Προγραμματιστική Άσκηση 2 2024

Γραφικά Υπολογιστών και Συστήματα Αλληλεπίδρασης

Δημούδης Γεώργιος 5212

Στρούγγης Γεώργιος 5357

20 Δεκεμβρίου 2024

1: Περιγραφή υλοποίησης των ερωτημάτων:

Ερώτημα 1:

Δημιουργήσαμε νέο project με όνομα “Project 2 – Treasure Bob” το οποίο αφου τελειώσαμε κάναμε build σε ξεχωριστό αρχείο ώστε να έχουμε την .exe εφαρμογή. Η ανάλυση του ορίστηκε αυτόματα από το unity ως 1024x768, για αυτό δεν χρειάστηκε κάποια αλλαγή από εμάς.

Ο λαβύρινθος σχεδιάστηκε μέσω ενός αντικειμένου plane αρχικών διαστάσεων 10x10 το οποίο μεγεθύναμε επι 10 στις παραμέτρους x και z του scale στο unity editor ώστε να αποκτήσει διαστάσεις 100x100 και του δόθηκε η υφή floor.jpg. Στη συνέχεια προστέθηκαν τετράγωνα/παραλληλόγραμμα που αποτέλεσαν τους τείχους του λαβυρίνθου. Τα τετράγωνα είχαν διαστάσεις 10x10x10 ενώ τα ορθογώνια είχαν ύψος 10 αλλά στους άλλους άξονες ήταν πιο μακριά. Σε όλα δόθηκε μπλε χρώμα.

Ερώτημα 2:

Δημιουργήσαμε σφαίρα διαστάσεων 7x7x7 στην οποία προσθέσαμε υφή bob.jpg. Ο κώδικας που εφαρμόσαμε για να ελέγξουμε την κίνηση και συναναστροφή του παίκτη με τον λαβύρινθο φαίνεται παρακάτω στο script MoveSphere.cs

Αρχικά ορίσαμε πίνακα 10x10 βασισμένο στις προηγούμενες ασκήσεις που να μαρκάρει τις θέσεις τω τοίχων με 1 και τον κενό χώρο με 0. Όσον αφορά τον πίνακα, η αλλαγή της τιμής στον άξονα ζ κινεί το παίκτη πάνω-κάτω μεταξύ των υποπινάκων ενώ η αλλαγή της τιμής στον άξονα χ κινεί τον παίκτη αριστερά-δεξιά μέσα σε έναν υποπίνακα. Κρατάμε τις αρχικές τιμές της θέσης του πίνακα που είναι 0 στον άξονα χ και 2 στον άξονα ζ ώστε να αρχικοποιείται στην πάνω αριστερή είσοδο του πίνακα, καθώς και την θέση του ζ σε έναν πίνακα καθρεπτισμένο ως πως τον άξονα χ ο οποίος χρησιμοποιείται στο TeleportPrize για τους θησαυρούς.

Όσων αφορά των λαβύρινθο, η θέση που θέλουμε να εμφανίζεται ο παίκτης σύμφωνα με το unity είναι χ=-45 ζ=25 που ορίζονται στα initx και initz. Στη συνέχεια προσθέτουμε ως GameObject όλους τους θησαυρούς και τις παγίδες θανάτου για αργότερα ερωτήματα και θέτουμε score = 0 για το bonus που θα αναφέρουμε αργότερα. Τέλος στο Start() ορίζουμε τα αντικείμενα με τα οποία ο παίκτης θα συναναστρέφεται καθώς και ορίζουμε την αρχική του θέση ως Vector3(initx, 8.5f, initz);

Initx, initz εξηγήσαμε πιο πάνω αλλά 8.5f στον άξονα y είναι ώστε ο παίκτης να μην είναι βυθισμένος στο πάτωμα αλλά να βρίσκεται πάνω του.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

using UnityEngine.SceneManagement; // For reloading the scene
using UnityEngine.UI;

public class MoveSphere : MonoBehaviour
{
    // Board representing the maze
    int[,] board = { // board[z axis, x axis]
        {1,1,1,1,1,1,1,1,1,1},
        {1,0,0,0,0,0,0,0,0,1},
        {0,0,1,1,1,1,0,1,0,1},
        {1,0,1,0,0,0,0,1,0,1},
        {1,0,1,0,1,1,0,1,0,1},
        {1,0,0,0,0,1,0,0,0,1},
        {1,0,1,1,0,1,1,1,0,1},
        {1,0,0,0,0,0,0,1,0,0},
        {1,0,1,0,1,1,0,0,0,1},
        {1,1,1,1,1,1,1,1,1,1}
    };

    int x = 0;
    int z = 0;
    public int posx = 0;
    int posz = 2;
    public int posz2 = 7; // Position in the TeleportPrize flipped board
    float initx = -45.0f;
    float initz = 25.0f;

    public GameObject PrizeCherry;
    TeleportPrize PrizeCherry_Script;

    public GameObject PrizeOrange;
    TeleportPrizeOrange PrizeOrange_Script;

    public GameObject PrizeLemon;
    TeleportPrizeLemon PrizeLemon_Script;

    public GameObject death;
    TeleportDeath death_Script;

    public GameObject death2;
    TeleportDeath2 death_Script2;

    int score = 0;

    void Start()
    {
        transform.position = new Vector3(initx, 8.5f, initz);
        PrizeCherry_Script = PrizeCherry.GetComponent<TeleportPrize>();
        PrizeOrange_Script = PrizeOrange.GetComponent<TeleportPrizeOrange>();
        PrizeLemon_Script = PrizeLemon.GetComponent<TeleportPrizeLemon>();
        death_Script = death.GetComponent<TeleportDeath>();
        death_Script2 = death2.GetComponent<TeleportDeath2>();
    }
```

Στο Update() γίνεται έλεγχος για τα πλήκτρα που πάτησε ο χρήστης. Αν το πρόγραμμα κρίνει ότι η θέση στην οποία ο παίκτης πάει να κινηθεί δεν έχει τιμή 1 στον πίνακα (δηλαδή δεν έχει τοίχο) τότε ενημερώνει την θέση του παίκτη στον λαβύρινθο αυξομειώνοντας κατά 10 εξ αίτιας των διαστάσεων των αντικειμένων και κατά 1 στον πίνακα για μελλοντικές κινήσεις. Επίσης η θέση του παίκτη αλλάζει και στον καθρεπτισμένο πίνακα που αναφέραμε νωρίτερα. Μέσω του Vector3(initx + x, 8.5f, initz + z); οι νέες συντεταγμένες προ θέτονται στις παλιές για να αλλάξει θέση το αντικείμενο.

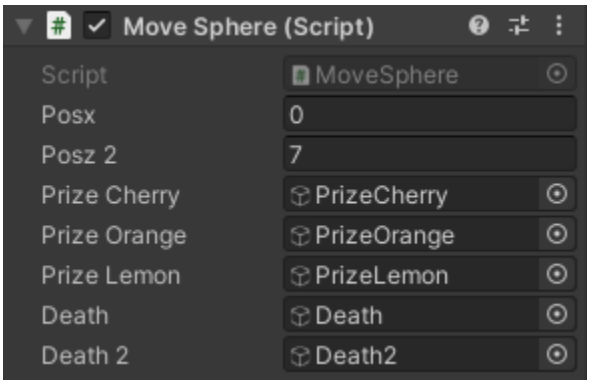
Η χρησιμότητα των getScore και touch θα αναφερθεί παρακάτω. Τέλος, ο λόγος που κάποιες μεταβλητές είναι public είναι επειδή θα τις χρειαστούν άλλα script που συναναστρέφονται με τον παίκτη.

```
    void Update()
    {
        if ((Input.GetKeyDown(KeyCode.J)) && (board[posz, posx - 1] == 0))
        {
            posx -= 1;
            x -= 10;
        }
        else if ((Input.GetKeyDown(KeyCode.L)) && (board[posz, posx + 1] == 0))
        {
            posx += 1;
            x += 10;
        }
        else if ((Input.GetKeyDown(KeyCode.I)) && (board[posz - 1, posx] == 0))
        {
            posz -= 1;
            posz2 += 1;
            z += 10;
        }
        else if ((Input.GetKeyDown(KeyCode.K)) && (board[posz + 1, posx] == 0))
        {
            posz += 1;
            posz2 -= 1;
            z -= 10;
        }

        transform.position = new Vector3(initx + x, 8.5f, initz + z);

        touch();
    }

    public int getScore() {
    return score;
    }

```

```
void touch()
{
    if ((posz2 == PrizeCherry_Script.poszp) && (posx == PrizeCherry_Script.posxp))
    {
    score += PrizeCherry_Script.getPoints();
        PrizeCherry_Script.contact();
    }

    else if ((posz2 == PrizeOrange_Script.poszp) && (posx == PrizeOrange_Script.posxp))
    {
    score += PrizeOrange_Script.getPoints();
        PrizeOrange_Script.contact();
    }

    else if ((posz2 == PrizeLemon_Script.poszp) && (posx == PrizeLemon_Script.posxp))
    {
    score += PrizeLemon_Script.getPoints();
        PrizeLemon_Script.contact();
    }

    else if (death_Script != null && (posz2 == death_Script.poszp) && (posx == death_Script.posxp))
    {
        death_Script.contact();
    }

    else if (death_Script2 != null && (posz2 == death_Script2.poszp) && (posx == death_Script2.posxp))
    {
        death_Script2.contact();
    }
}
```

Είναι σημαντικό επίσης να αναφέρουμε ότι ώστε να μπορεί ο παίκτης να αναγνωρίσει για

ποιες οντότητες συναναστρέφεται έπρεπε αφού του δώσαμε το script στην unity να ορίσουμε τις άλλες οντότητες σε πλαίσια.



Ερώτημα 3:

Πρώτα από όλα φτιάχτηκαν τρία τετράγωνα μεγέθους 5x5x5 οπού τους δώθηκαν οι υφές cheery.jpg, orange.jpg και lemon.jpg.

Για τους θησαυρούς τύπου κεράσι, λεμόνι, πορτοκάλι υλοποιήθηκαν τα script TeleportPrize, TeleportPrizeLemon, TeleportPrizeOrange αντίστοιχα. Αρχική ιδέα ήταν να υπάρχει ένα script για όλα, αλλά ώστε να μην τηλεμεταφέρονται το ένα πάνω από το άλλο αποφασίστηκε το κάθε ένα να έχει ένα ξεχωριστό αλλά παρόμοιο script.

Αρχικά υλοποιήθηκε πίνακας 10x10 ο οποίος όπως αναφέρθηκε παραπάνω είναι καθρεπτισμένος ως προς τον άξονα χ, αυτό έγινε επειδή ο προηγούμενος πίνακας δεν δούλευε για αναζήτηση ελεύθερης θέσης στον λαβύρινθο κατάλληλα εξ αιτίας των συντεταγμένων. Η θέση του θησαυρού ορίζεται μέσα στον πίνακα και αρχικά δημιουργείται μέσα σε έναν τοίχο για να κρυφτεί πριν οριστούν οι συντεταγμένες του. Ορίζουμε όλες τις οντότητες με τις οποία θα συναναστραφεί και κάνουμε public οποιοδήποτε πεδίο θα χρειαστούν αυτές οι οντότητες για να καθορίσουν τις δικές τους κινήσεις και ορίζουμε των χρόνο μεταξύ των εμφανίσεων, μια boolean τιμή η οποία αρχικά ορίζεται false και θα αποφασίσει αν (αφού ο παίκτης έχει αγγίξει τον θησαυρό) θα συρρικνωθεί στο μισό μέγεθος πριν εξαφανιστεί. Επίσης ορίζουμε διαφορετικούς πόντους για κάθε θησαυρό.

Στο start() αρχικοποιούμε τις άλλες οντότητες και κάλουμε το teleport για την αρχική εμφάνιση πριν αρχίσουμε να μετράμε χρόνο.

Στην update() εφόσον δεν έχει αγγιχτεί από το παίκτη και το shrink παραμένει false θα ελέγχει αν έχουν περάσει πέντε δευτερόλεπτα όπου θα καλλεί την teleport για να αλλάξει θέσει στον πίνακα και λαβύρινθο. Η μέθοδος teleport διαλέγει τυχαίες συντεταγμένες και

εφόσον δεν συμπίπτουν με τοίχο, τον παίκτη, άλλο θησαυρό η παγίδα θανάτου αλλάζει τις συντεταγμένες του θησαυρού ώστε να εμφανιστεί εκεί. Ποιές οντότητες θα ελέγχει εξαρτάτε από ποιος είναι ο ίδιος ο θησαυρός.

Όπως αναφέρθηκε παραπάνω το script MoveSphere περιέχει την μέθοδο touch η οποία παίρνει τις συντεταγμένες των θησαυρών και αν συμπίπτουν καλεί την μέθοδο contact. Η contact αλλάζει την τιμή του score σε 0 ώστε να μη συνεχίσει να προστίθεται και του shrink σε true ώστε στην επόμενη κλήση του update αντί για να αλλάξει την θέση του θησαυρού θα αλλάξει τις διαστάσεις του σε 2.5x2.5x2.5. Αφότου περάσει λίγος χρόνος ακόμα ώστε ο παίκτης να μπορεί να δει τη σμίκρυνση θα καλέσει Destroy(GameObject) ώστε να το εξαφανίσει ολοκληρωτικά.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class TeleportPrize : MonoBehaviour
{
    int[,] board = {      //board is board1 flipped on the x axis compared to the one in MoveSphere
        {1,1,1,1,1,1,1,1,1,1},
        {1,0,1,0,1,1,0,0,0,1},
        {1,0,0,0,0,0,0,1,0,0},
        {1,0,1,1,0,1,1,1,0,1},
        {1,0,0,0,0,1,0,0,0,1},
        {1,0,1,0,1,1,0,1,0,1},
        {1,0,1,0,0,0,0,1,0,1},
        {0,0,1,1,1,1,0,1,0,1},
        {1,0,0,0,0,0,0,0,0,1},
        {1,1,1,1,1,1,1,1,1,1},
    };

    public int posxp = 0;
    public int poszp = 0;

    float initx = -45.0f;
    float initz = -45.0f;

    public GameObject PlayerBob;
    MoveSphere PlayerBob_Script;

    public GameObject PrizeOrange;
    TeleportPrizeOrange PrizeOrange_Script;

    public GameObject PrizeLemon;
    TeleportPrizeLemon PrizeLemon_Script;

    public GameObject death;
    TeleportDeath death_Script;

    public GameObject death2;
    TeleportDeath2 death_Script2;

    float timer = 0f;
    float teleportInterval = 5f;

    bool shrink = false;

    Vector3 newSize = new Vector3(2.5f, 2.5f, 2.5f);

    public int points = 1;
```

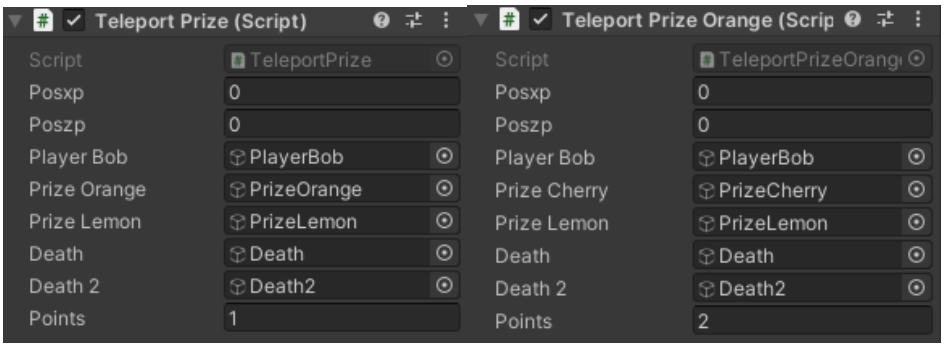
```
    void Start()
    {
        PlayerBob_Script = PlayerBob.GetComponent<MoveSphere>();
        PrizeOrange_Script = PrizeOrange.GetComponent<TeleportPrizeOrange>();
        PrizeLemon_Script = PrizeLemon.GetComponent<TeleportPrizeLemon>();
        death_Script = death.GetComponent<TeleportDeath>();
        death_Script2 = death2.GetComponent<TeleportDeath2>();

        teleport();
        timer = Time.time;
    }

    void Update()
    {
        if (shrink == false)
        {
            if (Time.time >= timer + teleportInterval)
            {
                teleport();
                timer = Time.time;
            }
        }
        else{
            transform.localScale = Vector3.Lerp(transform.localScale, newSize, 1);
            if (Time.time >= timer + 2f)
            {
                Destroy(gameObject);
            }

        }
    }

    void teleport(){
        do{
            posxp = Random.Range(0, 10);
            poszp = Random.Range(0, 10);
        } while ((board[poszp, posxp] != 0) || ((PlayerBob_Script.posz2 == poszp) && (PlayerBob_Script.posx == posxp))
        || ((PrizeOrange_Script.poszp == poszp) && (PrizeOrange_Script.posxp == posxp)) || ((PrizeLemon_Script.poszp == poszp) && (PrizeLemon_Script.posxp == posxp))
        || ((death_Script.posxp == posxp) && (death_Script.poszp == poszp))|| ((death_Script2.posxp == posxp) && (death_Script2.poszp == poszp)));

        transform.position = new Vector3(initx + posxp * 10, 8.5f, initz + poszp * 10);
    }

public int getPoints(){
    return points;
}
    public void contact()
    {
    points = 0;
        shrink = true;
    timer = Time.time;
    }
}
```

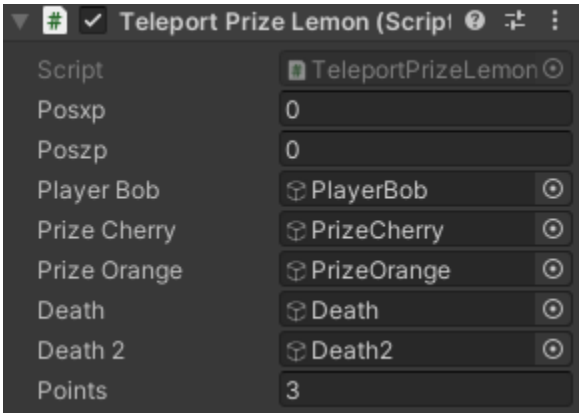
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class TeleportPrizeLemon : MonoBehaviour
{
    int[,] board = {      //board is board1 flipped on the x axis compared to the one in MoveSphere
        {1,1,1,1,1,1,1,1,1,1},
        {1,0,1,0,1,1,0,0,0,1},
        {1,0,0,0,0,0,0,1,0,0},
        {1,0,1,1,0,1,1,1,0,1},
        {1,0,0,0,0,1,0,0,0,1},
        {1,0,1,0,1,1,0,1,0,1},
        {1,0,1,0,0,0,0,1,0,1},
        {0,0,1,1,1,1,0,1,0,1},
        {1,0,0,0,0,0,0,0,0,1},
        {1,1,1,1,1,1,1,1,1,1},
    };

    public int posxp = 0;
    public int poszp = 0;

    float initx = -45.0f;
    float initz = -45.0f;

    public GameObject PlayerBob;
    MoveSphere PlayerBob_Script;

    public GameObject PrizeCherry;
    TeleportPrize PrizeCherry_Script;

    public GameObject PrizeOrange;
    TeleportPrizeOrange PrizeOrange_Script;

    public GameObject death;
    TeleportDeath death_Script;

    public GameObject death2;
    TeleportDeath2 death_Script2;

    float timer = 0f;
    float teleportInterval = 5f;

    bool shrink = false;

    Vector3 newSize = new Vector3(2.5f, 2.5f, 2.5f);

    public int points = 3;
```

```
    void Start()
    {
        PlayerBob_Script = PlayerBob.GetComponent<MoveSphere>();
        PrizeCherry_Script = PrizeCherry.GetComponent<TeleportPrize>();
        PrizeOrange_Script = PrizeOrange.GetComponent<TeleportPrizeOrange>();
    death_Script = death.GetComponent<TeleportDeath>();
    death_Script2 = death2.GetComponent<TeleportDeath2>();
        teleport();
        timer = Time.time;
    }

    // Update is called once per frame
    void Update()
    {
        if (shrink == false)
        {
            if (Time.time >= timer + teleportInterval)
            {
                teleport();
                timer = Time.time;
            }
        }
        else
        {
            transform.localScale = Vector3.Lerp(transform.localScale, newSize, 1);
            if (Time.time >= timer + 2f)
            {
                Destroy(gameObject);
            }

        }
    }

public int getPoints(){
    return points;
}

    void teleport()
    {
        do
        {
            posxp = Random.Range(0, 10);
            poszp = Random.Range(0, 10);
        } while ((board[poszp, posxp] != 0) || ((PlayerBob_Script.posz2 == poszp) && (PlayerBob_Script.posx == posxp))
        || ((PrizeCherry_Script.poszp == poszp) && (PrizeCherry_Script.posxp == posxp)) || ((PrizeOrange_Script.poszp == poszp) && (PrizeOrange_Script.posxp == posxp))
        || ((death_Script.posxp == posxp) && (death_Script.poszp == poszp)) || ((death_Script2.posxp == posxp) && (death_Script2.poszp == poszp)));
        transform.position = new Vector3(initx + posxp * 10, 8.5f, initz + poszp * 10);
    }

    public void contact()
    {
    points = 0;
        shrink = true;
    timer = Time.time;
    }
}
```

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class TeleportPrizeOrange : MonoBehaviour
{
    int[,] board = {      //board is board1 flipped on the x axis compared to the one in MoveSphere
        {1,1,1,1,1,1,1,1,1,1},
        {1,0,1,0,1,1,0,0,0,1},
        {1,0,0,0,0,0,0,1,0,0},
        {1,0,1,1,0,1,1,1,0,1},
        {1,0,0,0,0,1,0,0,0,1},
        {1,0,1,0,1,1,0,1,0,1},
        {1,0,1,0,0,0,0,1,0,1},
        {0,0,1,1,1,1,0,1,0,1},
        {1,0,0,0,0,0,0,0,0,1},
        {1,1,1,1,1,1,1,1,1,1},
    };

    public int posxp = 0;
    public int poszp = 0;

    float initx = -45.0f;
    float initz = -45.0f;

    public GameObject PlayerBob;
    MoveSphere PlayerBob_Script;

    public GameObject PrizeCherry;
    TeleportPrize PrizeCherry_Script;

    public GameObject PrizeLemon;
    TeleportPrizeLemon PrizeLemon_Script;

    public GameObject death;
    TeleportDeath death_Script;

    public GameObject death2;
    TeleportDeath2 death_Script2;

    float timer = 0f;
    float teleportInterval = 5f;

    bool shrink = false;

    Vector3 newSize = new Vector3(2.5f, 2.5f, 2.5f);

    public int points = 2;
```

```
    void Start()
    {
        PlayerBob_Script = PlayerBob.GetComponent<MoveSphere>();
        PrizeCherry_Script = PrizeCherry.GetComponent<TeleportPrize>();
        PrizeLemon_Script = PrizeLemon.GetComponent<TeleportPrizeLemon>();
    death_Script = death.GetComponent<TeleportDeath>();
    death_Script2 = death2.GetComponent<TeleportDeath2>();
        teleport();
        timer = Time.time;
    }

    void Update()
    {
        if (shrink == false)
        {
            if (Time.time >= timer + teleportInterval)
            {
                teleport();
                timer = Time.time;
            }
        }
        else
        {
            transform.localScale = Vector3.Lerp(transform.localScale, newSize, 1);
            if (Time.time >= timer + 2f)
            {
                Destroy(gameObject);
            }

        }
    }

public int getPoints(){
    return points;
}

    void teleport()
    {
        do
        {
            posxp = Random.Range(0, 10);
            poszp = Random.Range(0, 10);
        } while ((board[poszp, posxp] != 0) || ((PlayerBob_Script.posz2 == poszp) && (PlayerBob_Script.posx == posxp))
        || ((PrizeCherry_Script.poszp == poszp) && (PrizeCherry_Script.posxp == posxp)) || ((PrizeLemon_Script.poszp == poszp) && (PrizeLemon_Script.posxp == posxp))
        || ((death_Script.posxp == posxp) && (death_Script.poszp == poszp)) || ((death_Script2.posxp == posxp) && (death_Script2.poszp == poszp)));
        transform.position = new Vector3(initx + posxp * 10, 8.5f, initz + poszp * 10);
    }

    public void contact()
    {
    points = 0;
        shrink = true;
    timer = Time.time;
    }
}
```





Ερώτημα 4 + bonus γ και δ:

Οι παγίδες θανάτου σχεδιάστηκαν ως σφαίρες μεγέθους 5x5x5 και τους δόθηκε η υφή death.jpg.

Για τα scripts των παγίδων θανάτου, η λογική είναι παρόμοια με των θησαυρών, και προφανώς τα Death 1 και Death 2 έχουν πάρα πολύ παρόμοιο κώδικα με τη διαφορά ότι όπου έχουμε death2 στο death1, στο death 2 θα έχουμε death. Όσα λέω είναι για το death.

Με τα Prizes, έχουν τη διαφορά ότι δεν ξεκινούν στο board στην αρχή, (δηλαδή είναι invisible και χωρίς collider), και εμφανίζονται μετά από ένα τυχαίο διάστημα μεταξύ 1 και 4 δευτερολέπτων, και παραμένουν εμφανή για 2-6 δευτερόλεπτα. Στην αρχή καθορίζεται επίσης η ώρα που θα πάρουν για να εμφανιστούν. Επίσης παίρνει τα scripts των υπόλοιπων αντικειμένων ώστε να μην κάνουν teleport το ένα πάνω στο άλλο.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class TeleportDeath : MonoBehaviour
{
    int[,] board = {
        {1,1,1,1,1,1,1,1,1,1},
        {1,0,1,0,1,1,0,0,0,1},
        {1,0,0,0,0,0,0,1,0,0},
        {1,0,1,1,0,1,1,1,0,1},
        {1,0,0,0,0,1,0,0,0,1},
        {1,0,1,0,1,1,0,1,0,1},
        {1,0,1,0,0,0,0,1,0,1},
        {0,0,1,1,1,1,0,1,0,1},
        {1,0,0,0,0,0,0,0,0,1},
        {1,1,1,1,1,1,1,1,1,1},
    };

    public int posxp = 0;
    public int poszp = 0;

    float initx = -45.0f;
    float initz = -45.0f;

    public GameObject PlayerBob;
    private MoveSphere PlayerBob_Script;

    public GameObject PrizeCherry;
    private TeleportPrize PrizeCherry_Script;

    public GameObject PrizeLemon;
    private TeleportPrizeLemon PrizeLemon_Script;

    public GameObject PrizeOrange;
    private TeleportPrizeOrange PrizeOrange_Script;

    public GameObject death2;
    TeleportDeath2 death_Script2;

    private Renderer trapRenderer;
    private Collider trapCollider;

    public GameOverManager gameOverManager;

    private float teleportTimer = 0f;
    private float visibilityTimer = 0f;
    private bool isVisible = false;

    void Start()
    {
        PlayerBob_Script = PlayerBob.GetComponent<MoveSphere>();
        PrizeOrange_Script = PrizeOrange.GetComponent<TeleportPrizeOrange>();
        PrizeLemon_Script = PrizeLemon.GetComponent<TeleportPrizeLemon>();
        PrizeCherry_Script = PrizeCherry.GetComponent<TeleportPrize>();
        death_Script2 = death2.GetComponent<TeleportDeath2>();

        trapRenderer = GetComponent<Renderer>();
        trapCollider = GetComponent<Collider>();

        trapRenderer.enabled = false;
        trapCollider.enabled = false;
        SetNextTeleportTime();
    }
```

Η update ελέγχει αν ήρθε η ώρα για το teleport και την εμφάνιση και πράττει αντίστοιχα. Όταν έρχεται η ώρα να κρύψουμε τον παίκτη θέτουμε τα posxp = 0, επειδή αλλιώς (παρόλο

που έχουμε κλείσει τα colliders) ο έλεγχος που γίνεται στον bob είναι για τα positions. Και ο bob δεν μπορεί να πάει ποτέ στο 0,0, οπότε ξέρουμε πως δεν θα συγκρουστούν.

```
void Update()
{
    teleportTimer -= Time.deltaTime;

    if (!isVisible)
    {
        if (teleportTimer <= 0f)
        {
            teleport();
            isVisible = true;
            trapRenderer.enabled = true;
            trapCollider.enabled = true;
            SetNextVisibilityTime();
        }
    }
    else
    {
        visibilityTimer -= Time.deltaTime;
        if (visibilityTimer <= 0f)
        {
            isVisible = false;
            trapRenderer.enabled = false;
            trapCollider.enabled = false;
            posxp = 0;
            poszp = 0;
            SetNextTeleportTime();
        }
    }
}
```

Η συνάρτηση teleport είναι η ίδια με των φρούτων.

```
void teleport()
{
    do
    {
        posxp = Random.Range(0, 10);
        poszp = Random.Range(0, 10);
    } while (
        board[poszp, posxp] != 0 ||
        (PlayerBob_Script.posz2 == poszp && PlayerBob_Script.posx == posxp) ||
        (PrizeOrange_Script.poszp == poszp && PrizeOrange_Script.posxp == posxp) ||
        (PrizeLemon_Script.poszp == poszp && PrizeLemon_Script.posxp == posxp) ||
        (PrizeCherry_Script.poszp == poszp && PrizeCherry_Script.posxp == posxp) ||
        (death_Script2.posxp == posxp && death_Script2.poszp == poszp)
    );

    transform.position = new Vector3(initx + posxp * 10, 8.5f, initz + poszp * 10);
}
```

Οι setTeleportTime και SetVisibilityNext είναι υπεύθυνα για να βρίσκουν τα timers για τις επόμενες κινήσεις των παγίδων. Τέλος, όταν ο παίκτης αγγίξει την παγίδα, καλείται το GameOverManager ώστε να τερματίσει το παιχνίδι καθώς και να δείξει πόσους πόντους ο παίκτης μάζεψε.

```
void SetNextTeleportTime()
{
    teleportTimer = Random.Range(1f, 4f);
}

void SetNextVisibilityTime()
{
    visibilityTimer = Random.Range(2f, 6f);
}

public void contact()
    {
    gameOverManager.ShowGameOverScreen();
    }
```

Μόλις καλεστεί το GameOverManager, το ui που βάλαμε εμφανίζεται στον χρήστη.

Μόλις εμφανιστεί παίρνει το finalScore από τον παίκτη και το προβάλει δίπλα στο μήνυμα ότι τελείωσε το παιχνίδι.

Επίσης καλείται η freeze για να κάνει freeze τα script όλων των αντικειμένων μου όταν τελειώσει το παιχνίδι, το οποίο συνεπώς να σταματάει τα πάντα. Πρώτα όμως ελέγχει αν υπάρχουν τα scripts. Αν δεν το κάναμε αυτό πετούσε error η unity κάτω δεξιά που έλεγε ότι το αντικείμενο είχε καταστραφεί, αλλά συνεχίζαμε να προσπαθούμε να το κάνουμε access. Η freeze είναι συνάρτηση που κάνει ακριβώς αυτό.

```
using UnityEngine;
using UnityEngine.UI;

public class GameOverManager : MonoBehaviour
{
    public GameObject gameOverPanel;
    public Text scoreText;
    private MoveSphere playerScript;

    // References to the Prize scripts for Lemon, Orange, and Cherry
    private TeleportPrize CherryScript;
    private TeleportPrizeOrange OrangeScript;
    private TeleportPrizeLemon LemonScript;

    void Start()
    {
        GameObject player = GameObject.Find("PlayerBob");
        GameObject lemon = GameObject.Find("PrizeLemon");
        GameObject orange = GameObject.Find("PrizeOrange");
        GameObject cherry = GameObject.Find("PrizeCherry");

        if (player != null)
        {
            playerScript = player.GetComponent<MoveSphere>();
        }
        else
        {
            Debug.LogError("PlayerBob not found! Make sure the player is named 'PlayerBob'.");
        }

        if (lemon != null)
        {
            LemonScript = lemon.GetComponent<TeleportPrizeLemon>();
        }

        if (cherry != null)
        {
            CherryScript = cherry.GetComponent<TeleportPrize>();
        }

        if (orange != null)
        {
            OrangeScript = orange.GetComponent<TeleportPrizeOrange>();
        }

        if (gameOverPanel != null)
        {
            gameOverPanel.SetActive(false);
        }
    }
```

```
    public void ShowGameOverScreen()
    {
        if (playerScript != null)
        {
            gameOverPanel.SetActive(true);

            int finalScore = playerScript.getScore();

            scoreText.text = "FINISHED SCORE " + finalScore;

            FreezeGame();
        }
        else
        {
            Debug.LogError("PlayerScript is not assigned.");
        }
    }

    private void FreezeGame()
    {
        Time.timeScale = 0;

        if (playerScript != null)
        {
            playerScript.enabled = false;
        }

        if (LemonScript != null)
        {
            LemonScript.enabled = false;
        }
        if (OrangeScript != null)
        {
            OrangeScript.enabled = false;
        }
        if (CherryScript != null)
        {
            CherryScript.enabled = false;
        }
    }
}
```

Όπως είπαμε οι πόντοι μαζεύονται από τον Bob κάθε φορά που ακουμπάει ένα από τα αντικείμενα. Τα αντικείμενα δίνουν 1,2 και 3 πόντους αντίστοιχα.

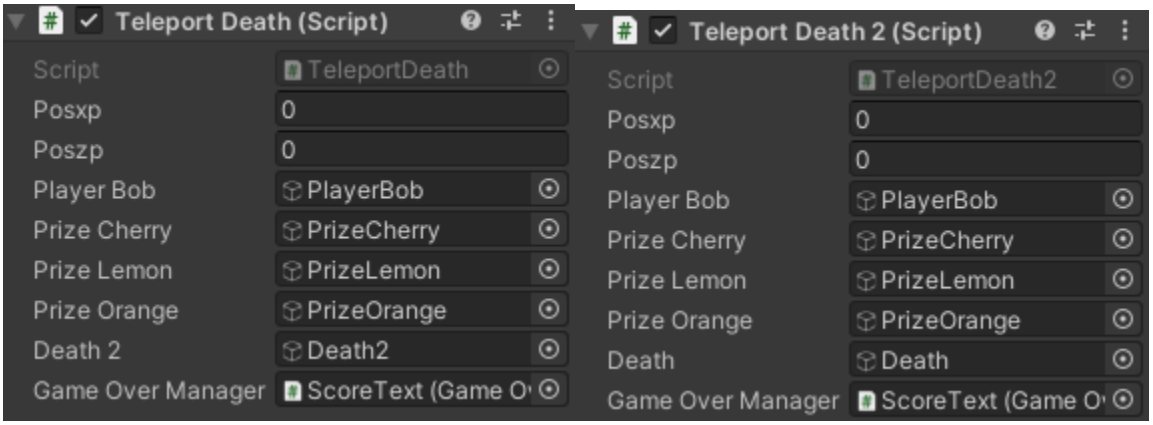
Το score του Bob γίνεται score += το score το οποίο παίρνεται από τα Prizes με μια public συνάρτηση getPoints (δηλαδή τα points του κάθε φρούτου). Το βασικό πρόβλημα που είχαμε ήταν ότι για κάθε frame που ακουμπούσε ο παίκτης το Prize, παίρναμε πόντους και τελειώναμε με πόντους στα 4,000.

```
void touch()
{
    if ((posz2 == PrizeCherry_Script.poszp) && (posx == PrizeCherry_Script.posxp))
    {
        score += PrizeCherry_Script.getPoints();
        PrizeCherry_Script.contact();
    }

    else if ((posz2 == PrizeOrange_Script.poszp) && (posx == PrizeOrange_Script.posxp))
    {
        score += PrizeOrange_Script.getPoints();
        PrizeOrange_Script.contact();
    }

    else if ((posz2 == PrizeLemon_Script.poszp) && (posx == PrizeLemon_Script.posxp))
    {
        score += PrizeLemon_Script.getPoints();
        PrizeLemon_Script.contact();
    }
```

```
public int getPoints(){
        return points;
}
```

Για να το λύσουμε αυτό, προσθέτουμε τα points την πρώτη φορά και μετά τα μηδενίζουμε ώστε να μην επηρεάζουν το αποτέλεσμα του προγράμματος. Για αυτό υπάρχει αυτό το points = 0 στην contact. Κοινώς, οι πόντοι προσθέτονται στην αρχή και μετά για κάθε άλλο frame που ακουμπάει το Prize, προσθέτεται 0 στο τελικό αποτέλεσμα του Bob.

```
public void contact()
{
    points = 0;
    shrink = true;
    timer = Time.time;
}
```



Ερώτημα 5:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class CameraController : MonoBehaviour
{
    float movementSpeed = 75f;

    float moveX;
    float moveZ;
    float moveY;

    // Start is called before the first frame update
    void Start()
    {
        transform.position = new Vector3(0.0f, 110.0f, -20.0f);
    }

    void Update()
    {
        //x and y axis movement
        moveX = 0f;
        moveY = 0f;

        if (Input.GetKey(KeyCode.UpArrow))
        {
            moveY += 1f;
        }
        else if (Input.GetKey(KeyCode.DownArrow))
        {
            moveY -= 1f;
        }
        else if (Input.GetKey(KeyCode.RightArrow))
        {
            moveX += 1f;
        }
        else if (Input.GetKey(KeyCode.LeftArrow))
        {
            moveX -= 1f;
        }

        //z axis movement
        moveZ = 0f;

        if (Input.GetKey(KeyCode.KeypadPlus))
        {
            moveZ = 1f;
        }
        else if (Input.GetKey(KeyCode.KeypadMinus))
        {
            moveZ -= 1f;
        }

        //rotate around the y axis
        if (Input.GetKey(KeyCode.R))
        {
            transform.Rotate(Vector3.up, movementSpeed * Time.deltaTime, Space.Self);
        }

        Vector3 moveDirection = new Vector3(moveX, moveY, moveZ) * movementSpeed * Time.deltaTime;
        transform.Translate(moveDirection, Space.Self);
    }
```

Το παραπάνω script υλοποιήθηκε για την κίνηση της κάμερας η οποία ορίζεται σε μια θέση ακριβώς από πάνω από το κέντρο του πίνακα ως αρχική θέση. Με τα πλήκτρα που μας ζητήθηκαν στην εκφώνηση μπορεί ο χρήστης να κουνηθεί πάνω, κάτω, αριστερά, δεξιά, να κάνει ζουμ πιο κοντά και πιο μακριά καθώς και να περιστρέφει ώστε να κοιτάξει τον ουρανό από διάφορες γωνίες. Η χρήση του Space.Self στην γραμμή 61 εγγυείται ότι η περιστροφή θα γίνει γύρω από την ίδια την κάμερα, που σημαίνει ότι όταν έχει περιστραφεί σε κάποια γωνία και ο χρήστης πατήσει κάποιο βέλος να κινηθεί η κάμερα αυτή θα κινηθεί με “ακολουθεί” τη νέα γωνία ως κέντρο των συντεταγμένων τις παρά την αρχική. Για

παράδειγμα, αν περιστραφεί 90 μοίρες για να κοιτάει δεξιά, τότε το αριστερό βέλος θα στείλει την κάμερα πιο κοντά στο πάτωμα, το δεξί πιο ψηλά στον ουρανό κτλ.

2: Πληροφορίες σχετικά με την υλοποίηση:

Και οι 2 φοιτητές χρησιμοποιούμε Windows 10, σε Unity 2020.3.24f1. Το εκτελέσιμο είναι στον φάκελο build.

Το checksum utility export δίνει αυτό:

File path and name: C:\Users\George\Desktop\grafikaAskisi2.zip

MD5: 96183FF345A3BD06D893CFEEAF13267A

SHA-1: D31E47DA43F31156EBBC61B6CBD0F487F91DF417

SHA-256:
2D99FAFA63CB5E5DE00DF82B65235B02170412B9849DDF2FF250A2532170F226

SHA-512:
61ADF170C74D0A92DA66799D96E03090EE710718B0C086F52F5AC9A2BE6E546C603A5
5505BCDBF3DF54F586047F2722005EB9E8186BC717567654AD6D0CC535C

RIPEMD: 6A0E1009DBEE170D2E93914ED575A67178598E5D

Το link για το drive είναι αυτό:
https://drive.google.com/file/d/1xRV4NYby0ZXKgHJcbPjNPYcZYdIhqmmU/view?usp=sharing

3: Αξιολόγηση συνεργασίας:

Ο φοιτητής 5357 υλοποίησε τα ερωτήματα 1,2,3,5 ενώ ο 5212 ήταν υπεύθυνος για το 4 καθώς και τα δύο bonus που πραγματοποίηθηκαν. Στο τέλος της μέρας στέλναμε το progress που είχαμε για να έχει ο άλλος τον πιο πρόσφατο κώδικα καθώς και μεταφέραμε ολόκληρο το project μέσω usb (αρχικά) και μετά μέσω drive. Η συνεργασία ήταν καλή, ο φόρτος εργασίας βγήκε στο τέλος περίπου ο ίδιος.

4: αναφορές:

https://www.youtube.com/watch?v=m8evpNWSzIA

https://www.youtube.com/watch?v=ByzSgBefNwU