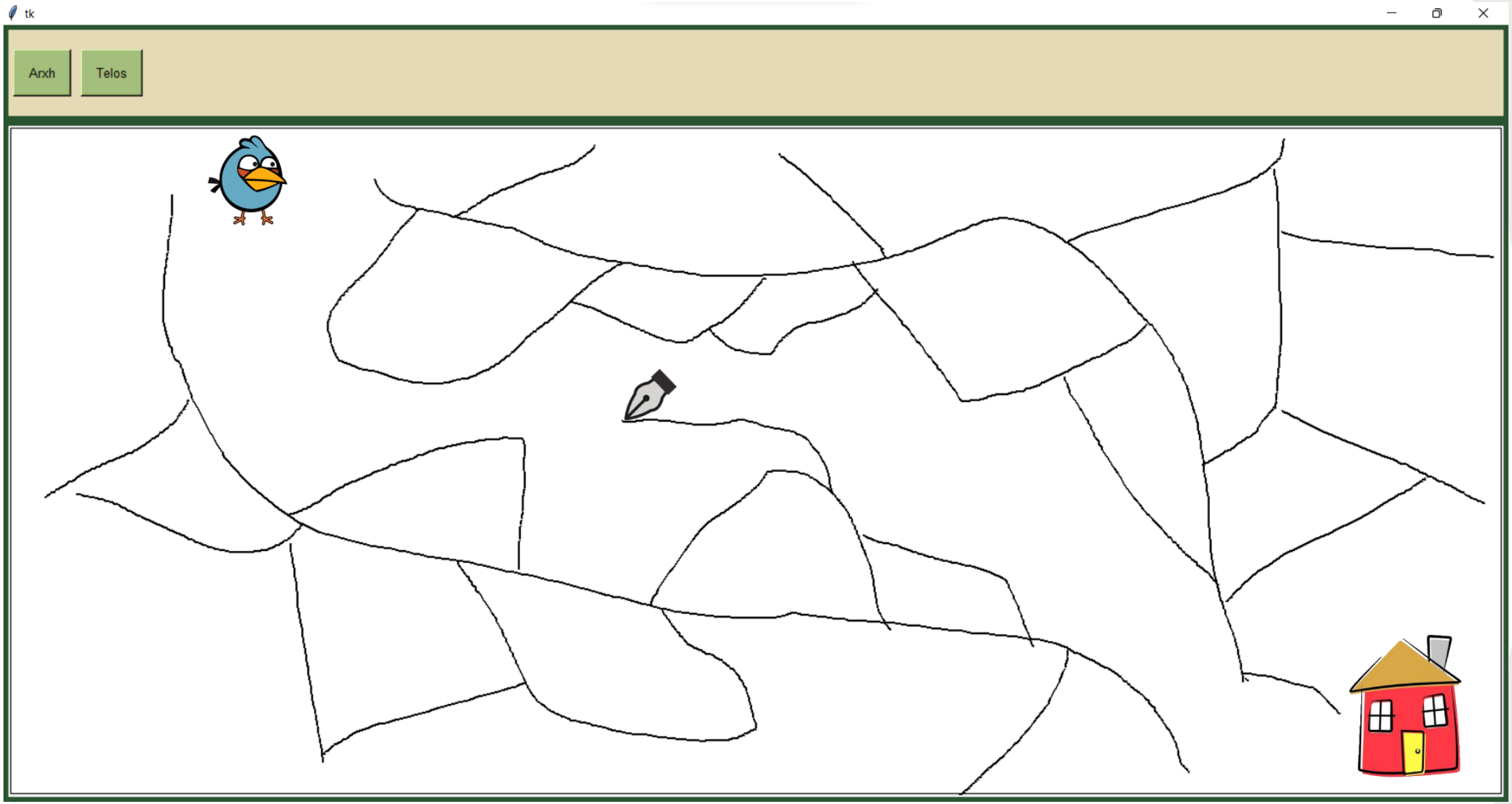
ΤΑ ΠΡΏΤΑ ΒΉΜΑΤΑ

Στην αρχή αποφασίσαμε να διερευνήσουμε το θέμα ξεχωριστά ώστε να δημιουργηθούν διάφορες ιδέες από τον καθένα. Το κάθε μέλος ανέλαβε κάποιο συγκεκριμένο σκέλος του προγράμματος για την πιο αποτελεσματική αντιμετώπιση του θέματος. Αναλυτικότερα, οι υποχρεώσεις της εργασίας χωρίστηκαν ως εξής:

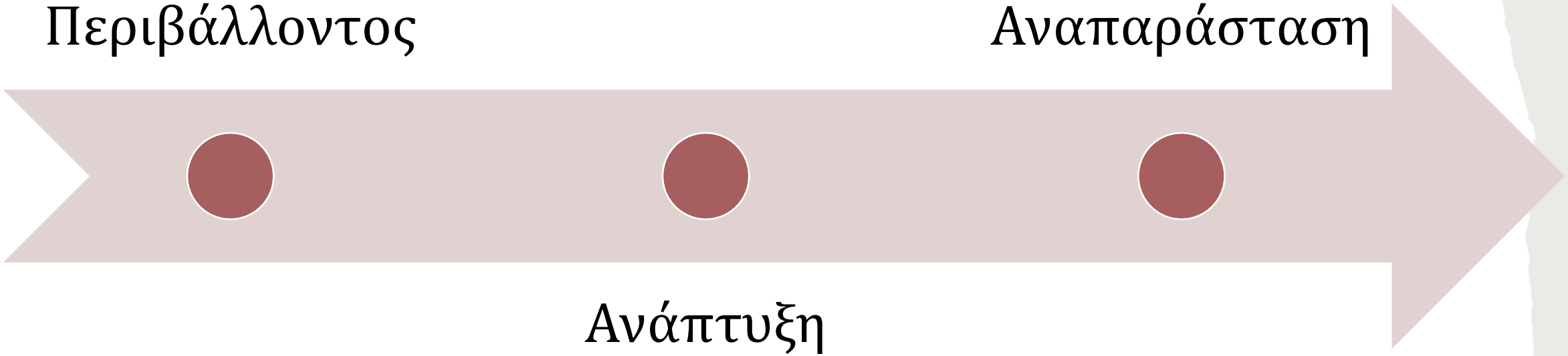
1. Ο Μάρκος Κόρδας ασχολήθηκε με τα γραφικά της tkinter
2. Ο Φώτης Γκόνος ασχολήθηκε με τον αλγόριθμο της πλοήγησης
3. Ο Βασίλης Βλάχος ασχολήθηκε με τον αλγόριθμο της δημιουργίας του λαβυρίνθου

Έπειτα από ένα συλλογικό brainstorming και μια καλή κατανόηση του τι θέλουμε να κάνουμε πλέον, καταλήξαμε στην ιδέα του Μάρκου κατά την οποία η πίστα στην οποία θα κινείται το όχημα θα ζωγραφίζεται από τον ίδιο τον χρήστη μέσω ενός καμβά. Δυστυχώς, έπειτα από περεταίρω ανάλυση καταλήξαμε στο συμπέρασμα ότι μια τέτοια αντιμετώπιση δεν είναι εφικτή. Οπότε, αναγκαστήκαμε να αλλάξουμε τελείως κατεύθυνση και να ξεκινήσουμε από την αρχή. Η νέα μας προσέγγιση βασίστηκε σε πιο δημοφιλή τρόπο επίλυσης τέτοιου είδους προβλημάτων αυτόματης πλοήγησης, την χρήση τετραγωνισμένου λαβύρινθου.

Εδώ φαίνεται αυτή η προσπάθεια σε αρκετά πρόωρο στάδιο ανάπτυξης



ΧΡΟΝΟΔΙΆΓΡΑΜΜΑ ΥΛΟΠΟΊΗΣΗΣ ΤΗΣ ΕΡΓΑΣΊΑΣ



ΔΗΜΙΟΥΡΓΊΑ ΠΕΡΙΒΆΛΛΟΝΤΟΣ

Ύστερα από αρκετή έρευνα και διάφορους αλγόριθμους που βρήκα ήμουν ανάμεσα στον Binary Tree και τον Randomized Prim's Algorithm και αποφάσισα πως καταλληλότερος για το project ήταν ο δεύτερος. Ειδικότερα, ο τρόπος με τον οποίο λειτουργεί ο randomized prim's algorithm είναι ο εξής:

- Ξεκινάμε με ένα πλέγμα γεμάτο τοίχους
- Διαλέγουμε ένα κελί και το σημειώνουμε το ως μέρος του λαβύρινθου. Προσθέτουμε τα τοιχώματα του κελιού σε μια καινούργια λίστα, την λίστα των τοίχων.
- Όσο υπάρχουν τοίχοι στη λίστα:

    1. Επιλέγουμε έναν τυχαίο τοίχο από τη λίστα. Εάν μόνο ένα από τα δύο κελιά που χωρίζει ο τοίχος έχει επισκεφθεί, τότε:

        α) Κάνουμε τον τοίχο ένα πέρασμα και σημειώνουμε το μη επισκέψιμο κελί ως μέρος του λαβύρινθου

        β) Προσθέτουμε τους γειτονικούς τοίχους του κελιού στη λίστα τοίχων.

    2.Αφαιρούμε τον τοίχο από τη λίστα

ΟΠΤΙΚΉ ΑΝΑΠΑΡΆΣΤΑΣΗ ΤΟΥ ΛΑΒΎΡΙΝΘΟΥ

```
IDLE Shell 3.10.6
File  Edit  Shell  Debug  Options  Window  Help
Python 3.10.6 (tags/v3.10.6:9c7b4bd, Aug  1 2022, 21:53:49) [MSC v.1932 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\lenov\Desktop\askhseis\Project\Algorithmos gia labirintho.py
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
>>>
= RESTART: C:\Users\lenov\Desktop\askhseis\Project\Algorithmos gia labirintho.py
['w', 'e', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w']
['w', 'c', 'w', 'c', 'w', 'c', 'c', 'w', 'w', 'c', 'w', 'c', 'w', 'c', 'w', 'c', 'c', 'c', 'c', 'w']
['w', 'c', 'c', 'c', 'w', 'c', 'w', 'w', 'c', 'c', 'w', 'c', 'c', 'c', 'c', 'c', 'w', 'w', 'w', 'w']
['w', 'w', 'w', 'c', 'c', 'c', 'w', 'w', 'c', 'w', 'w', 'c', 'w', 'w', 'w', 'w', 'w', 'w', 'c', 'w']
['w', 'c', 'w', 'w', 'w', 'c', 'c', 'w', 'c', 'c', 'w', 'c', 'w', 'w', 'c', 'w', 'c', 'c', 'c', 'w']
['w', 'c', 'w', 'w', 'w', 'w', 'c', 'w', 'c', 'w', 'w', 'c', 'c', 'c', 'c', 'c', 'c', 'w', 'c', 'w']
['w', 'c', 'w', 'c', 'w', 'w', 'c', 'w', 'c', 'w', 'w', 'c', 'w', 'w', 'w', 'c', 'w', 'w', 'w', 'w']
['w', 'c', 'c', 'c', 'c', 'c', 'c', 'c', 'c', 'c', 'c', 'c', 'w', 'w', 'w', 'w', 'w', 'c', 'w', 'w']
['w', 'w', 'w', 'c', 'w', 'c', 'w', 'w', 'w', 'c', 'w', 'c', 'c', 'c', 'c', 'w', 'c', 'c', 'c', 'w']
['w', 'c', 'c', 'c', 'w', 'c', 'c', 'c', 'w', 'c', 'w', 'w', 'c', 'w', 'w', 'w', 'c', 'w', 'w', 'w']
['w', 'w', 'w', 'c', 'w', 'w', 'c', 'w', 'w', 'w', 'w', 'c', 'c', 'c', 'c', 'c', 'c', 'c', 'c', 'w']
['w', 'c', 'c', 'c', 'c', 'w', 'c', 'c', 'c', 'c', 'w', 'w', 'c', 'w', 'c', 'w', 'c', 'w', 'c', 'w']
['w', 'c', 'w', 'c', 'w', 'w', 'w', 'c', 'w', 'w', 'w', 'w', 'w', 'w', 'c', 'w', 'w', 'w', 'w', 'w']
['w', 'c', 'w', 'c', 'c', 'c', 'w', 'c', 'c', 'w', 'w', 'c', 'c', 'c', 'c', 'c', 'c', 'c', 'c', 'w']
['w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'w', 'x', 'w']
>>>
Ln: 37  Col: 0
```

ΑΝΆΠΤΥΞΗ ΑΛΓΟΡΊΘΜΟΥ ΕΎΡΕΣΗΣ ΔΙΑΔΡΟΜΉΣ

Για την δημιουργία αυτού του μέρους του προγράμματος αξιοποιήθηκε ο Breadth First Search αλγόριθμος. Ο τρόπος που λειτουργεί είναι αρκετά απλός και εύκολα εφαρμόστηκε στο περιβάλλον με μορφή πινάκων που είχαν δημιουργηθεί.

Αρχικά, δημιουργούμε έναν αντίστοιχο πίνακα ίδιων διαστάσεων και αναθέτουμε σε κάθε θέση την τιμή 0 και την τιμή 1 στο σημείο εισόδου. Πηγαίνοντας σε αυτήν την θέση, ελέγχουμε για δύο συνθήκες:

1. Αν τα διπλανά κελιά έχουν ήδη ανατεθεί με κάποιον αριθμό
2. Αν τα διπλανά κελιά είναι ελεύθερα ή αποτελούν τοίχο

Αφότου επαναληφθεί αυτή η διαδικασία έως ότου να φτάσουμε στην προκαθορισμένη έξοδο, χρειάζεται να βρεθεί η διαδρομή που εκτέλεσε τα λιγότερα ‘βήματα’. Για αυτήν την διαδικασία, πηγαίνουμε στις συντεταγμένες του κελιού εξόδου, βρίσκουμε γειτονικό κελί με τιμή κατά 1 μικρότερη του κελιού που ήμαστε και προσθέτουμε τις συντεταγμένες σε μια λίστα, επαναλαμβάνοντας τη διαδικασία αυτή μέχρι να φτάσουμε στην αρχική θέση.

ΟΠΤΙΚΉ ΑΝΑΠΑΡΆΣΤΑΣΗ ΤΟΥ ΑΛΓΟΡΊΘΜΟΥ ΕΎΡΕΣΗΣ ΔΙΑΔΡΟΜΉΣ

```
[0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```

ΑΡΧΙΚΟΣ ΠΙΝΑΚΑΣ

```
[0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 2, 0, 0, 0, 16, 0, 0, 0, 0]
[0, 3, 4, 0, 14, 15, 0, 17, 0, 0]
[0, 4, 0, 0, 13, 0, 0, 16, 0, 0]
[0, 5, 0, 0, 12, 13, 14, 15, 16, 0]
[0, 6, 7, 0, 11, 0, 0, 0, 0, 0]
[0, 0, 8, 9, 10, 11, 12, 13, 14, 0]
[0, 10, 9, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 10, 11, 12, 13, 14, 15, 16, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 17, 0]
```

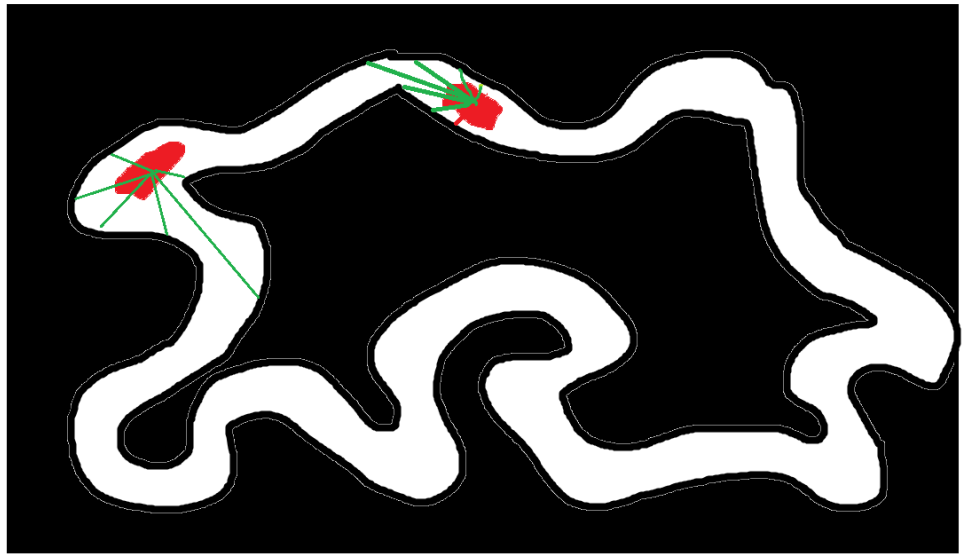
ΠΙΝΑΚΑΣ ΜΕΤΑ ΤΗΝ ΑΝΑΘΕΣΗ ΤΙΜΩΝ

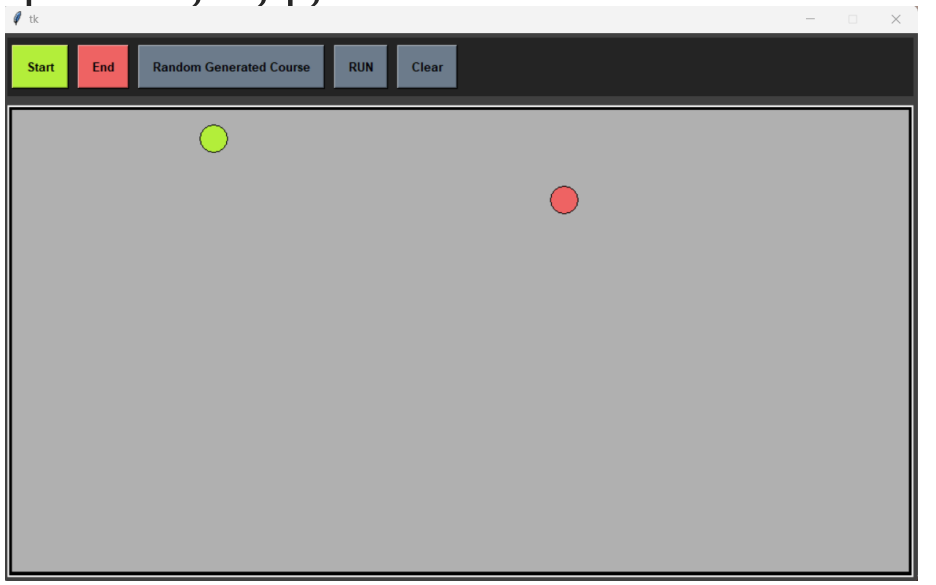
ΣΥΝΤΕΤΑΓΜΕΝΕΣ ΤΕΛΙΚΗΣ ΔΙΑΔΡΟΜΗΣ

```
[(9, 8), (8, 8), (8, 7), (8, 6), (8, 5), (8, 4), (8, 3), (8, 2), (7, 2), (6, 2), (
5, 2), (5, 1), (4, 1), (3, 1), (2, 1), (1, 1), (0, 1)]
>>>
```

ΑΡΧΙΚΌ ΓΡΑΦΙΚΌ ΠΕΡΙΒΆΛΛΟΝ TKINTER

- Στην αρχική ιδέα μας το πρόγραμμα θα ήταν ως εξής:

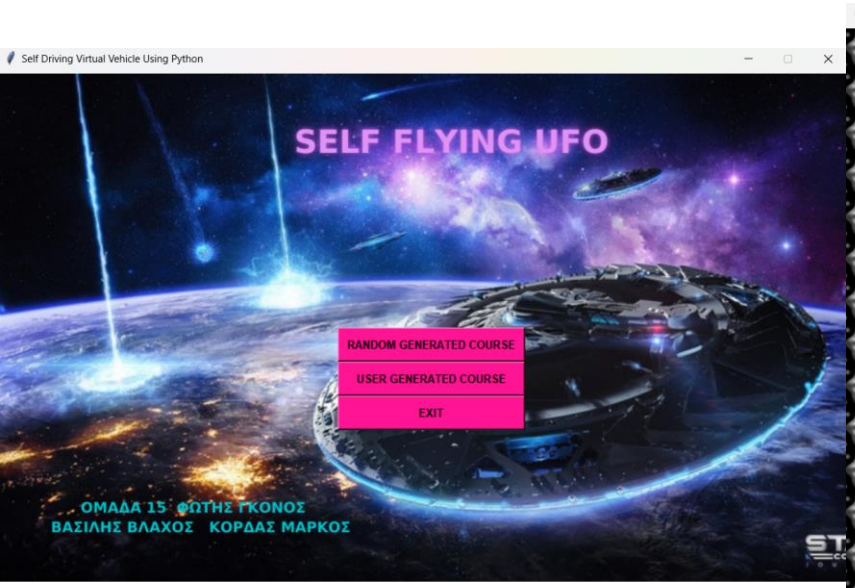


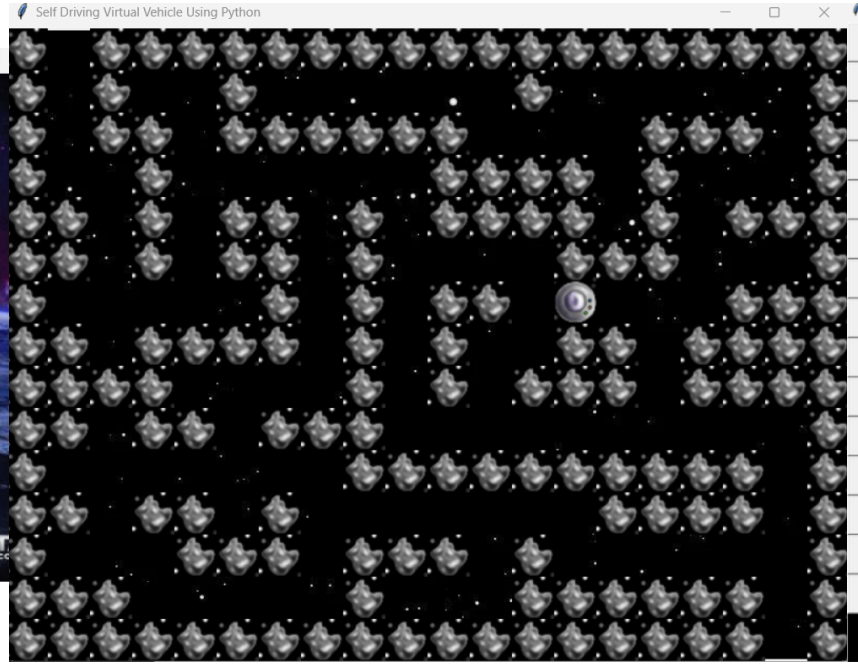


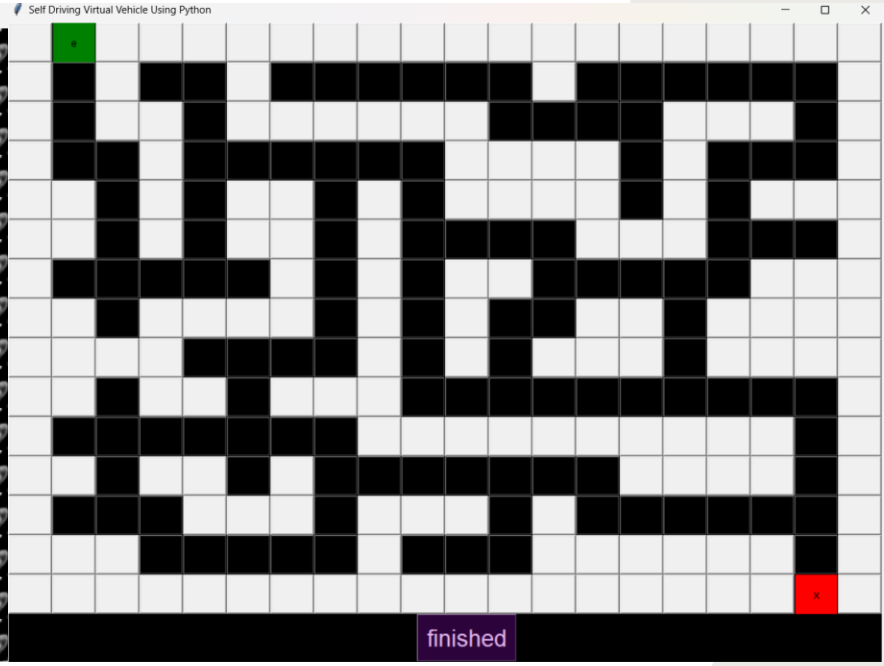
- Όπου το όχημα (κόκκινο σχήμα) θα έχει 'αισθητήρες' (οι πράσινες γραμμές) και θα παίρνει νέα δείγματα για αυτές σε κάθε του step και θα κατευθύνεται προς την μακρύτερη γραμμή. Έτσι θα μπορούσε να κινείται μέσα σε πίστα που θα σχεδίαζε ο χρήστης
- Επίσης με κουμπιά θα τοποθετούταν αντίστοιχα τυχαία αρχή, τυχαίο τέλος (χωρίς να συμπίπτουν), τυχαία διαδρομή και θα έτρεχε το πρόγραμμα, καθώς και θα σβηνόταν η διαδρομή για τη δημιουργία νέας.

ΤΕΛΙΚΉ ΜΟΡΦΉ

- Όταν όμως συναντήσαμε αδιέξοδο (όπως αναφέρθηκε) καταλήξαμε εδώ:







- 1η εικόνα:Αρχική οθόνη
- 2η εικόνα:Τελική μορφή πίστας (είτε random είτε user generated)
- 3η εικόνα:User generated λειτουργία

ΕΙΣΑΓΩΓΗ ΗΧΩΝ

- Με τη χρήση της βιβλιοθήκης winsound η οποία είναι προ εγκατεστημένη στην python, μπορούμε να χρησιμοποιήσουμε αρχεία ήχου τύπου wav στο πρόγραμμά μας.
- Έτσι με κατάλληλη τοποθέτηση εντολών της winsound σε συναρτήσεις του κώδικα επιτυγχάνεται η διαδραστική μορφή του προγράμματος με ήχους στα buttons, στα steps του οχήματος αλλά και στην ολοκλήρωση της διαδρομής με τον χαρακτηριστικό ήχο της προσγείωσης UFO.

ΤΕΛΙΚΆ ΣΥΜΠΕΡΆΣΜΑΤΑ

Γενικά, καθ' όλη τη διάρκεια του project υπήρχε πολύ καλή συνεργασία και όλοι ήταν συνεπείς στις υποχρεώσεις τους παρότι τον μεγάλο φόρτο εργασίας που υπήρχε καθώς τα μέλη της ομάδας ήταν 2 λιγότερα από το προβλεπόμενο, υπήρχε απόλυτη επικοινωνία και όλα λειτούργησαν ομαλά. Θεωρούμε όλοι μας πως μέσω του project μάθαμε πολλά πράγματα για την python, την tkinter και τον προγραμματισμό γενικότερα μέσω των διαφόρων εργασιών που είχαμε για την ολοκλήρωση της εργασίας αυτής. Τα εμπόδια που συναντήσαμε ο καθένας ξεχωριστά στο έργο του, παρόλο το μεγάλο όγκο τους, σίγουρα συνέβαλαν σε μια βαθύτερη κατανόηση του χώρου και του αντικειμένου ενός προγραμματιστή αλλά ταυτόχρονα μας δίδαξαν συνεργατικό πνεύμα, ενίσχυσαν τις ικανότητές μας για επίλυση προβλημάτων και αποτέλεσαν εξαιρετική ευκαιρία για αναζήτηση και προβληματισμό πάνω στο συγκεκριμένο αντικείμενο.