



Πρόγραμμα Σπουδών: **Πληροφορικής**

Θεματική Ενότητα: ΠΛΗΠΡΟ

ID Project: 54

Ακαδημαϊκό Έτος: 2022 – 2023

Συμμετέχοντες:

κ. Δημήτριος Ματαφιάς

κ. Λεωνίδας Σαραφίδης

κα. Κρυσταλλία Μακατού

Περιεχόμενα

Περιεχόμενα.....	2
Εισαγωγή	3
Κατανομή Εργασιακού Φόρτου.....	3
Κατασκευή Κλάσης	4
Κατασκευή Γραφικού Περιβάλλοντος	7
Κατασκευή Αλγορίθμου Επιλογής του Εχθρού.....	11
Κατασκευή του Τελικού Προγράμματος.....	11
Διευκρινήσεις.....	11
Βιβλιογραφία	12

Εισαγωγή

Σαν ομάδα κληθήκαμε να γράψουμε κώδικα στη γλώσσα προγραμματισμού python ο οποίος να δημιουργεί γραφικό περιβάλλον, στο οποίο ένας χρήστης να παίζει ηλεκτρονικά το παιχνίδι της ναυμαχίας με αντίπαλο τον υπολογιστή.

Οι απαραίτητες προϋποθέσεις ή καλύτερα οι απαιτήσεις που μας ζητήθηκαν ως προ απαιτούμενα για την υλοποίηση του προγράμματος είναι οι παρακάτω:

1. Οι στόλοι των δύο παιχτών θα αποτελούνται από 4 πλοία (1 πλοίο που καταλαμβάνει 5 θέσεις, 1 πλοίο που καταλαμβάνει 4 θέσεις, 1 πλοίο που καταλαμβάνει 3 θέσεις, και 1 πλοίο 2 θέσεων).
2. Στην πρώτη φάση του παιχνιδιού οι δύο παίκτες (χρήστης/ υπολογιστής) θα πρέπει να τοποθετήσουν τα πλοία τους στο ταμπλό.
 - a. Για την τοποθέτηση των πλοίων του υπολογιστή θα χρησιμοποιηθούν γεννήτριες τυχαίων τιμών, από τις οποίες θα επιλέγεται α) η θέση της αρχής ενός πλοίου, β) ο προσανατολισμός του πλοίου (αν τοποθετείται κατακόρυφα ή οριζόντια). Θα πρέπει να υλοποιηθεί αμυντικός προγραμματισμός ώστε να μην βγαίνουν τα πλοία από τα όρια του ταμπλό.
 - b. Ο χρήστης θα πρέπει να εισάγει τα στοιχεία αυτά από το γραφικό περιβάλλον.
3. Ο χρήστης θα πρέπει να έχει εικόνα και στο δικό του ταμπλό αλλά και στις βολές που έχει πραγματοποιήσει (ταμπλό αντιπάλου). Επίσης, θα πρέπει να φαίνονται οι βολές του αν είναι επιτυχημένες ή άστοχες με χρώμα κόκκινο ή λευκό αντιστοίχως.
4. Ο υπολογιστής θα πραγματοποιεί εν γένει τις βολές του με τυχαίο τρόπο, ωστόσο αν επιτύχει κάποιο πλοίο θα πρέπει «να έχει την ευφυΐα» να στοχεύσει στις επόμενες βολές του κοντινά σημεία μέχρι να βυθίσει το πλοίο που έχει πλήξει.

Λαμβάνοντας υπόψη τα παραπάνω καταλήξαμε στα εξής συμπεράσματα και παρατηρήσεις:

1. Αρχικά χρειαζόμαστε ένα γραφικό περιβάλλον το οποίο και αποφασίσαμε να πραγματοποιήσουμε με τη βιβλιοθήκη Tkinter.
2. Χρειαζόμαστε 4 πλοία με μεγέθη από 5 μέχρι 2 στα οποία δώσαμε και τις εξής ονομασίες:
 - a. Aircraft Carrier
 - b. Battleship
 - c. Cruiser
 - d. Destroyer

αντίστοιχα έπειτα από έρευνα περί μεγεθών πλοίων.

3. Απαιτείται η κατασκευή ενός αλγόριθμου που θα κάνει την επιλογή της επόμενης κίνησης του εχθρού.
4. Και τέλος κρίναμε απαραίτητο να χρησιμοποιήσουμε τις βιβλιοθήκες random και time για την παραγωγή τυχαίων αριθμών.

Κατανομή Εργασιακού Φόρτου

Έπειτα από αρκετή συζήτηση και αφού αναπτύξαμε στο χαρτί ένα πρότυπο σχετικά με το πως σκοπεύουμε να κινηθούμε ως προς την υλοποίηση της εργασίας, καταλήξαμε στο συμπέρασμα πως η εργασία πρέπει να χωριστεί σε τουλάχιστον τέσσερα κομμάτια. Τα κομμάτια αυτά είναι:

- Κατασκευή μίας κλάσης η οποία θα αποθηκεύει όλες τις πληροφορίες για τον κάθε παίκτη Φίλο/Εχθρό.

- Κατασκευή του γραφικού περιβάλλοντος το οποίο θα χρησιμοποιεί ο εκάστοτε παίχτης για να εισάγει τις απαραίτητες πληροφορίες καθώς και να αλληλοεπιδρά με το παιχνίδι.
- Κατασκευή ενός αλγόριθμου που θα αναπαριστά τον ακριβή τρόπο σκέψης με τον οποίο ένας άνθρωπος επιλέγει την επόμενη κίνησή του στο συγκεκριμένο παιχνίδι.
- Και τελικά την χρησιμοποίηση κατάλληλων συναρτήσεων και μεθόδων με σκοπό τη συνένωση των παραπάνω τμημάτων έτσι ώστε να δουλεύουν αρμονικά μαζί, με τον βέλτιστο δυνατό τρόπο.

Κατά τη διάρκεια τις κατασκευής των παραπάνω τμημάτων υπήρχε κατά τακτικά διαστήματα επικοινωνία μεταξύ μας έτσι ώστε τα τμήματα να είναι συμβατά το ένα με το άλλο.

Τα τμήματα τα αναλάβαμε με τη σειρά:

- Με το πρώτο τμήμα της κλάσης ασχολήθηκε ο κ. Σαραφίδης Λεωνίδας
- Με το δεύτερο που αφορά το γραφικό περιβάλλον η κα. Μακατού Κρυσταλλία
- Με το τρίτο τμήμα που αφορά τον αλγόριθμο του εχθρού ο κ. Ματαφιάς Δημήτριος
- Και με το τελευταίο τμήμα της συνένωσης ασχολήθηκε ο κ. Ματαφιάς Δημήτριος

Κατασκευή Κλάσης

Η κλάση που θέλαμε να κατασκευάσουμε θα περιέχει όλες τις πληροφορίες του κάθε παίχτη. Θέλαμε δηλαδή δύο αντικείμενα, ένα για τον εκάστοτε παίχτη, το οποίο ονομάσαμε τελικά και friend, και ένα για τον υπολογιστή, το οποίο ονομάσαμε τελικά enemy.

Αποφασίσαμε πως πριν εργαστούμε με το γραφικό περιβάλλον θα έπρεπε να φτιάξουμε ένα αντίστοιχο παιχνίδι το οποίο να δουλεύει στο IDLE της python δηλαδή «στο χαρτί».

Ξεκινήσαμε αποφασίζοντας πως χρειαζόμαστε δύο καθολικές μεταβλητές κλάσης ROWS και COLUMNS οι οποίες θα περιέχουν πληροφορίες για τις γραμμές και τις στήλες αντίστοιχα, κοινές για κάθε δημιουργούμενο αντικείμενο. Έπειτα, ομοίως με τις προηγούμενες θέλουμε 4 μεταβλητές AIRCRAFT_CARRIER, BATTLESHIP, CRUISER και DESTROYER οι οποίες θα περιέχουν το μήκος κάθε πλοίου.

Έπειτα αποφασίσαμε ότι κάθε αντικείμενο που θα δημιουργείται χρειάζεται τις εξής πληροφορίες:

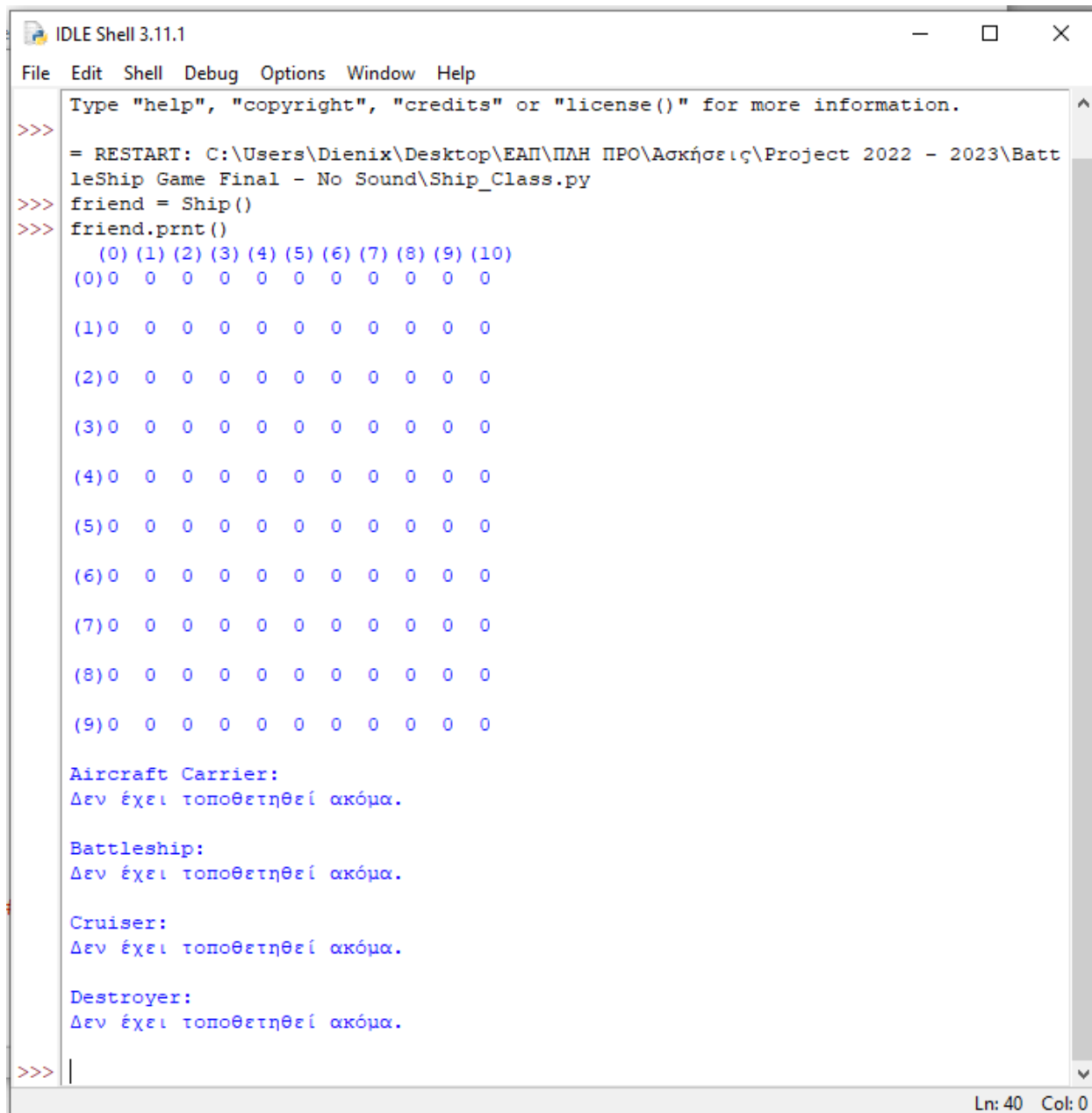
- Έναν πίνακα self.is_occupied ο οποίος είναι μία λίστα η οποία περιέχει μέσα λίστες όσες απαιτεί η μεταβλητή ROWS και η κάθε λίστα αρχικοποιείται με 0 τόσα όσα απαιτεί η μεταβλητή COLUMNS. Ουσιαστικά πρόκειται για έναν πίνακα (10x11) δηλαδή με 10 γραμμές και 11 στήλες.
- Τέσσερις μεταβλητές:
 - self.aircraft_carrier
 - self.battleship
 - self.cruiser
 - self.destroyer
 οι οποίες αποτελούν λεξικά με 2 κλειδιά, “position” και “adjacent” όπου το πρώτο θα περιέχει ως τιμή μία λίστα με τις θέσεις μέσα στον πίνακα is_occupied του κάθε κομματιού του εκάστοτε πλοίου και το δεύτερο θα περιέχει ως τιμή μία λίστα με τις θέσεις μέσα στον πίνακα is_occupied του κάθε κομματιού περιμετρικά του εκάστοτε πλοίου.

Το “adjacent” δηλαδή το γεγονός πως τα πλοία δεν πρέπει να έχουν επαφή μεταξύ τους το αποφασίσαμε λαμβάνοντας υπόψη τον αλγόριθμο επιλογής του εχθρού. Αν ο εχθρός έχει πετύχει πλοίο και ψάχνει να βρει τα κομμάτια του με σκοπό να το καταστρέψει, αν κατά τη διαδικασία αυτή πετύχει κατά λάθος κάποιο άλλο γειτονικό πλοίο θα μπερδευτεί σχετικά με το ποιο πλοίο αναζητά να καταστρέψει, καθώς δεν του είναι γνωστές οι συντεταγμένες των πλοίων μέσα στο ταμπλό.

Έχοντας λοιπόν τις πληροφορίες αυτές ξεκινήσαμε να κατασκευάζουμε τις μεθόδους.

Αρχικά θέλαμε μία μέθοδο η οποία να εκτυπώνει το ταμπλό, δηλαδή τον πίνακα is_occupied, καθώς και τις συντεταγμένες κάθε πλοίου και τις συντεταγμένες της περιοχής γύρω από αυτό.

Τη μέθοδο αυτήν την ονομάσαμε print και όταν την τρέχουμε με αρχικοποιημένο αντικείμενο παίρνουμε το εξής αποτέλεσμα:



```
IDLE Shell 3.11.1
File Edit Shell Debug Options Window Help
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\Dienix\Desktop\ΕΑΠ\ΠΛΗ ΠΡΟ\Ασκήσεις\Project 2022 - 2023\BattleShip Game Final - No Sound\Ship_Class.py
>>> friend = Ship()
>>> friend.print()
(0) (1) (2) (3) (4) (5) (6) (7) (8) (9) (10)
(0) 0 0 0 0 0 0 0 0 0 0 0
(1) 0 0 0 0 0 0 0 0 0 0 0
(2) 0 0 0 0 0 0 0 0 0 0 0
(3) 0 0 0 0 0 0 0 0 0 0 0
(4) 0 0 0 0 0 0 0 0 0 0 0
(5) 0 0 0 0 0 0 0 0 0 0 0
(6) 0 0 0 0 0 0 0 0 0 0 0
(7) 0 0 0 0 0 0 0 0 0 0 0
(8) 0 0 0 0 0 0 0 0 0 0 0
(9) 0 0 0 0 0 0 0 0 0 0 0

Aircraft Carrier:
Δεν έχει τοποθετηθεί ακόμα.

Battleship:
Δεν έχει τοποθετηθεί ακόμα.

Cruiser:
Δεν έχει τοποθετηθεί ακόμα.

Destroyer:
Δεν έχει τοποθετηθεί ακόμα.

>>> |
```

Ln: 40 Col: 0

Το αμέσως επόμενο βήμα είναι να προσθέσουμε πλοία στο ταμπλό. Για να γίνει επιτυχώς αυτό θα πρέπει, αρχικά τα πλοία προς τοποθέτηση να μη ξεπερνάνε τις διαστάσεις του πίνακα και επιπλέον να μην ακουμπάνε πάνω σε άλλο πλοίο καθώς και στο χώρο γύρω από αυτό.

Έτσι φτιάξαμε μία μέθοδο στην οποία δίνουμε τις αρχικές συντεταγμένες στις οποίες θέλουμε να τοποθετήσουμε την κεφαλή του πλοίου, τον προσανατολισμό του πλοίου καθώς και το μήκος του. (Σχετικά με το αν η κεφαλή του πλοίου ακουμπάει άλλο πλοίο, αποφασίσαμε να αντιμετωπίσουμε το πρόβλημα αυτό με τη βοήθεια του γραφικού περιβάλλοντος, καθώς όταν θα προστίθεται ένα πλοίο στο ταμπλό θα απενεργοποιούνται και τα αντίστοιχα κουμπιά.)

Σειρά έχει η μέθοδος τοποθέτηση πλοίου η οποία τοποθετεί το πλοίο στον πίνακα αντικαθιστώντας τα 0 με *. Τέλος φτιάξαμε μία μέθοδο η οποία αντικαθιστά όλα τα 0 γύρω από το πλοίο με 1.

Χρησιμοποιώντας τις παραπάνω μεθόδους με σκοπό την εισαγωγή πλοίου στη θέση [3, 4] έχουμε τα εξής αποτελέσματα:

```

IDLE Shell 3.11.1
File Edit Shell Debug Options Window Help
Python 3.11.1 (tags/v3.11.1:a7a450f, Dec 6 2022, 19:58:39) [MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\Dienix\Desktop\ΕΑΠ\ΠΑΗ ΠΡΟ\Ασκήσεις\Project 2022 - 2023\BattleShip Game Final - No Sound\Ship_Class.py
>>> friend = Ship()
>>> if friend.check_in_direction([3,4], "up", friend.AIRCRAFT_CARRIER):
...     friend.place([3,4], "up", friend.AIRCRAFT_CARRIER)
...     friend.fill_adjacent_space("up", friend.AIRCRAFT_CARRIER)
... else:
...     print("Το πλοίο δεν μπορεί να τοποθετηθεί")
...
... To πλοίο δεν μπορεί να τοποθετηθεί
>>> if friend.check_in_direction([3,4], "right", friend.AIRCRAFT_CARRIER):
...     friend.place([3,4], "right", friend.AIRCRAFT_CARRIER)
...     friend.fill_adjacent_space("right", friend.AIRCRAFT_CARRIER)
... else:
...     print("Το πλοίο δεν μπορεί να τοποθετηθεί")
...
...
>>> friend.pprint()
(0) (1) (2) (3) (4) (5) (6) (7) (8) (9) (10)
(0) 0 0 0 0 0 0 0 0 0 0 0
(1) 0 0 0 0 0 0 0 0 0 0 0
(2) 0 0 0 1 1 1 1 1 1 1 0
(3) 0 0 0 1 * * * * 1 0
(4) 0 0 0 1 1 1 1 1 1 1 0
(5) 0 0 0 0 0 0 0 0 0 0 0
(6) 0 0 0 0 0 0 0 0 0 0 0
(7) 0 0 0 0 0 0 0 0 0 0 0
(8) 0 0 0 0 0 0 0 0 0 0 0
(9) 0 0 0 0 0 0 0 0 0 0 0

Aircraft Carrier:
[3, 4][3, 5][3, 6][3, 7][3, 8] [3, 3][2, 3][4, 3][2, 4][4, 4][2, 5][4, 5][2, 6][4, 6][2, 7][4, 7][2, 8][4, 8][3, 9][2, 9][4, 9]

Battleship:
Δεν έχει τοποθετηθεί ακόμα.

Cruiser:
Δεν έχει τοποθετηθεί ακόμα.

Destroyer:
Δεν έχει τοποθετηθεί ακόμα.
>>> |

```

Τελικά αποφασίσαμε ότι χρειαζόμαστε μία μέθοδο για να αφαιρούμε πλοία η οποία θα καθάριζε τα δεδομένα από τα λεξικά του εκάστοτε πλοίου και μία μέθοδο ακόμα η οποία θα ενημέρωνε το ταμπλό για τις όποιες αλλαγές. Είναι δηλαδή δύο μέθοδοι οι οποίες εκτελούνται κατά κύριο λόγο μαζί.

Έτσι αν θέλουμε να αφαιρέσουμε το παραπάνω πλοίο και να καθαρίσει το ταμπλό θα έχουμε:

```

IDLE Shell 3.11.1
File Edit Shell Debug Options Window Help
(4) 0 0 0 1 1 1 1 1 1 0
(5) 0 0 0 0 0 0 0 0 0 0
(6) 0 0 0 0 0 0 0 0 0 0
(7) 0 0 0 0 0 0 0 0 0 0
(8) 0 0 0 0 0 0 0 0 0 0
(9) 0 0 0 0 0 0 0 0 0 0

Aircraft Carrier:
[3, 4][3, 5][3, 6][3, 7][3, 8] [3, 3][2, 3][4, 3][2, 4][4, 4][2, 5][4, 5][2, 6][4, 6][2, 7][4, 7][2, 8][4, 8][3, 9][2, 9][4, 9]

Battleship:
Δεν έχει τοποθετηθεί ακόμα.

Cruiser:
Δεν έχει τοποθετηθεί ακόμα.

Destroyer:
Δεν έχει τοποθετηθεί ακόμα.

>>> friend.remove(friend.AIRCRAFT_CARRIER)
>>> friend.is_occupied_update()
>>> friend.print()
(0) (1) (2) (3) (4) (5) (6) (7) (8) (9) (10)
(0) 0 0 0 0 0 0 0 0 0 0 0
(1) 0 0 0 0 0 0 0 0 0 0 0
(2) 0 0 0 0 0 0 0 0 0 0 0
(3) 0 0 0 0 0 0 0 0 0 0 0
(4) 0 0 0 0 0 0 0 0 0 0 0
(5) 0 0 0 0 0 0 0 0 0 0 0
(6) 0 0 0 0 0 0 0 0 0 0 0
(7) 0 0 0 0 0 0 0 0 0 0 0
(8) 0 0 0 0 0 0 0 0 0 0 0
(9) 0 0 0 0 0 0 0 0 0 0 0

Aircraft Carrier:
Δεν έχει τοποθετηθεί ακόμα.

Battleship:
Δεν έχει τοποθετηθεί ακόμα.

Cruiser:
Δεν έχει τοποθετηθεί ακόμα.

Destroyer:
Δεν έχει τοποθετηθεί ακόμα.

>>> |

```

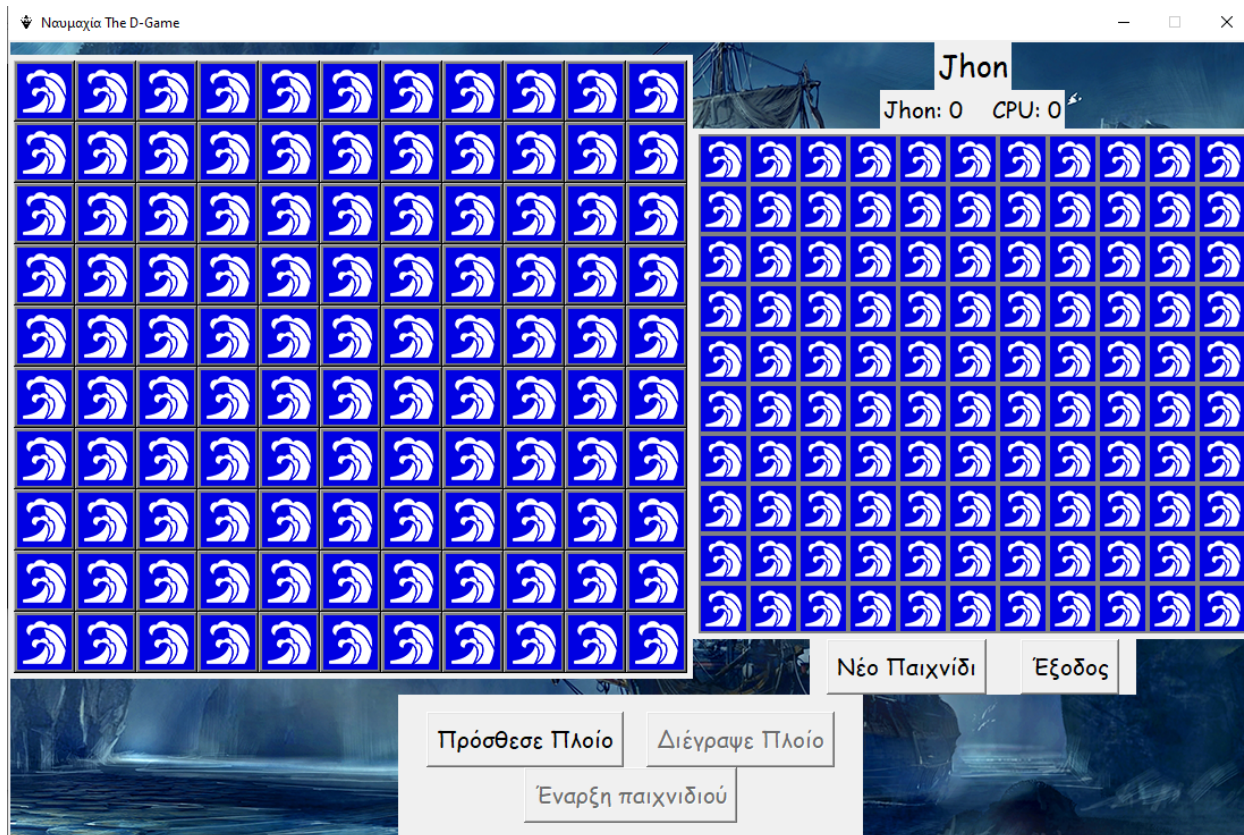
Κατασκευή Γραφικού Περιβάλλοντος

Για την κατασκευή του γραφικού περιβάλλοντος, η ιδέα ήταν απλή. Αποφασίσαμε ένα πλαίσιο το οποίο θα περιέχει:

- αριστερά έναν πίνακα από κουμπιά μεγέθους (10x11) με τα οποία θα γίνεται:
 - σε πρώτη φάση, η εισαγωγή των πλοίων του εκάστοτε παίχτη
 - και σε δεύτερη φάση, η επιλογή προς καταστροφή τμήματος από το ταμπλό του εχθρού
- δεξιά έναν πίνακα από ταμπέλες οι οποίες μετά την έναρξη του παιχνιδιού θα αντιπροσωπεύουν τον χώρο πάνω στον οποίο θα ψάχνει ο εχθρός
- στην πάνω δεξιά γωνία, το όνομα του εκάστοτε παίχτη καθώς και τη βαθμολογία των δύο αντιπάλων

- στην κάτω δεξιά γωνία, δύο κουμπιά για την επανέναρξη του παιχνιδιού ή την έξοδο από το παιχνίδι
- και στο κάτω μέρος τρία κουμπιά για την εισαγωγή πλοίου, τη διαγραφή αυτού καθώς και την έναρξη παιχνιδιού υπό προϋποθέσεις.

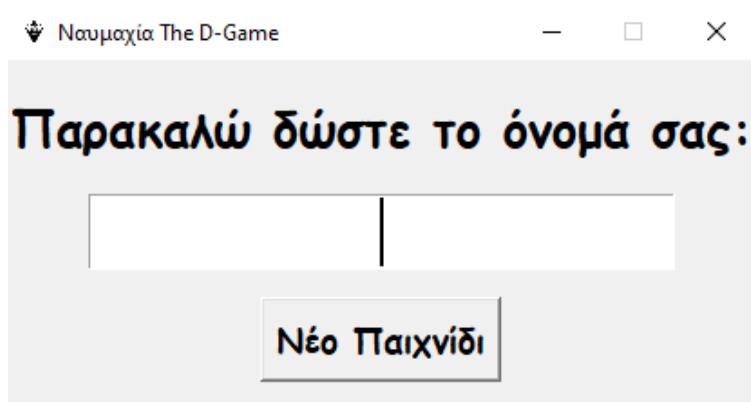
Το τελικό γραφικό περιβάλλον για το παιχνίδι μας είχε αυτή τη μορφή:



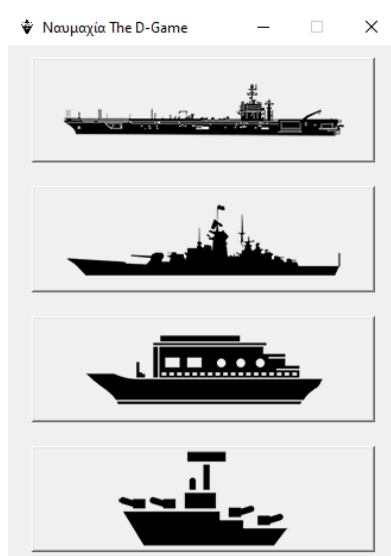
Για τη χρήση όλων αυτών των τμημάτων του παιχνιδιού αποφασίσαμε ότι θα χρησιμοποιήσουμε επιπλέον παράθυρα με τα οποία ο χρήστης θα παρέχει πληροφορίες στο παιχνίδι.

Αρχικά αποφασίσαμε να φτιάξουμε ένα παράθυρο με το οποίο ο χρήστης θα δίνει το όνομά του. Αυτό θα περιέχει μία ταμπέλα που ρωτάει τον χρήστη για το όνομά του, ένα πεδίο στο οποίο ο χρήστης μπορεί να συμπληρώσει το όνομά του και ένα κουμπί για επιβεβαίωση. Ακόμη δίνεται η δυνατότητα στον χρήστη να επιβεβαιώσει την απάντηση του πατώντας το Enter.

Το παράθυρο αυτό έχει την πιο κάτω μορφή:

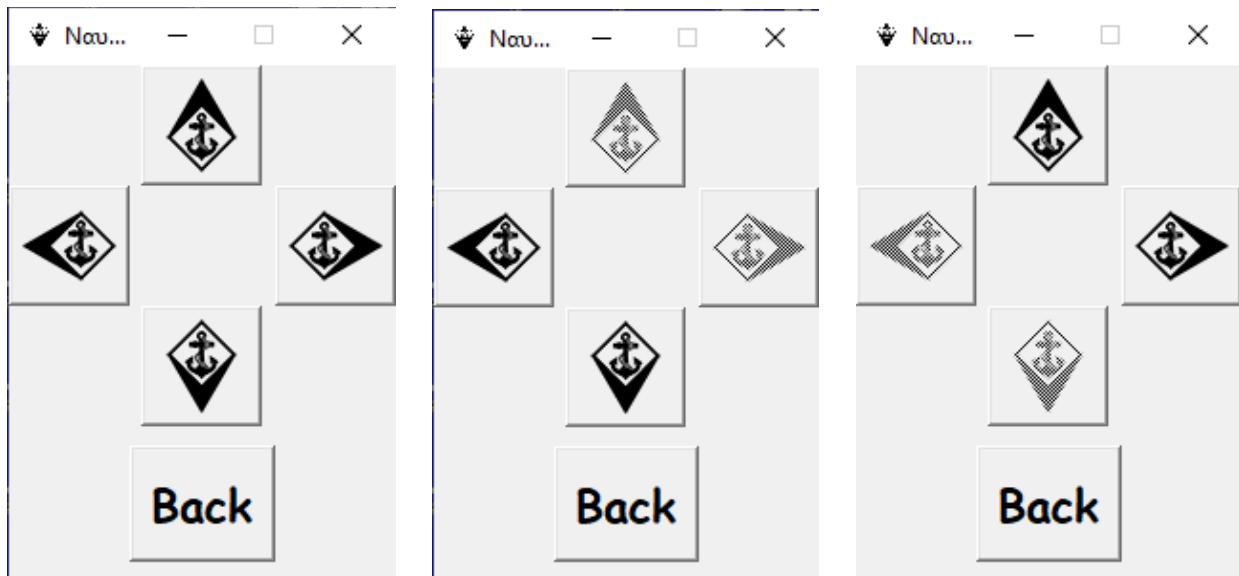


Έπειτα φτιάξαμε ένα παράθυρο με το οποίο ο χρήστης θα δίνει πληροφορίες για το πλοίο που επιθυμεί να τοποθετήσει. Αυτό περιέχει 4 κουμπιά τα οποία αντιπροσωπεύουν κάθε ένα από τα τέσσερα είδη πλοίων του παιχνιδιού. Το παράθυρο είναι το παρακάτω:



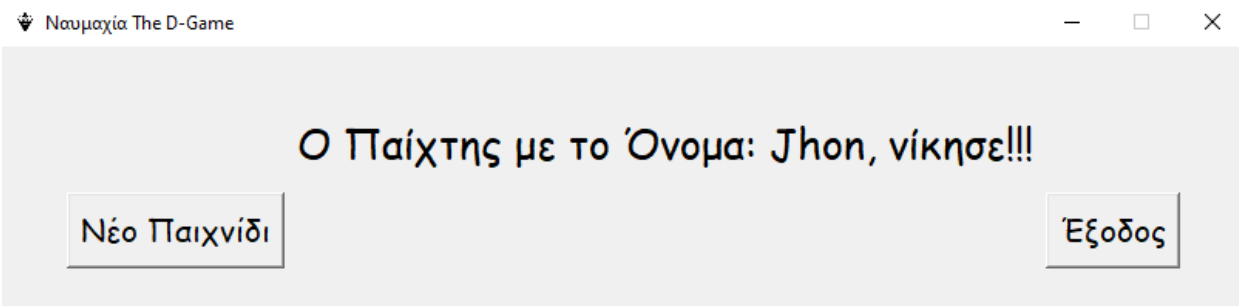
Επιπλέον κατασκευάσαμε ένα παράθυρο με το οποίο ο χρήστης θα εισάγει δεδομένα τα οποία θα καθορίζουν την κατεύθυνση του πλοίου, δηλαδή προς τα ποια μεριά θα τοποθετηθεί. Το παράθυρο αυτό έχει πέντε κουμπιά. Τέσσερα για την κατεύθυνση που επιθυμεί να δώσει ο χρήστης και ένα σε περίπτωση που ο χρήστης θέλει να επιστρέψει. Τα κουμπιά της κατεύθυνσης εμφανίζονται απενεργοποιημένα σε περίπτωση που το πλοίο που θέλει να τοποθετήσει ο χρήστης αντιμετωπίσει κάποιο εμπόδιο κατά την τοποθέτηση του, όπως στην περίπτωση να ξεπεράσει τα όρια του πίνακα ή να πέσει πάνω σε άλλο πλοίο. Για παράδειγμα αν ο χρήστης επιλέξει να τοποθετήσει πλοίο στην πάνω δεξιά γωνία, τότε το πάνω και το δεξιά κουμπί θα εμφανιστούν απενεργοποιημένα. Αντίστοιχα στην περίπτωση που ο χρήστης επιλέξει την κάτω αριστερή γωνία.

Το παράθυρο της κατεύθυνσης έχει την πιο κάτω μορφή:

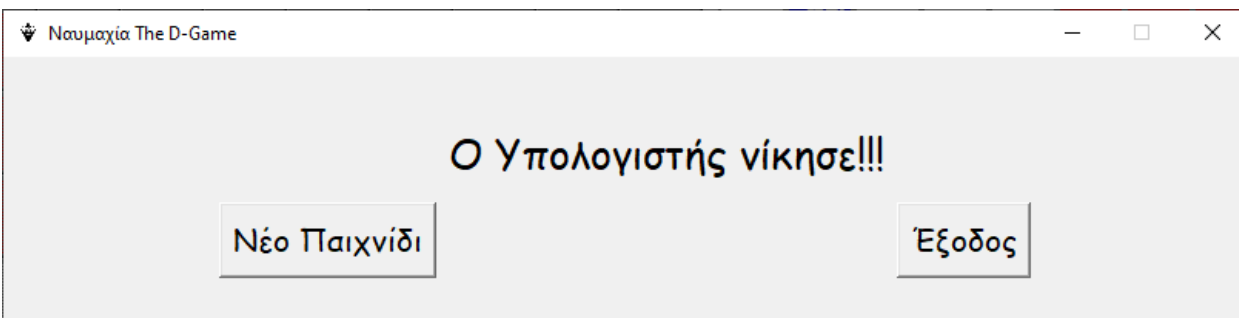


Τέλος φτιάξαμε ένα παράθυρο το οποίο εμφανίζεται με τη λήξη του παιχνιδιού και εμφανίζει ένα μήνυμα ανάλογα με το αν ο χρήστης κέρδισε ή έχασε. Το παράθυρο αυτό έχει μία ταμπέλα με το μήνυμα και δύο κουμπιά για επανέναρξη του παιχνιδιού ή έξοδο από αυτό.

Το παράθυρο έχει τις δύο παρακάτω μορφές:



και



Κατασκευή Αλγορίθμου Επιλογής του Εχθρού

Για την κατασκευή του Αλγορίθμου του εχθρού ο τρόπος σκέψης ήταν ο εξής: Θέλουμε ο εχθρός να σκέφτεται και να επιλέγει την επόμενη κίνησή του σαν άνθρωπος. Αυτό σημαίνει ότι δεν πρέπει απλά να κάνει τυχαίες επιλογές μέσα στο ταμπλό αλλά να έχει τη ευφυΐα να ψάχνει ολόκληρο το πλοίο, αφού φυσικά έχει βρει πρώτα ένα.

Τα βήματα που ακολουθήσαμε για να το πετύχουμε αυτό, αρχικά περιέχουν τη μελέτη του τρόπου με τον οποίο σκέφτεται κάποιος που παίζει αυτό το παιχνίδι. Αναλύσαμε τον απλό αυτόν αλγόριθμο ως εξής:

1. Επέλεξε μία τυχαία συντεταγμένη μέσα από το ταμπλό.
2. Αν η συντεταγμένη αυτή:
 - a. Είναι εχθρικό πλοίο τότε ψάξε με τη σειρά στις 4 αμέσως κοντινές συντεταγμένες κυκλικά ξεκινώντας αρχικά από πάνω.
 Αν η συντεταγμένη αυτή:
 - i. Είναι εχθρικό πλοίο τότε το πλοίο αυτό βρίσκεται στον άξονα $x'x$, αν η συντεταγμένη αυτή βρίσκεται δεξιά ή αριστερά της αρχικής, αλλιώς στον άξονα $y'y$.
 1. Στον άξονα του πλοίου ψάξε όλες τις κοντινές συντεταγμένες μέχρι να βρεις τις συντεταγμένες αμέσως μετά τα δύο άκρα του πλοίου.
 2. Ξεκίνα ξανά από την αρχή.
 - ii. Δεν είναι εχθρικό πλοίο έλεγξε την αμέσως επόμενη σε σειρά συντεταγμένη του βήματος a.
 - b. Δεν είναι εχθρικό πλοίο τότε ξεκίνα ξανά από την αρχή.

Αναλύοντας αυτόν τον αλγόριθμο τον μετατρέψαμε σε κώδικα στη γλώσσα Python.

Ο κώδικας αυτός αποτελείται από τρία κομμάτια:

1. Μία συνάρτηση η οποία ελέγχει αν γίνεται αναζήτηση προς την επιλεγμένη κατεύθυνση
2. μία συνάρτηση η οποία πραγματοποιεί την αναζήτηση προς την επιλεγμένη κατεύθυνση
3. και μία συνάρτηση η οποία με κάποιες επιπλέον γραμμές κώδικα και χρησιμοποιώντας τις δύο παραπάνω συναρτήσεις πραγματοποιεί την αναζήτηση του εχθρού.

Κατασκευή του Τελικού Προγράμματος

Για την κατασκευή του τελικού προγράμματος χρειάστηκαν συνδυαστικές κατευθύνσεις και υποδείξεις από όλα τα μέλη της ομάδας.

Η διαδικασία αυτή περιλαμβάνει τη δημιουργία συναρτήσεων για την εμφάνιση και την απόκρυψη παραθύρων όπου χρειάζεται, την ενεργοποίηση ή απενεργοποίηση καθώς και την απόδοση κατάλληλων συναρτήσεων σε κουμπιά με σκοπό τη σωστή και αποδοτική λειτουργία του προγράμματος.

Διευκρινήσεις

Στη παρούσα αναφορά δεν επεξηγείται ακριβώς ο τρόπος λειτουργίας του κώδικα του προγράμματος καθώς αυτό ζητείται ξεχωριστά για κάθε αντικείμενο που ασχολήθηκε ο εκάστοτε φοιτητής από τον ίδιο. Έχει πραγματοποιηθεί μία περιληπτική αναφορά, με ακρίβεια και αρκετό βάθος, για τα κύρια μέρη και τις λειτουργίες του προγράμματος καθώς και του τρόπου εργασίας, της κατανομής των επιμέρους εργασιών, και των επιθυμητών αλλά και των επιτευχθέντων αποτελεσμάτων των επιδιώξεων μας.

Βιβλιογραφία

Ο κώδικας της εργασίας δεν έχει αντιγραφεί από πουθενά και είναι αποκλειστικά δική μας δημιουργία. Οι γνώσεις που χρειαστήκαμε για την κατασκευή του προγράμματος προήλθαν αποκλειστικά από το διδακτικό υλικό που παρέχει το ΕΑΠ το οποίο περιλαμβάνει τα εξής βιβλία:

1. Μανής, Γ., 2015. Εισαγωγή στον Προγραμματισμό με αρωγό τη γλώσσα Python. Αθήνα: Σύνδεσμος Ελληνικών Ακαδημαϊκών Βιβλιοθηκών.
Διαθέσιμο στο σύνδεσμο: <http://hdl.handle.net/11419/2745>
2. Μαγκούτης, Κ., Νικολάου, Χ., 2015. Εισαγωγή στον αντικειμενοστραφή προγραμματισμό με Python. Αθήνα: Σύνδεσμος Ελληνικών Ακαδημαϊκών Βιβλιοθηκών.
Διαθέσιμο στο σύνδεσμο: <http://hdl.handle.net/11419/1708>
3. Αγγελιδάκης, Ν.Α., 2015. Εισαγωγή στον προγραμματισμό με την Python, Διαθέσιμο στο σύνδεσμο: <http://aggelid.mysch.gr/pythonbook/>
4. Swaroop, C. H. 2013. A Byte of Python, Επιμ. μετάφρασης Ubuntu community,
Διαθέσιμο στο σύνδεσμο: http://dide.flo.sch.gr/Plinet/Meetings/Meeting23/A_Byte_of_Python-el.pdf

Πρωτότυπο στο σύνδεσμο: <https://open.umn.edu/opentextbooks/textbooks/581>

Έρευνα για το μέγεθος και τις ιδιότητες των διάφορων πλοίων έγινε από τη Wikipedia στον παρακάτω σύνδεσμο: https://en.wikipedia.org/wiki/Naval_ship#Size

Όλα τα εικονίδια που χρησιμοποιήθηκαν κατασκευάστηκαν από εμάς ή επιλέχθηκαν από τον σύνδεσμο: <https://icon-icons.com>