘συγκρουστεί’ με αυτές. Αντίστοιχα πρέπει να προσέχει τις βόμβες, οι οποίες εκρήγνυνται και του αφαιρούν ζωή εφόσον βρεθεί κοντά σε αυτές.

Το παιχνίδι διαθέτει 6 είδη μενού, τα *start,difficulty,modechoice,help,level,end*.Το μενού που υλοποιεί τη λειτουργικότητα του παιχνιδιού είναι το προτελευταίο.Το πρώτο μας καλωσορίζει στο παιχνίδι,στο δεύτερο καλούμαστε να επιλέξουμε επίπεδο δυσκολίας (υπάρχουν 2, τα **normal**,**hard**.) Η διαφορά ανάμεσα στα 2 επίπεδα είναι ότι στο hard έχουμε περισσότερα pipes, άρα χρειάζεται και περισσότερη προσπάθεια απο τον παίχτη.

Το τρίτο μας ρωτά για τον τύπο του παιχνιδιού που θέλουμε να παίξουμε, **Single Player** ή **2-player**.Επιλέγουμε το δεύτερο όταν θέλουμε να παίξουμε με κάποιον φίλο μας από τον ίδιο υπολογιστή.Στο 2-player mode δεν έχει σημασία το σκορ του καθενός (σχετικά με τα πόσα pipes περνά),αλλά ο νικητής κάθε γύρου,ο οποίος είναι αυτός που θα αντέξει παραπάνω μέχρι ο αντίπαλός του να ‘τρακάρει’.

Μόλις το παιχνίδι τελειώσει (μετά δηλαδή απο κάποιο ‘τρακάρισμα’), οδηγούμαστε στο τελικό μενού (end) , το οποίο μας λέει

α)Στην περίπτωση του Single Player , το σκορ μας στη πίστα και το high score γενικά (από οποιαδήποτε στιγμή,χωρίς να χάνεται όταν κλείσει η εφαρμογή).

β) Στο 2-player, ανιχνεύει το νικητή του γύρου και εμφανίζει το συνολικό σκορ των 2 παιχτών (κάθε φορά που ανιχνεύεται νικητής κάθε γύρου,αυξάνεται το σκορ του κατά 1).

Πριν αρχίσουμε να παίζουμε, πάντα υπάρχει η δυνατότητα να πάρουμε κάποιες πληροφορίες σχετικά με το παιχνίδι.Το μενού help μας εξηγεί κάποια βασικά πράγματα για αυτό.

Σημείωση: Από κάθε μενού μπορούμε να πάμε σε άλλο ανάλογα με τις επιλογές που μας δίνει το παιχνίδι.

Σχόλια σχετικά με την υλοποίηση του κώδικα

Έχουμε οντότητες GameObject,που είναι οι Player, Player2, Friend, Pipe, Bomb. Διαθέτουν τις τυπικές μεθόδους update,draw για την ενημέρωση και τη σχεδίαση αντίστοιχα.Χρησιμοποιείται η κληρονομικότητα και ο πολυμορφισμός για την υλοποίηση αυτών των οντοτήτων. Οι παίχτες,τα pipes και ο Friend (δηλαδή οι καρδιές) επίσης δημιουργούνται δυναμικά και καταστρέφονται όταν πλέον δεν χρειάζονται. Προσπάθησα επίσης να εντάξω συλλογές στην εφαρμογή (αρχικοποίηση των pipes σε vector,update των αντικειμένων διατρέχοντας με iterator και erase του vector) ,αλλά το πείραμα δεν πέτυχε, διότι τα pipes έρχονταν αστραπιαία ☹. Τις εντολές αυτές τις έχω αφήσει σε σχόλια για να τις αξιολογήσετε και να σας ρωτήσω για να λύσω την απορία μου γιατί συμβαίνει αυτό.

Ενδεικτικές μέθοδοι είναι:

• η *CheckCollision* , η οποία καλεί τις *CheckCollision*1 και *CheckCollision2* για να βρει αν υπήρξε σύγκρουση του παίχτη 1 και 2 αντίστοιχα, με pipe ή καρδιά.

• Οι μέθοδοι του high score που παίρνουν το υψηλό σκορ και συγκρίνουν με το τωρινό. Αν το τωρινό υπερτερεί, ενημερώνουν το high score του αντίστοιχου .txt φακέλου.

• Όπως είπαμε πριν, οι *update* και *draw* , οι οποίες καλούν τις υπομεθόδους *update<X>Screen*, *draw<X>Screen* , όπου <X> το επίπεδο που βρισκόμαστε κάθε φορά (start,difficulty,modechoice,help,level,end), και η μέθοδος αρχικοποίησης *init*.

Επεξήγηση κάποιων μεθόδων στην κλάση **Game**:

• *getHighScore*: Επιστρέφει το high score από το txt αρχείο

• *drawHighScore*: Τυπώνει το high score

• *updateHighScore*:Ενημερώνει το high score του txt αρχείου

• *checkScore*: Αν ο παίχτης έχει περάσει από 2 εμπόδια(πάνω/κάτω) αυξάνεται το σκορ κατά 1.

• *DeathPlayerX*: Διαγράφει τον παίχτη που πεθαίνει (ο X, όπου X={1,2}) και μεταβαίνει στο τελικό μενού.(καλείται στην CheckCollision)

• *checkCollision*: Χρησιμοποιώντας τη checkCollision1 και checkCollision2 ψάχνει για συγκρούσεις με τα pipes των παιχτών 1 και 2 αντίστοιχα.

• *resetPipes*: Κάνει reset τις συντεταγμένες των pipes.

• *spawnPipe*: Κάνει spawn τα pipes ανάλογα με τη δυσκολία.

• *deletePipe*: Διαγράφει τα pipes (αποδεσμεύει μνήμη)

• *StartScreen*: Με τις αντίστοιχές της update και draw,εμφανίζει την αρχική οθόνη και υλοποιεί την αρχική λειτουργικότητα.

• *DifficultyScreen*: Τα ίδια με πάνω αλλά εδώ ο παίχτης επιλέγει επίπεδο δυσκολίας.

• *ChoiceScreen*:Τα ίδια με πάνω αλλά ο παίχτης επιλέγει game mode (Single Player / 2-player).

• *HelpScreen*: Επεξηγήσεις σχετικά με το παιχνίδι

• *LevelScreen*:Μέσω των αντίστοιχων update και draw για τους 2 παίχτες υλοποιεί τη λειτουργικότητα του παιχνιδιού.Αρχικά δημιουργεί τους παίχτες,τον Friend και τα pipes και υλοποιεί τα αντίστοιχα update από τη καθεμία κλάση αυτών,ανάλογα με τη δυσκολία που επιλέχθηκε.Ταυτόχρονα ελέγχει για τα collisions (καλώντας την checkCollision)

• *EndScreen*: Διαγράφει τα pipes καλώντας την deletePipe και υλοποιεί τις αντιστοιχες update,draw.

• *DrawPlayerXLife*: Εμφανίζει τη μπάρα ζωής του παίχτη X , όπου X={1,2}.

• Μέθοδοι *update*,*draw* στο τέλος καλούν την σωστή υπομέθοδο τους κάθε φορά ανάλογα με το status.