ΤΟ PROJECT ΜΑΣ

1. Εισαγωγή & Motivation

Το συγκεκριμένο project αποτελεί την υλοποίηση ενός δισδιάστατου (2D) παιχνιδιού οδήγησης με τίτλο «Drunk Drivers», αναπτυγμένο σε γλώσσα προγραμματισμού C++, με χρήση της βιβλιοθήκης SFML (*Simple and Fast Multimedia Library*).
Πρόκειται για ένα απλό και εύκολο παιχνίδι στο οποίο ο παίκτης καλείται να ελέγξει ένα όχημα που κινείται σε δρόμο, αποφεύγοντας τα επερχόμενα αυτοκίνητα, υπό τις συνθήκες ενός «μεθυσμένου» χειρισμού.

Ο σκοπός του παιχνιδιού είναι να ψυχαγωγήσει τον χρήστη μέσω ενός διασκεδαστικού gameplay αλλά συγχρόνως και να καλλιεργήσει τις ικανότητές του. Το παιχνίδι απαιτεί διάφορες δεξιότητες,όπως για παράδειγμα συγκέντρωση,γρήγορα αντανακλαστικά και συντονισμό χεριού-ματιού για την αποφυγή εμποδίων.Εξασφαλίζεται έτσι μία ευχάριστη εμπειρία που προκαλεί το ενδιαφέρον των παικτών.

Η επιλογή της συγκεκριμένης ιδέας προκύπτει από τη δική μας προσωπική ενασχόληση με τα βιντεοπαιχνίδια, η οποία ξεκίνησε από πολύ νεαρή ηλικία. Τα παιχνίδια αποτελούσαν από πάντα για εμάς έναν αποτελεσματικό τρόπο διαφυγής από την πιεστική καθημερινότητα,προσφέροντας στιγμές χαλάρωσης και ψυχαγωγίας. Η διαδικασία δημιουργίας ενός 2D παιχνιδιού μας έδωσε τη δυνατότητα να εφαρμόσουμε τις ήδη υπάρχουσες γνώσεις μας στον προγραμματισμό σε πρακτικό επίπεδο αλλά και να τις διευρύνουμε.

2. Objective & Scope

Βασικός στόχος του project είναι η ανάπτυξη ενός απλού, λειτουργικού και ψυχαγωγικού 2D παιχνιδιού, με τη χρήση της γλώσσας προγραμματισμού C++ και την αξιοποίηση της βιβλιοθήκης SFML. Μέσα από την κατασκευή του παιχνιδιού, δοκιμάζονται οι γνώσεις μας στον αντικειμενοστραφή προγραμματισμό, στη διαχείριση γραφικών περιβάλλοντος, καθώς και στη χρήση εξωτερικών βιβλιοθηκών.Παράλληλα, επιτυγχάνεται η εξοικείωση με τον σχεδιασμό τόσο παιχνιδιών όσο και γενικότερα εφαρμογών σε C++.

Εντός του παιχνιδιού, ο παίκτης έχει τη δυνατότητα ελεύθερης κίνησης του οχήματος προς όλες τις κατευθύνσεις (πάνω, κάτω, αριστερά και δεξιά), εντός των ορίων του δρόμου. Η βασική επιδίωξη του παιχνιδιού είναι η παραμονή στον αυτοκινητόδρομο και αποφυγή σύγκρουσης με τα επερχόμενα οχήματα, τα οποία μάλιστα εμφανίζονται με τυχαίο τρόπο. Σε περίπτωση που το όχημα συγκρουστεί ή βγει εκτός ορίων το παιχνίδι τερματίζεται και ο παίκτης χάνει.

3. System Architecture

Με το άνοιγμα του αρχείου(.exe), ο κώδικας αρχικοποιεί τους τυχαίους αριθμούς με τη χρήση της srand, ώστε να υπάρχουν τυχαία spawns στα εμπόδια (δηλαδή τα επερχόμενα αντικείμενα). Κατά την εκκίνηση εκτελούνται όλα τα απαραίτητα αρχεία, όπως το φόντο, το αυτοκίνητο, τα κινούμενα αυτοκίνητα, η γραμματοσειρά και τα αρχεία ήχου.

Πριν την έναρξη του παιχνιδιού, εμφανίζεται στην αρχική οθόνη το μενού, μέσα από το οποίο ο χρήστης μπορεί να επιλέξει αν θα παίξει με έναν ή δύο παίκτες, να αλλάξει τις ρυθμίσεις (π.χ. ανάλυση) ή να κλείσει την εφαρμογή.

Μόλις ξεκινήσει το παιχνίδι, εκτελείται ένας βρόχος που διαχειρίζεται τα events (π.χ. πατήματα πλήκτρων), ενημερώνει τη θέση των αντικειμένων (αυτοκίνητα, εμπόδια, φόντο), ελέγχει τις συγκρούσεις και επανασχεδιάζει την οθόνη ώστε εμφανίζει τα τρέχοντα στοιχεία.

Το παιχνίδι Drunk Drivers βασίζεται σε μια σειρά από κύριες συνιστώσες, οι οποίες συνεργάζονται αρμονικά για να εξασφαλίσουν τη σωστή λειτουργία του συστήματος.

Αρχικά, η main είναι υπεύθυνη για τη δημιουργία και τη διαχείριση του παραθύρου του παιχνιδιού, χρησιμοποιώντας τη βιβλιοθήκη SFML. Με αυτό το παράθυρο, γίνεται η απόδοση όλων των γραφικών στοιχείων, όπως το φόντο και τα sprites των αυτοκινήτων και των εμποδίων.

Η runGame αναλαμβάνει τη συλλογή και επεξεργασία των εντολών από το πληκτρολόγιο, επιτρέποντας στους παίκτες να κινούν τα αυτοκίνητά τους με τη βοήθεια συγκεκριμένων πλήκτρων (τα βελάκια ή τα πλήκτρα W, A, S, D).

Οι παίκτες αντιπροσωπεύονται από δομές (structures) που περιλαμβάνουν τις ιδιότητες της θέσης, της ταχύτητας και της εμφάνισης. Ο ρόλος τους είναι η ενημέρωση της κίνησης και η διαχείριση συγκρούσεων με τα εμπόδια ή τον αντίπαλο παίκτη.

Τα εμπόδια, δηλαδή τα κινούμενα οχήματα προς την αντίθετη κατεύθυνση, δημιουργούνται με χρήση δομών και συναρτήσεων. Τοποθετούνται τυχαία στην οθόνη με τη βοήθεια της αρχικοποίησης που πραγματοποιεί η συνάρτηση srand στην αρχή του παιχνιδιού, ώστε να διασφαλιστεί πως τα οχήματα δεν εμφανίζονται με συγκεκριμένη σειρά. Η runGame διαχειρίζεται επίσης την κίνηση των εμποδίων, την αύξηση της δυσκολίας και τον έλεγχο σύγκρουσης με τους παίκτες.

Η διαχείριση των πόρων επιτυγχάνεται κυρίως με την main, η οποία φορτώνει και αποθηκεύει όλα τα σημαντικά αρχεία, όπως εικόνες (sprites), γραμματοσειρές για την εμφάνιση του σκορ και των μηνυμάτων, καθώς και αρχεία ήχου για τον ήχο φόντου και τα ηχητικά εφέ.

Ο κύριος βρόχος του παιχνιδιού (game loop) μπορεί να γίνεται στην main, αλλά η runGame αναλαμβάνει την συνεχή εκτέλεση βασικών λειτουργιών του παιχνιδιού. Συγκεκριμένα, σε κάθε επανάληψη του βρόχου γίνεται η λήψη και επεξεργασία των εντολών του χρήστη, ενημέρωση της θέσης των αντικειμένων, ανίχνευση πιθανών συγκρούσεων και ανανέωση της οθόνης με τα νέα γραφικά στοιχεία.

Τέλος, η κατάσταση του παιχνιδιού (όπως αν το παιχνίδι βρίσκεται στο μενού, σε

εξέλιξη, ή σε κατάσταση νίκης/ήττας) ελέγχεται μέσω μεταβλητών, επιτρέποντας την ομαλή μετάβαση μεταξύ των διαφόρων φάσεων του παιχνιδιού.

Διάγραμμα ροής

1. Εκκίνηση Προγράμματος

Ορίζονται οι συναρτήσεις του προγράμματος.
Εκτελείται η main()η οποία είναι υπεύθυνη για την αρχική προετοιμασία του παιχνιδιού και την εμφάνιση του μενού.

---

2. Αρχικοποίηση Παιχνιδιού

Δημιουργείται το κύριο παράθυρο του παιχνιδιού με χρήση της sf::RenderWindow, σύμφωνα με την ανάλυση που επιλέγει ο χρήστης από το μενού.
Οι πόροι (εικόνες, ήχοι, γραμματοσειρές) φορτώνονται απευθείας στις συναρτήσεις main() και runGame(), χωρίς ξεχωριστή συνάρτηση ή διαχείριση από αρχείο ρυθμίσεων.
Τα sprites για τα οχήματα, το φόντο και τα εμπόδια φορτώνονται με loadFromFile(), ενώ οι ήχοι και οι γραμματοσειρές αρχικοποιούνται αντίστοιχα μέσω SFML.

---

3. Εμφάνιση Κύριου Μενού

Το μενού υλοποιείται μέσω της main() με χρήση sf::Text για την εμφάνιση των επιλογών.
Ο χρήστης μπορεί να επιλέξει:

- Αριθμός παικτών(1 ή 2)
- Αλλαγή ανάλυσης
- Έξοδος από την εφαρμογή

Η αλληλεπίδραση γίνεται με πληκτρολόγιο μέσω sf::Event, χωρίς τη χρήση ποντικιού.

---

4. Έναρξη Παιχνιδιού

Κατά την επιλογή "Έναρξη παιχνιδιού", καλείται η runGame() η οποία περιλαμβάνει την κύρια λογική του παιχνιδιού.
Αρχικοποιούνται μεταβλητές και αντικείμενα όπως:

- Τα sprites των παικτών

- Τα sprites των εμποδίων (οχήματα)

- Ο μετρητής χρόνου (μέσω sf::Clock και sf::Text)

- Ηχητικά εφέ και μουσική υπόκρουση

---

5. Κύριος Βρόχος Παιχνιδιού

Ο βασικός βρόχος εκτελείται συνεχώς όσο το παιχνίδι είναι ενεργό. Περιλαμβάνει:

5.1 Διαχείριση Συμβάντων
Καταγραφή ενεργειών του χρήστη (π.χ. πατήματα πλήκτρων, κλείσιμο παραθύρου) μέσω sf::Event και Keyboard::isKeyPressed.

5.2 Ενημέρωση Κατάστασης

- Μετακίνηση των παικτών βάσει των πατημένων πλήκτρων

- Δημιουργία και μετακίνηση των εμποδίων

- Έλεγχος συγκρούσεων με intersects() μεταξύ sf::Sprite

- Ενημέρωση χρόνου και άλλων γραφικών στοιχείων

5.3 Σχεδίαση Περιβάλλοντος
Όλα τα γραφικά σχεδιάζονται σε κάθε frame με window.draw():

- Φόντο

- Οχήματα παικτών
- Εμπόδια
- Εμφάνιση πληροφοριών όπως χρόνος και μηνύματα

5.4 Έλεγχος Τερματισμού
Ελέγχεται αν κάποιος παίκτης συγκρούστηκε με εμπόδιο ή εξαντλήθηκαν οι ζωές. Αν συμβεί αυτό, εμφανίζεται μήνυμα τερματισμού και ο βρόχος σταματά.

---

6. Τερματισμός Παιχνιδιού

Στην περίπτωση ήττας ή τέλους παιχνιδιού, εμφανίζεται σχετικό μήνυμα (Game Over) με χρήση sf::Text.
Η μουσική σταματά και το παιχνίδι οδηγείται σε τερματισμό. Δεν χρησιμοποιείται ξεχωριστή συνάρτηση showEndScreen(), καθώς η εμφάνιση γίνεται απευθείας μέσα από τη runGame().

4. Τεχνολογίες που Χρησιμοποιήθηκαν

Για την υλοποίηση του παιχνιδιού «Drunk Drivers» χρησιμοποιήθηκαν συγκεκριμένες τεχνολογίες και εργαλεία, με στόχο τη δημιουργία ενός λειτουργικού και αποδοτικού 2D παιχνιδιού.

Η γλώσσα προγραμματισμού που χρησιμοποιήθηκε ήταν η C++, όπως ορίστηκε από τον διδάσκοντα. Η συγκεκριμένη γλώσσα κρίνεται ιδανική διότι προσφέρει πολλά πλεονεκτήματα,όπως η υψηλή απόδοση, ο αντικειμενοστραφής χαρακτήρας και η άμεση διαχείριση μνήμης. Τα χαρακτηριστικά αυτά ευνοούν την ανάπτυξη εφαρμογών που απαιτούν ταχύτητα και αποτελεσματικότητα, όπως για παράδειγμα τα βιντεοπαιχνίδια.

Για τη διαχείριση των γραφικών, του ήχου και των εισόδων του χρήστη χρησιμοποιήθηκε η βιβλιοθήκη SFML (Simple and Fast Multimedia Library). Συνιστά μία open-source βιβλιοθήκη για την C++ που παρέχει εύχρηστο API για τον χειρισμό πολυμέσων. Συγκεκριμένα, η SFML αξιοποιήθηκε για:

- δημιουργία παραθύρων
- απεικόνιση εικόνων και γραφικών
- αναπαραγωγή ήχων
- διαχείριση γραμματοσειρών και κειμένου

- την επεξεργασία εισόδων από πληκτρολόγιο και ποντίκι.

Η ανάπτυξη του λογισμικού πραγματοποιήθηκε στο περιβάλλον του Visual Studio Code, το οποίο υποστηρίζει την C++ μέσω διαφόρων επεκτάσεων. Μεταξύ αυτών χρησιμοποιήθηκαν:

- υποστήριξη για syntax highlighting
- debugging εργαλείο
- GitHub Copilot για αυτόματες προτάσεις κώδικα
- και διαχείριση αρχείων έργου.

Ο μεταγλωττιστής που χρησιμοποιήθηκε ήταν ο g++ (GNU Compiler Collection). Η ενσωμάτωσή του στη ροή εργασίας επιτεύχθηκε μέσω του Windows Subsystem for Linux (WSL), που επέτρεψε την αξιοποίηση εργαλείων ανάπτυξης σε περιβάλλον Linux από λειτουργικό σύστημα Windows.

Για την επεξεργασία των πολυμέσων χρησιμοποιήθηκαν επιπλέον τα εξής εργαλεία:

- Audiomass : διαδικτυακή εφαρμογή για τη δημιουργία και βασική επεξεργασία ήχου
- Adobe Photoshop : προσαρμογή και βελτιστοποίηση των εικόνων του παιχνιδιού
- ChatGPT : καθοδήγηση στη συγγραφή του κώδικα,τεκμηρίωση,επεξεργασία των μέσων.

5. Κώδικας & Υλοποίηση

Η δημιουργία του παιχνιδιού στηρίζεται σε δύο κύριες συναρτήσεις, τη main() και τη runGame(). Η main() είναι υπεύθυνη για την παρουσίαση του αρχικού μενού και την επιλογή ρυθμίσεων από τον χρήστη, ενώ η runGame() εκτελεί τον κύριο βρόχο του παιχνιδιού. Στην αρχή φορτώνονται οι απαραίτητοι πόροι του παιχνιδιού, όπως εικόνες για φόντο, αυτοκίνητα παικτών και εμπόδια, μέσω των κλάσεων sf::Texture και loadFromFile(). Παράλληλα φορτώνονται ήχοι χρησιμοποιώντας sf::SoundBuffer. Το μενού επιλογών δημιουργείται με τη χρήση sf::Text και sf::Font, ενώ η αλληλεπίδραση του χρήστη με το παιχνίδι γίνεται μέσω πληκτρολογίου, με ελέγχους sf::Keyboard. Η κίνηση των παικτών πραγματοποιείται σε πραγματικό χρόνο με βάση την είσοδο του χρήστη, και τα sprites των αυτοκινήτων ενημερώνονται διαρκώς ανά frame. Οι παίκτες χειρίζονται διαφορετικά πλήκτρα (WASD για τον παίκτη 2 και τα βελάκια για τον παίκτη 1), ενώ η ταχύτητα και η θέση τους ελέγχονται δυναμικά. Τα εμπόδια εμφανίζονται τυχαία και κινούνται συνεχώς προς τα κάτω. Σε κάθε frame γίνεται έλεγχος για συγκρούσεις ανάμεσα στα αυτοκίνητα και τα εμπόδια χρησιμοποιώντας την intersects() και την getGlobalBounds(). Το παιχνίδι περιλαμβάνει απλό HUD με χρονόμετρο το οποίο

ενημερώνεται με sf::Clock και προβάλλεται με sf::Text, ενώ παρουσιάζονται και μηνύματα που ενημερώνουν για την έκβαση του παιχνιδιού, δηλαδή ήττα. Στον κώδικα του παιχνιδιού δεν χρησιμοποιούνται περίπλοκες δομές ή αλγόριθμοι. Τα εμπόδια προκύπτουν μέσω vector από sprites και οι έλεγχοι βασίζονται σε if statements . Η τυχαία θέση αυτη των εμποδίων γίνεται με χρήση της srand() και της rand(). Το σύνολο της υλοποίησης υπάρχει σε ένα μόνο αρχείο main.cpp χωρίς επιμέρους αρχεία για κλάσεις ή modules και ο κώδικας είναι άμεσα εκτελέσιμος. Για την εκτέλεση απαιτείται η βιβλιοθήκη SFML και η εκκίνηση του προγράμματος μπορεί να πραγματοποιηθεί είτε μέσω του Visual Studio, με τον συνδυασμό πλήκτρων Ctrl + F5 ή με το αντίστοιχο κουμπί εκτέλεσης, είτε απευθείας με την εκτέλεση του παραγόμενου αρχείου .exe που βρίσκεται στον φάκελο του έργου. Ο φάκελος του έργου θα περιλαμβάνει έναν φάκελο με όνομα assets, ο οποίος θα περιέχει όλες τις εικόνες, τους ήχους και τις γραμματοσειρές που χρησιμοποιούνται στο παιχνίδι. Αυτο επιτρέπει την άμεση εκτέλεση του κώδικα χωρίς να απαιτούνται επιπλέον βήματα ή αλλαγές στις διαδρομές των αρχείων.

6.Αποτελέσματα

Ο τελικός κώδικας του παιχνιδιού μας "Drunk Drivers", καθώς και ο τρόπος εκτέλεσής του, έχουν παραδοθεί στον φάκελο με όνομα code.

7. Comparison with AI generated code

//ai generated code :

Code generated by ChatGPT with the command: 'Give me a C++ code where the player controls a car that avoids oncoming obstacles, with textures and audio output.

```
#include <SFML/Graphics.hpp>
#include <SFML/Audio.hpp>
#include <cstdlib>
#include <ctime>

int main() {
    sf::RenderWindow window(sf::VideoMode(800, 600), "Car Dodge");
    window.setFramerateLimit(60);
```

```
// Load textures
sf::Texture carTex, obsTex;
carTex.loadFromFile("assets/car.png");
obsTex.loadFromFile("assets/obstacle.png");

sf::Sprite car(carTex);
car.setPosition(375, 500);

sf::Sprite obstacle(obsTex);
float obsY = -100;
float obsSpeed = 5;
std::srand(std::time(nullptr));
obstacle.setPosition(std::rand() % 720, obsY);

// Sound
sf::SoundBuffer crashBuf;
crashBuf.loadFromFile("assets/crash.mp3");
sf::Sound crashSound(crashBuf);

while (window.isOpen()) {
    sf::Event e;
    while (window.pollEvent(e))
        if (e.type == sf::Event::Closed)
            window.close();

    // Car movement
    if (sf::Keyboard::isKeyPressed(sf::Keyboard::Left) && car.getPosition().x > 0)
        car.move(-6, 0);
```

```
        if (sf::Keyboard::isKeyPressed(sf::Keyboard::Right) && car.getPosition().x <
740)
            car.move(6, 0);

        // Move
        obsY += obsSpeed;
        if (obsY > 600) {
            obsY = -100;
            obstacle.setPosition(std::rand() % 720, obsY);
        }
        obstacle.setPosition(obstacle.getPosition().x, obsY);

        // Collision
        if (car.getGlobalBounds().intersects(obstacle.getGlobalBounds())) {
            crashSound.play();
            sf::sleep(sf::seconds(1));
            window.close();
        }

        // Draw
        window.clear();
        window.draw(car);
        window.draw(obstacle);
        window.display();
    }
    return 0;
}
```

8. Conclusions & Lessons learned

Ο σχεδιασμός και η ανάπτυξη ενός 2D παιχνιδιού από την αρχή κρίνεται ιδιαίτερα απαιτητική αλλά συγχρόνως αποτελεί και μία δημιουργική διαδικασία. Η ενασχόληση με το project <<Drunk Drivers>> συνέβαλε στη βαθύτερη κατανόηση σημαντικών προγραμματιστικών εννοιών, καθώς εφαρμόστηκαν σε πρακτικό επίπεδο. Ταυτόχρονα, χρειάστηκε να αντλήσουμε πληροφορίες από διαφορετικές πηγές με αποτέλεσμα να διευρύνουμε τις γνώσεις μας τόσο στον Αντικειμενοστραφή Προγραμματισμό όσο και στη χρήση καινούργιων εργαλείων. Επιπλέον, το project μας έδωσε την ευκαιρία να ασχοληθούμε με εφαρμογές που μπορεί να μην είχαμε ανακαλύψει στο παρελθόν, όπως το visual studio, το photoshop και το audiomass. Μάθαμε, ακόμη, να διαχειριζόμαστε καλύτερα τον χρόνο και τους πόρους αποδίδοντας τους κατάλληλους ρόλους στα μέλη της ομάδας. Μέσα από αυτή τη διαδικασία αναπτύξαμε δεξιότητες επίλυσης προβλημάτων, καθώς συχνά έπρεπε να αντιμετωπίσουμε τεχνικά ζητήματα και να προσαρμοστούμε σε νέα δεδομένα. Είναι σημαντικό να αναφερθεί πως καλλιεργήσαμε την ομαδικότητα και τη συνεργασία, στοιχεία που θεωρούνται απαραίτητα. Τέλος, η συγκεκριμένη εργασία αποτέλεσε κίνητρο για εμάς ώστε να ασχοληθούμε περαιτέρω μελλοντικά με το game development και γενικότερα με τον Προγραμματισμό.

9. Πηγες

Διάφορα tutorials και ιστοσελίδες που συνέβαλαν στην ανάπτυξη του παιχνιδιού

How to do Rendering using C++ - From installing the tools to implementation

https://youtu.be/t0z3RojiKFg?si=XF0nUNpAnUw1RHn2

Creating a Window using C++ and Win32 | Tutorial

https://youtu.be/Kx5CN-V6FvQ?si=EN2EeZBVUTSpcGbv

C++ GUI Programming For Beginners | Episode 2 - Creating a Window

https://youtu.be/cQalRGqRRp4?si=uTI6Izb9iCVnF6L8

SFML

https://www.sfml-dev.org

sfml Opening a window

https://www.sfml-dev.org/tutorials/3.0/window/window/