Πανεπιστήμιο Πειραιώς Τμήμα Πληροφορικής Ακαδημαϊκό Έτος 2022-2023

Απαλλακτική Εργασία για το Μάθημα: «ΕΥΦΥΕΙΣ ΠΡΑΚΤΟΡΕΣ» 8ου Εξαμήνου

Ομάδα:
Καλλίας-Βερβεγιώτης Αντώνιος Π19056,
Βαλή-Σαράφογλου Ευρυδίκη Π19023

Μας ζητήθηκε να υλοποιήσουμε μία από τις παρακάτω εργασίες:

- Εργασίες στην πόλη
- Συγκέντρωση αγαθών
- Ανάπτυξη generic planner
- Χρήση στοιχείων γενετικών αλγορίθμων για την εξέλιξη πληθυσμού στο πρόβλημα «Συγκέντρωση Αγαθών».
- ML-Agents (Machine Learning Agents) σύμφωνα με όσα είπαμε στο μάθημα
- Affective Agents σύμφωνα με όσα είπαμε στο μάθημα

Επιλέξαμε την εργασία «Εργασίες στην πόλη».
https://gunet2.cs.unipi.gr/modules/document/file.php/TMD113/%ce%95%cf%81%ce%b3%ce%b1%cf%83%ce%af%ce%b5%cf%82/Apallaktiki1_2017.html

Για την υλοποίηση της εργασίας επιλέξαμε την γλώσσα προγραμματισμού **PYTHON.**
Για την υλοποίηση της διεπαφή χρήστη χρησιμοποιήσαμε την βιβλιοθήκη **tkinter.**

1. Περιγραφή του προβλήματος

Μας ζητήθηκε να υλοποιήσουμε μία ομάδα ευφυών εικονικών πρακτόρων που κινούνται σε μία εικονική πόλη προσπαθώντας να βρεθούν στα κατάλληλα σημεία προκειμένου να εκτελέσουν συγκεκριμένα πλάνα ενεργειών.

Ο εικονικός κόσμος είναι επίπεδος και αποτελείται από wx htετράγωνες θέσεις.

Ο χώρος μέσα στον οποίο βρίσκονται οι πράκτορες έχει τη μορφή εξωτερικών χώρων πόλης. Συγκεκριμένες περιοχές του θεωρείται ότι ανήκουν σε συγκεκριμένα κτίρια.

Στην πόλη υπάρχουν “σημαντικά” κτίρια, σπίτια πρακτόρων και άλλα “ασήμαντα” κτίρια.

Σε κάθε χρονική στιγμή, σε κάθε θέση μπορεί να υπάρχει ένα τμήμα τοίχου συγκεκριμένου κτιρίου, ή τίποτα.

Οι πράκτορες έχουν τη δυνατότητα μετακίνησης μόνο γύρω από τα κτίρια και όχι στο εσωτερικό τους.

Το πλάνο του κάθε πράκτορα αποτελείται από ενέργειες μετακίνησης σε συγκεκριμένα κτίρια με συγκεκριμένη σειρά (π.χ. ταχυδρομείο, αγορά, τράπεζα).

Η ομάδααποτελείται από έναν έως ηπράκτορες.

Οι πράκτορες έχουν δυνατότητες αντίληψης του περιβάλλοντα χώρου τους: πιο συγκεκριμένα, μπορούν να αντιληφθούν αν οι γειτονικές τους θέσεις είναι κενές ή περιέχουν κάποιον άλλο πράκτορα ήτμήμα τοίχου συγκεκριμένου κτιρίου.

Όλοι οι πράκτορες μπορούν να μετακινηθούν σε γειτονική θέση και να ανταλλάξουν γνώση με πράκτορες σε γειτονικές θέσεις

Αρχικά οι πράκτορες δεν γνωρίζουν τη δομή της πόλης, δηλαδή σε ποιές θέσεις υπάρχουν τμήματα τοίχων.

2. Περιγραφή της θεωρητικής βάσης της εφαρμογής

Αρχικά η υλοποίηση έγινε στη γλώσσα προγραμματισμού python και χρησιμοποιήσαμε διάφορες δομές δεδομένων όπως:

- Λίστες
    - Για την αναπαράσταση του χώρου
    - Την γνώση κάθε πράκτορα
    - Το πλάνο κάθε πράκτορα
- Κλάσεις
    - Αναπαράσταση της πόλης
    - Αναπαράσταση των πρακτόρων
    - Για την υλοποίηση του UI μέσο tkinter
- Σωρούς
    - Για την υλοποίηση του A* αλγορίθμου.

Από αλγόριθμους χρησιμοποιήσαμε τον **αλγόριθμο A*** για την εύρεση της πιο σύντομης διαδρομής για τον στόχο κάθε πράκτορα την στιγμή που αποκτούσαν γνώση της τοποθεσίας του.

Για την εφαρμογή του αλγορίθμου στο συγκεκριμένο πρόβλημα κάναμε τις εξής προσαρμογές:

1. Ο αλγόριθμος δέχεται ως είσοδο:
    a. Την τωρινή τοποθεσία του πράκτορα.
    b. Την τοποθεσία του στόχου,.
    c. Την τωρινή γνώση του πράκτορα. (Ώστε το μονοπάτι που θα παραχθεί να αποτελείτε μόνο από γνωστές για τον πράκτορα εκείνο τοποθεσίες)
    d. Τα εμπόδια (δλδ κτήρια) που βρίσκονται στην τωρινή γνώση του πράκτορα επειδή οι πράκτορες δεν μπορούν να μετακινηθούν μέσα από κτήρια.
2. Από τη στιγμή που κάθε πράκτορας μπορεί να κουνηθεί μπροστά, πίσω, αριστερά και δεξιά(όχι διαγώνια) οι γειτονικές θέσεις κάθε node είναι επίσης μπροστά, πίσω, αριστερά ή δεξιά.
3. Πέρα από τον έλεγχο αν μια γειτονική θέση βρίσκεται στο κλειστό σύνολο ελέγχουμε αν κάποια γειτονική θέση είναι εμπόδιο.

3. Περιγραφή σημαντικών σχεδιαστικών αποφάσεων και στοιχείων υλοποίησης

Αρχικά για τον σχεδιασμό του εικονικού κόσμου χρησιμοποιήσαμε αρχεία κειμένου όπου τα σημαντικά κτήρια είναι σημειωμένα με ένα από τα παρακάτω γράμματα και αντιστοιχούν σε συγκεκριμένα χρώματα.

- W- κίτρινο
- T - μπλε
- A - κόκκινο
- G – πράσινο

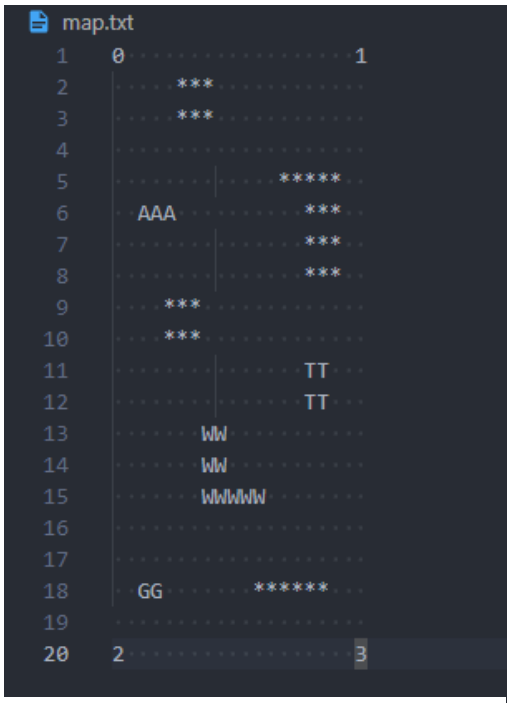
Για την αναπαράσταση των ασήμαντων κτιρίων χρησιμοποιούμε το σύμβολο * και το εμφανίζουμε ως γκρι.

Για την αναπαράσταση των σπιτιών των πρακτόρων χρησιμοποιούμε τους αριθμούς «0-9».

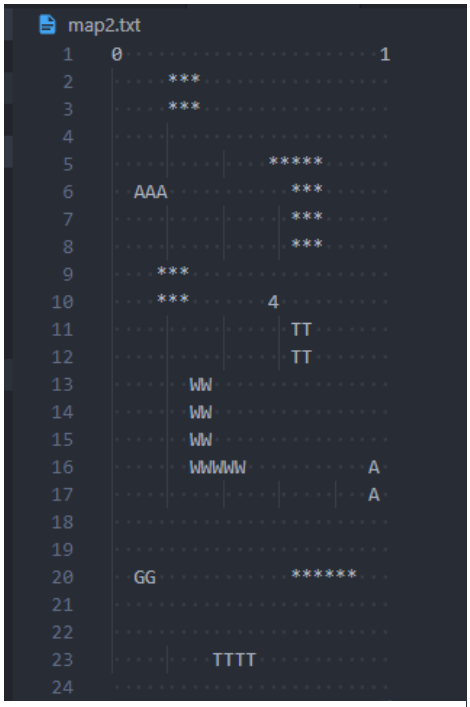
(κάθε σπίτι μπορεί να περιέχει πολλούς πράκτορες με το ίδιο πλάνο. Δηλαδή αν έχουμε 100 πράκτορες και 4 σπίτια θα πρέπει να ορίσουμε στο αρχείο πλάνου 4 πλάνα. Έτσι κάθε σπίτι θα έχει 25 πράκτορες με το ίδιο πλάνο.)

Για την αναπαράσταση του κενού χρησιμοποιούμε το κενό (space).

Κάθε χάρτης πρέπει να έχει το ίδιο μήκος και πλάτος



Εικόνα 2 Χάρτης 20X20



Εικόνα 1 Χάρτης 25X25

Για τον ορισμό των πλάνων των πρακτόρων χρησιμοποιούμε json αρχεία με την παρακάτω μορφή:

Με Agent αναγράφεται ο αριθμός του σπιτιού και στο Plan το πλάνο των πρακτόρων του σπιτιού αυτού.

Το πρόγραμμα αποτελείτε από 4 βασικές κλάσεις και τον αλγόριθμο α*.

1. Την κλάση MainMenu που είναι υπεύθυνη για την εμφάνιση του κεντρικού μενού, την είσοδο του αρχείου χάρτη και πλάνων.

2. Την κλάση City που αναπαριστά τον εικονικό κόσμο και περιέχει τις παρακάτω μεθόδους:

    - createAgents(agentN, planFile)
      Όπου δέχεται ως είσοδο των αριθμό των πρακτόρων και το αρχείο πλάνων και δημιουργεί όλα τα αντικείμενα πράκτορες αναθέτοντας τους ένα σπίτι(όπου θα είναι η αρχική τους θέση) και ένα πλάνο.

    - getAgentHouses()
      Όπου βρίσκει την τοποθεσία των σπιτιών των πρακτόρων από το αρχείο χάρτη.

    - getBuildings()
      Όπου βρίσκει την τοποθεσία κάθε κτήριού από το αρχείο χάρτη.

3. Την κλάση Agent που αναπαριστά τους πράκτορες και περιέχει τις παρακάτω μεθόδους:

    - getPlans(planFile:)
      Η συνάρτηση αυτή δέχεται το αρχείο πλάνων και επιστρέφει το πλάνο του συγκεκριμένου πράκτορα.

    - move(newLoc)
      Η συνάρτηση αυτή δέχεται ως είσοδο μια τοποθεσία και μετακινεί τον πράκτορα σε αυτή. Έπειτα προσθέτει την τοποθεσία στην γνώση του πράκτορα.

    - updateKnowledge(blocks)
      Δέχεται ως είσοδο μια λίστα από τοποθεσίες και τις προσθέτει στην γνώση του πράκτορα.

    - checkKnowledgeForGoal()
      Ελέγχει την γνώση του πράκτορα για τον τωρινό στόχο στο πλάνο του. Αν υπάρχει τότε καλεί την συνάρτηση goToLocation(location)

    - goToLocation(location)
      Δέχεται ως όρισμα μια τοποθεσία και έπειτα χρησιμοποιεί τον αλγόριθμο A* για να βρει την βέλτιστη διαδρομή χρησιμοποιώντας μόνο τις γνωστές περιοχές του πράκτορα.

    - lookAround(nearbyBlocks, nearbyAgents)
      Η συνάρτηση δέχεται ως όρισμα τις γειτονικές περιοχές του πράκτορα και τους γειτονικούς πράκτορες (αν υπάρχουν). Αν σε γειτονική θέση υπάρχει ο τωρινός στόχος στο πλάνο του τότε τον αφαιρεί από το πλάνο του. Αν ολοκληρώσει το πλάνο του τότε καλεί την συνάρτηση goHome().

    - tradeInfo(agent)
      Η συνάρτηση δέχεται ως είσοδο έναν άλλο πράκτορα και ανταλλάζει τις γνώσεις τους μεταξύ τους.

    - goHome()
      Καλεί την συνάρτηση goToLocation() με τις συντεταγμένες του σπιτιού του πράκτορα. Έτσι ο πράκτορας πάει σπίτι του χρησιμοποιώντας την πιο βέλτιστη διαδρομή χρησιμοποιώντας τον αλγόριθμο A*.

4. Την κλάση Application που είναι υπεύθυνη για την εκτέλεση της προσομοίωσης και αποτελείτε από τις παρακάτω βασικές μεθόδους:
    - drawPath(path)
      Δέχεται ως είσοδο ένα μονοπάτι (πχ μια διαδρομή που δημιούργησε ο A* για να πάει ένας πράκτορας στον στόχο του) και το ζωγραφίζει με πράσινο στην προσομοίωση.

    - build_interface()
      Δημιουργεί όλο το UI (Canvas, Grid, Labels, Buttons)

    - updateSelected()
      Βρίσκει τον επιλεγμένο πράκτορα και έπειτα εμφανίζει διάφορες πληροφορίες για αυτόν όπως τις κινήσεις του και το πλάνο του.

      Έπειτα εμφανίζει τα μέρη της πόλης για τα οποία ο επιλεγμένος πράκτορας έχει γνώση μαυρίζοντας όλα τα μέρη της πόλης που δεν γνωρίζει.

      Τέλος εμφανίζει το ενεργό του μονοπάτι(αν υπάρχει) χρησιμοποιώντας την μέθοδο drawPath().

    - createControls()
      Δημιουργεί όλα τα controls που έχει ο χρήστης για τον έλεγχο της προσομοίωσης όπως την επιλογή πράκτορα, pause, resume, cancel και την χειροκίνητη συνέχεια της προσομοίωσης.

    - create_city(mapFile,planFile,agentN)
      Παίρνει ως είσοδο το αρχείο χάρτη, πλάνων και τον αριθμό τον πρακτόρων και έπειτα δημιουργεί το αντικείμενο city χρησιμοποιώντας την κλάση City.

    - beginSimulation()
      Ξεκινά την προσομοίωση ξεκινώντας τον βασικό βρόχο ο οποίος καλεί την συνάρτηση moveAgentsFrame() και drawNewFrame() κάθε φορά.

    - stopSimulation()
      Σταματά την προσομοίωση εμφανίζοντας στατιστικά στην κονσόλα και τα αποθηκεύει στο αρχείο stats.txt.
      Τα στατιστικά είναι ο χρόνος εκτέλεσης της προσομοίωσης, τα βήματα κάθε πράκτορα και πόσες ανταλλαγές γνώσεις κάνανε.

    - drawNewFrame()
      Διαβάζει τον χάρτη και έπειτα τον ζωγραφίζει στον καμβά ως ένα grid.
      Τέλος ζωγραφίζει τους πράκτορες στον καμβά βάση την τωρινή τους τοποθεσία.

- moveAgentsFrame()
  Αρχικά για κάθε πράκτορα βρίσκει τις πιθανές τους κινήσεις και έπειτα τις φιλτράρει βάση αν είναι μέσο ορίων, αν βρίσκεται κτήριο εκεί και αν ο πράκτοράς αυτός έχει ήδη επισκεφτεί την τοποθεσία αυτή.

  (Αν ο πράκτορας έχει ενεργό κάποιο μονοπάτι A* αγνοεί τα παρακάτω βήματα.

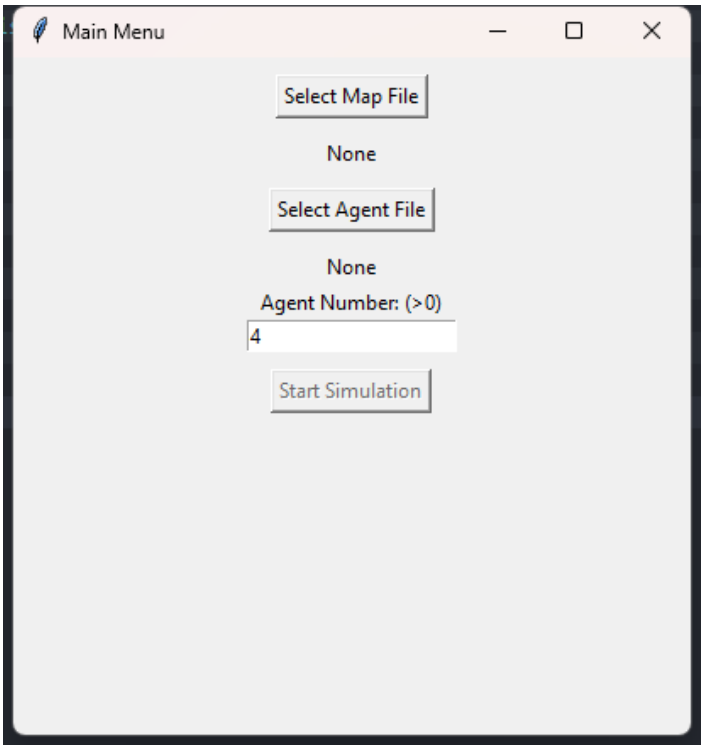
  Διαλέγει τυχαία μία από της γειτονικές περιοχές που δεν έχει επισκεφτεί. (Άμα δεν γίνεται τότε διαλέγει τυχαία μία από της γειτονικές θέσεις)

  Έπειτα μετακινεί τον πράκτορα αυτόν χρησιμοποιώντας και ενημερώνει την γνώση του «κοιτώντας» τι βρίσκεται στις γειτονικές του πλέον θέσεις.

  Τέλος ελέγχει αν κάποιος πράκτορας έχει άδειο πλάνο και βρίσκεται σπίτι του. Αν ναι τότε η προσομοίωση τελείωσε.

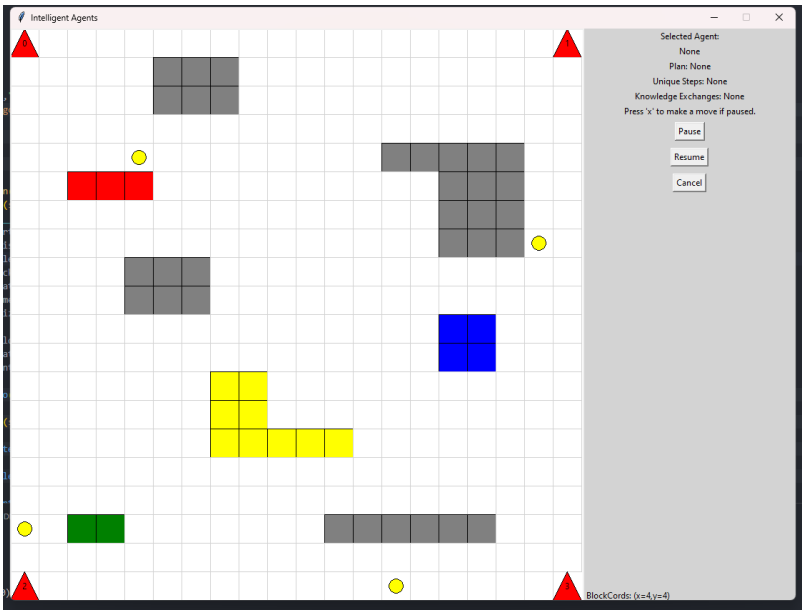
Εκτέλεση Προσομοίωσης

Κατά την εκτέλεση του προγράμματος το πρώτο πράγμα που βλέπουμε είναι το παράθυρο επιλογής χάρτη και πλάνων των πρακτόρων. Επίσης μπορούμε να επιλέξουμε τον αριθμό των πρακτόρων.



Για την εκτέλεση του παραδείγματος θα χρησιμοποιήσουμε τα αρχεία map.txt και plans.json με αριθμό πρακτόρων 4.

Μόλις πατάμε Start Simulation μας ανοίγει το παρακάτω παράθυρο το οποίο εκτελεί την προσομοίωση. Οι πράκτορες ξεκινάνε από τα σπίτια τους και αρχίζουν να αναζητούν το πρώτο κτήριο στο πλάνο τους.

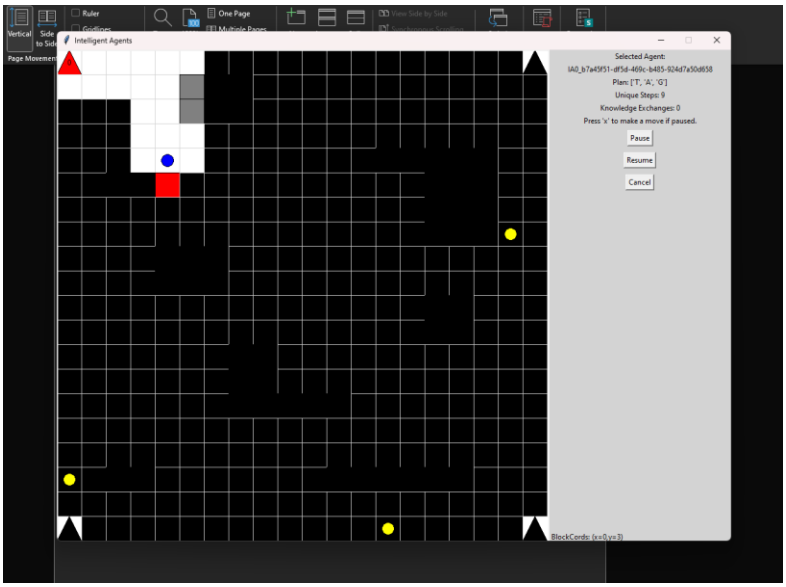


Αριστερά μας δίνονται 3 βασικές επιλογές. Pause / Resume / Cancel.

Επίσης μας δίνεται η δυνατότητα αν η προσομοίωση είναι paused να πατήσουμε X όπου θα προχωρήσει την προσομοίωση ένα βήμα.

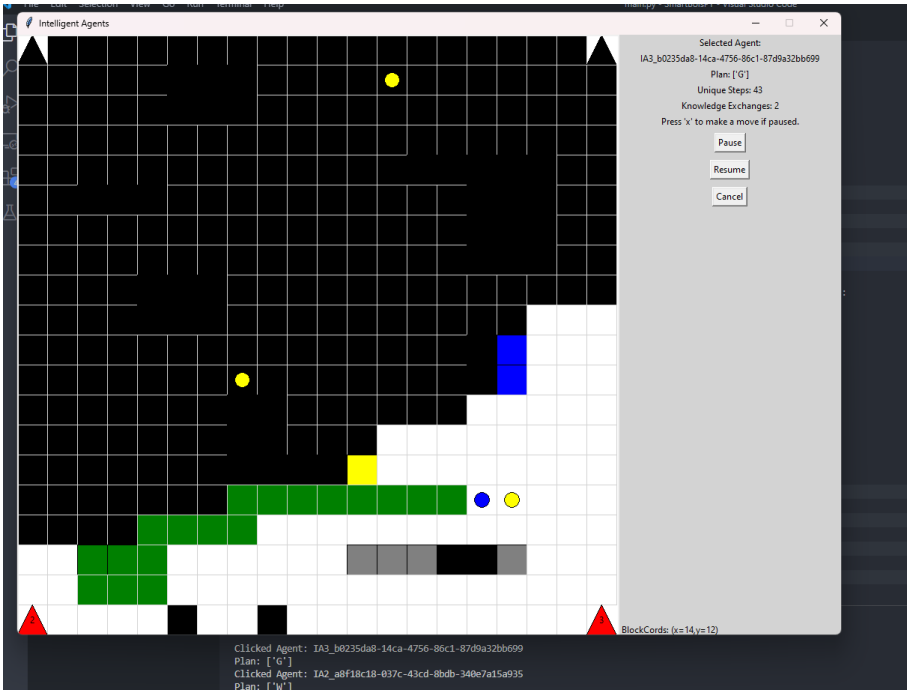
Πατώντας πάνω σε έναν πράκτορα μας εμφανίζει με σκίαση την τωρινή του γνώση της πόλης.

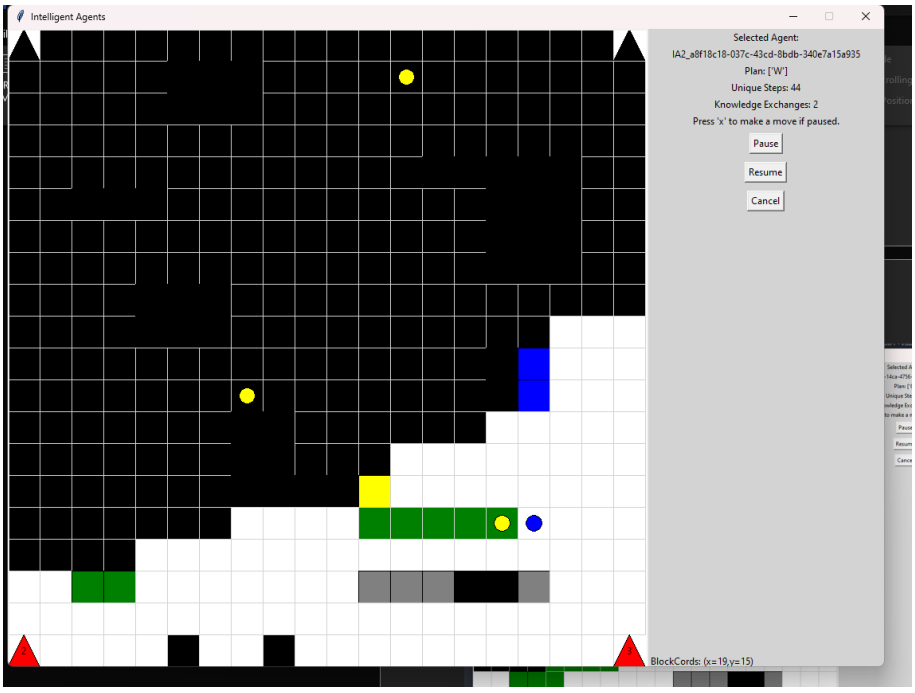
Δεξιά βλέπουμε διάφορες πληροφορίες του επιλεγμένου πράκτορα όπως το όνομα του (Το οποίο αποτελείτε από IA + [τον αριθμό του σπιτιού που ανήκει] + UUID) , το πλάνο του, πόσα βήματα έχει κάνει και πόσες ανταλλαγές γνώσεις έχει κάνει.



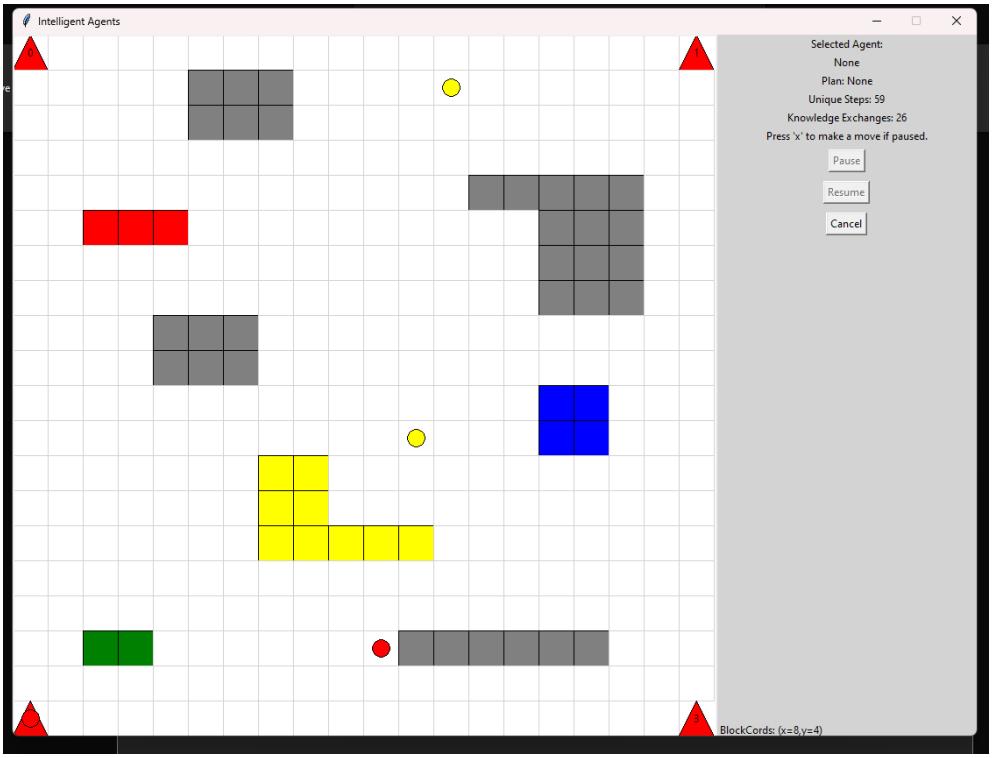
Παρακάτω βλέπουμε ένα παράδειγμα ανταλλαγής γνώσης μεταξύ 2 πρακτόρων. Ο επιλεγμένος πράκτορας(μπλε) μόλις συναντήθηκε με τον διπλανό του πράκτορα και ανταλλάξανε πληροφορίες με αποτέλεσμα να μάθει την τοποθεσία του πρώτου κτηρίου στο πλάνο του. Με χρήση του A* δημιούργησε ένα μονοπάτι (πράσινο) που θα ακολουθήσει για να φτάσει σε αυτό το κτήριο.

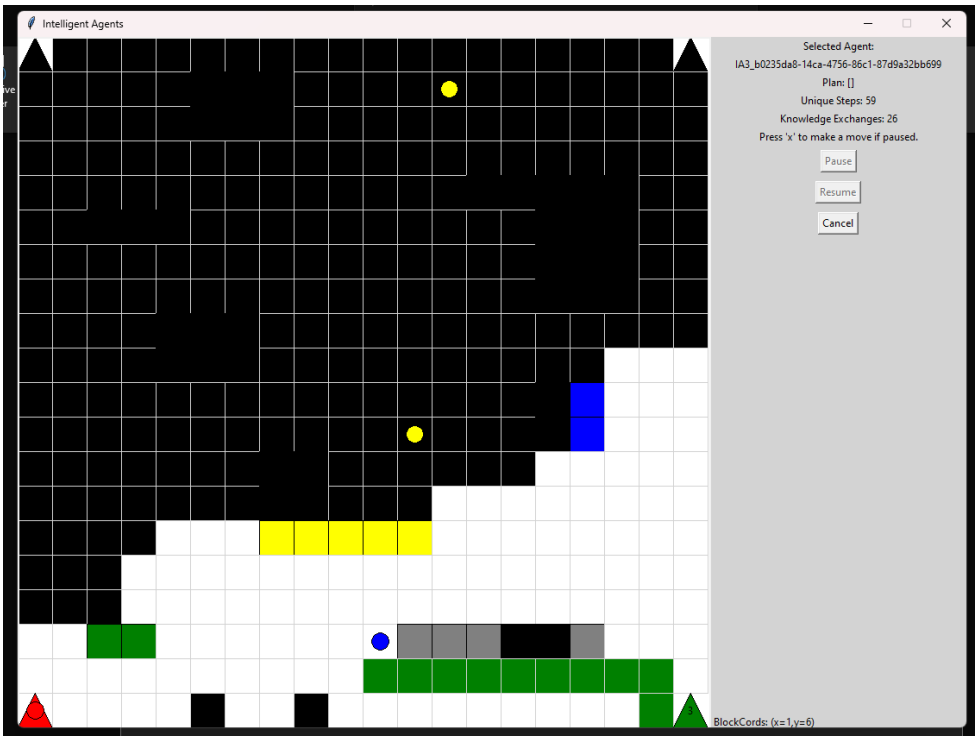
Παρατηρούμε ότι οι 2 πράκτορες που συναντήθηκαν έχουν την ίδια γνώση από την στιγμή που ανταλλάξανε την γνώση τους.



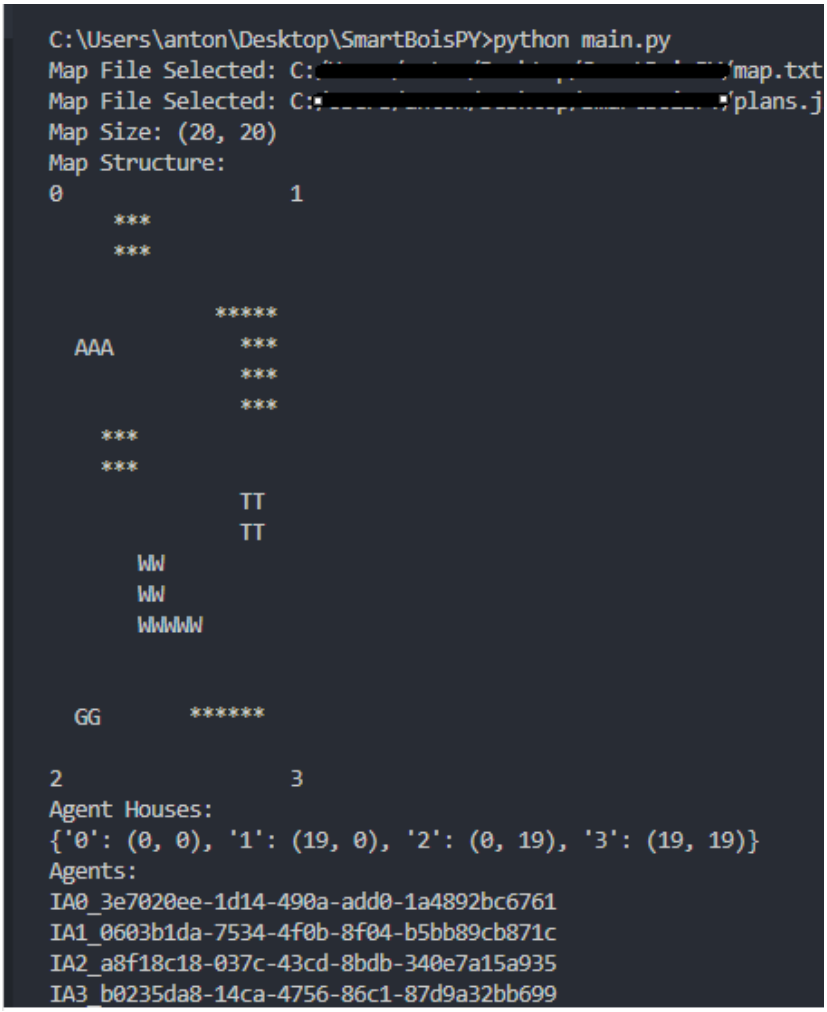


Τέλος όταν ένας πράκτορας ολοκληρώσει το πλάνο του και φτάσει σπίτι του τότε τερματίζει η προσομοίωση. Με κόκκινο χαρακτηρίζονται οι πράκτορες που έχουν τελειώσει το πλάνο τους και γυρνάνε σπίτι.

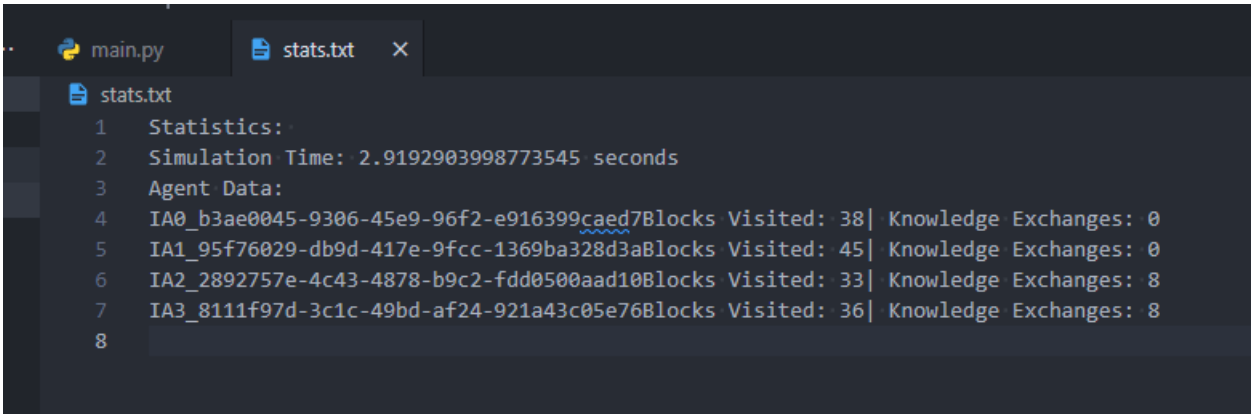




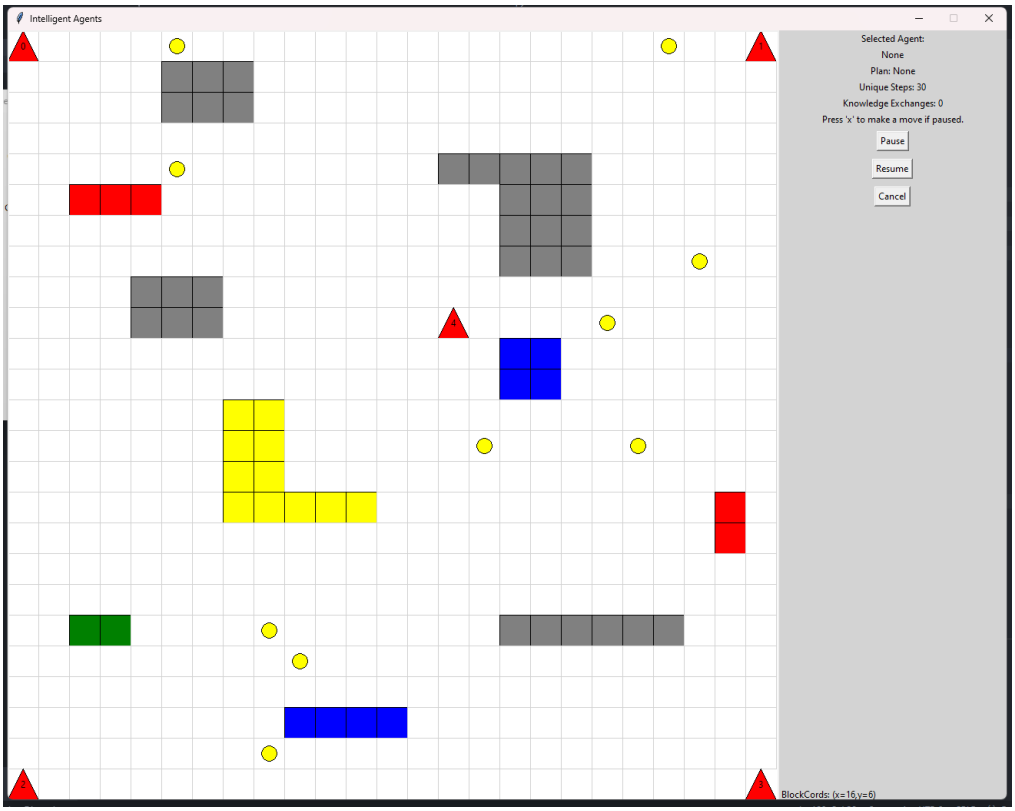
Στην κονσόλα εμφανίζονται διάφορες πληροφορίες όπως τα αρχεία που χρησιμοποιήσαμε, το μέγεθος της πόλης, αναπαράσταση της πόλης, τα σπίτια των πρακτόρων και οι συντεταγμένες τους και τα ονόματα των πρακτόρων.



Τέλος στο αρχείο stats.txt αναγράφονται τα στατιστικά που μας ζητήθηκαν.



Παράδειγμα εκτέλεσεις του χάρτη map2.txt (25X25) με 5 σπίτια πρακτόρων και 10 πράκτορες(δεν υπάρχει όριο στον αριθμό πρακτόρων):



Αναλυτική περιγραφή της διαδικασίας εγκατάστασης

Το μόνο που χρειάζεται για την εκτέλεση του προγράμματος είναι η εγκατάσταση της γλώσσας προγραμματισμού Python https://www.python.org/downloads/ . Όλες οι βιβλιοθήκες που χρησιμοποιήσαμε έρχονται προ εγκαταστημένες.

Αναλυτική περιγραφή της συμβολής του κάθε μέλους της ομάδας

Καλλίας-Βερβεγιώτης Αντώνιος Π19056:

- Κλάση Agents και τις λειτουργίες της
- Κλάση Application και τις λειτουργίες της
- Εμφάνιση γνώσης και ενεργά μονοπάτια
- Σχεδιασμός χάρτη και πλάνων

Βαλή-Σαράφογλου Ευρυδίκη Π19023:

- Αλγόριθμο A*
- Κλάση MainMenu και της λειτουργίες της
- Ένα κομμάτι του UI στην προσομοίωση (κλάση Application).
- Κλάση City και της λειτουργίες της

Αναλυτική περιγραφή ανοικτών θεμάτων, ανεπίλυτων προβλημάτων και πιθανοτήτων εμφάνισης σφαλμάτων κατά την εκτέλεση

Τα μόνα πιθανά σφάλματα που μπορεί να εμφανιστούν προκύπτουν από την εισαγωγή μη εγκύρων αρχείων χάρτη και πλάνων.

Πρέπει τα αρχεία χάρτη να έχουν το ίδιο μήκος και πλάτος.

Ο αριθμός των πλάνων να είναι τουλάχιστον ίδιος με τον αριθμό των σπιτιών πρακτόρων.