CHESS

Ομάδα 37

Ρηγίνος Σάββας
Ιωάννης Σιντόρης
Χριστιάνα Σωτηρίου
Σμαράγδα Τσαναή
Αλέξανδρος Τσαπάρας
Στέφανος Φαρδέλλας

CHESS

Πρόλογος

Είμαστε η ομάδα 37 και στόχος μας ήταν η εκπόνηση του παιχνιδιού <<σκάκι>> , μέσω της γλώσσας προγραμματισμού "Python" και της βιβλιοθήκης "tkinter" . Για την πραγματοποίηση της ομαδικής εργασίας συμμετείχαν εξίσου όλα τα μέλη της ομάδας, όπου υπό κλίμα καλής θέλησης, αποφασιστικότητας, συνεργασίας και ανερμάτιστης δουλειάς, ευοδώθηκε η ψυχική και συνάμα πνευματική προσπάθειά μας.
Για τον κώδικα του παιχνιδιού στηριχθήκαμε σε τρεις (3) κλάσεις, την MyApp, Piece και ChessGameMainMenu. Το κάθε άτομο του γκρουπ είχε αναλάβει συγκεκριμένα καθήκοντα, για την ορθότερη διεκπεραίωση του προτζεκτ. Πιο συγκεκριμένα:

§ Ο Στέφανος με τον Ιωάννη ασχολήθηκαν με τις μεθόδους, μέσω των οποίων επιτυγχάνεται η κίνηση των πιονιών στην σκακιέρα και τους περιορισμούς στους οποίους αυτά υπάγονται.
§ Ο Ρηγίνος αφιέρωσε χρόνο στην δημιουργία του αρχικού μενού.
§ Η Σμαράγδα σχεδίασε με λεπτομέρεια την σκακιέρα και τις πιθανές κινήσεις κάθε πιονιού, δίνοντας το έναυσμα για την έναρξη αυτής της ομαδικής εργασίας.
§ Ο Αλέξανδρος υλοποίησε το τελικό μενού, το σκορ και άφησε το στίγμα του στην λήξη του παιχνιδιού.
§ Η Χριστιάνα ανακάλυψε πως θα εισάγουμε την καταμέτρηση χρόνου στο παιχνίδι και συνεργάστηκε με την Σμαράγδα για την σύνταξη των δυνατών κινήσεων των πιονιών.

CHESS

Το πρόγραμμα ξεκινάει με το <<κάλεσμα>> της tkinter, μιας ενσωματωμένης βιβλιοθήκης της γλώσσας "Python", η οποία παρέχει την δυνατότητα χειρισμού γεγονότων, που προκαλούνται από ενέργειες του χρήστη, αλλά και την χρήση των επονομαζόμενων "widgets", τα οποία είναι παράθυρα και άλλοι υποδοχείς, μέσα στους οποίους εμφανίζονται άλλα γραφικά στοιχεία, όπως κουμπιά επιλογών, εικόνες, μενού.

Η ΚΛΑΣΗ MyApp

Όπως ήδη φαίνεται από τον χώρο που καταλαμβάνει η "MyApp" στον κώδικα, είναι η κλάση με την μεγαλύτερη έκταση από τις άλλες δυο. Η "MyApp" περιέχει 28 συναρτήσεις: __init__(constructor), root_center, chess_score, countdown1, countdown2, countdown3, countdown4, create_canvas, black_play, white_play, images, position, targetisvalid, userattack, aiattack, colorpossmoves, get_possmoves, game_ends, announcement, clicked, menu, restart, block, the_end, end_round,next_round restart_game και exit_game.
Αρχικά, η μέθοδος __init__ έχει έξι (6) ορίσματα, τα self, root, color1, color2, cs1, cs2. Η μέθοδος αυτή ορίζει πως θα συμπεριφερθεί κάθε αντικείμενο που δημιουργεί η κλάση MyApp και το όρισμα self, το χρειάζεται για να αναφέρεται σε κάθε αντικείμενο που εμείς δημιουργούμε.Έτσι, δημιουργούμε το παράθυρο root, στο οποίο λαμβάνουν χώρα, όλα τα συμβάντα αναφορικά με το παιχνίδι. Το παράθυρο ονομάζεται ChessBoard, και με την εντολή resizable απαγορεύεται να αλλάξουν οι διαστάσεις του παραθύρου, σε όποιον υπολογιστή και αν τρέξει το

πρόγραμμα. Επίσης, το παράθυρο έχει διαστάσεις 1120 στον άξονα των χ και 640 στον άξονα των y. Ακόμη, εάν ο ένας χρήστης διαλέξει το άσπρο χρώμα, τότε ο άλλος θα έχει τα μαύρα και αντίστροφα. Στην συνέχεια, καλείται η μέθοδος create_canvas, μέσω της οποίας ορίζεται ένας χώρος, στον οποίο θα εισαχθούν άλλα γραφικά στοιχεία, στην περίπτωσή μας γραμμές και χρώματα, που θα
σχηματίζουν τα τετράγωνα της σκακιέρας, και τα GIF που αναπαριστούν τα πιόνια. Ο καvβάς <<ανήκει>> στο αρχικό παράθυρο root (που έχει τον τίτλο :" ChessBoard" , έχει ύψος 600 εικονοστοιχεία και πλάτος 600 εικονοστοιχεία (pixels). Θέτουμε σε μια μεταβλητή "color" ένα χρώμα από τα δύο("white" or "royalblue"), το οποίο επιλέχθηκε να είναι το "royalblue" . Μετέπειτα, με τη βοήθεια ενός βρόχου επανάληψης πραγματοποιούμε 8 φορές(όσες και οι σειρές της σκακιερας) μια διαδικασία. Στη διαδικασία αυτή η τιμή της μεταβλητής color μετατρέπεται στο άλλο χρώμα της σκακιέρας και έπειτα με μια ακόμα επανάληψη για όσες είναι οι στήλες της σκακιέρας δημιουργούμε ένα τετραγωνάκι χρώματος color .Τέλος, μέσα στην δεύτερη επανάληψη ακολουθούμε την ίδια διαδικασία εναλλαγής μεταξύ των 2 χρωμάτων, καθώς αν το χρώμα είναι άσπρο το μετατρέπουμε σε "royalblue" και το αντίστροφο. Συνοπτικά, η πρώτη επανάληψη δίνεται το χρωμα του πρώτου τετραγωνακίου μιας σειράς, ενώ με τη δεύτερη επανάληψη σχηματίζονται 8 τετραγωνάκια εναλασσόμενων χρωμάτων .Η μέθοδος ολοκληρώνεται με την χρήση της μεθόδου της βιβλιοθήκης "tkinter" , "bind" , η οποία περιμένει ένα γεγονός από τον χρήστη, ώστε να δώσει εντολή να λάβει έναρξη μία άλλη μέθοδος. Στον κώδικα αυτό, μόλις πατηθούν από τον χρήστη το δεξί κλικ(""), η το αριστερό κλικ("") και το πλήκτρο ελέγχου "esc "(""), καλεί τις μεθόδους "black_play" , "white_play" και "menu" αντίστοιχα.
Συνεχίζοντας με την μέθοδο constructor από εκεί που την

αφήσαμε, με την μεταβλητή "self.symbols= "PRNBQK" , έχουμε συγκεντρωμένα όλα τα αρχικά γράμματα των πιονιών και έτσι φτιάχνουμε αντικείμενα της κλάσης Piece, τα οποία έχουν τα τέσσερα ορίσματα της, το s που είναι το σύμβολο(symbol) του πιονιού, το self.color1-self.color2 χρώμα(color) του, τις συντεταγμένες(position) του τετραγώνου που θα τοποθετηθεί αρχικά, (πχ το αριστερά-πάνω τετραγωνάκι είναι το (1,1), το δεξιά του είναι το (1,2) κ.ο.κ.) και τέλος την αξία(value) του, δηλαδή πόσους πόντους κερδίζει ο αντίπαλος αν αιχμαλωτήσει το συγκεκριμένο πιόνι. Οπότε δημιουργήσαμε το κάθε πιόνι έτσι.

Οι μέθοδοι countdown

Οι μέθοδοι countdown2, countdown4 αναφέρονται στα άσπρα πιόνια, ενώ οι μέθοδοι countdown1,3 σχετίζονται με τα πιόνια μαύρου χρώματος. Με το που ξεκινάει το παιχνίδι, καλούμε την countdown2, της οποίας το χρονόμετρο αρχίζει να τρέχει, δίχως κάποιο event από τον χρήστη. Μετά την σηματοδότηση του πρώτου <<κλικ>> του πρώτου χρονομέτρου(clock2), χρησιμοποιούμε την μέθοδο "destroy" για την <<καταστροφή>> του και δημιουργούμε το δεύτερο χρονόμετρο(clock4), το οποίο αναφέρεται εξίσου στα άσπρα, καλώντας την countdown 4. Το αξιοσημείωτο είναι ότι ο χρόνος του δεύτερου χρονομέτρου αρχίζει να κυλάει με σημείο έναρξης το σημείο διακοπής μέτρησης χρόνου του πρώτου . Αφότου πραγματοποιηθεί το δεύτερο <<κλικ>>, διαγράφεται το δεύτερο χρονόμετρο και δημιουργούμε το τρίτο χρονόμετρο(clock1) και καλούμε την countodown1, η οποία κάνει ακριβώς το ίδιο πράγμα και για τα μαύρα πιόνια. Όταν τελειώσει η πρώτη κίνηση των μαύρων σκακιστικών αντικειμένων, επαναλαμβάνεται αέναα η ίδια διαδικασία.
Οι μέθοδοι countdown έχουν δύο ορίσματα, το self και το

remainingx, όπου x είναι ο αριθμός της κάθε countdown, το οποίο έχει μία αρχική τιμή "None" και αν το remainingx δεν έχει αυτή την αρχική τιμή, τότε την τιμή του παίρνει η μεταβλητή self.ramainingx, που είχε οριστεί στην __init__ και αν είναι μικρότερη του μηδενός, εμφανίζει στον χρήστη το μήνυμα "time' s up!". Άν όμως self.remainingx >0, μέσω της "configure" το ρολόι ξεκινάει να τρέχει, μετρώντας αντίστροφα.

Το αρχικό μενού(Η κλάση ChessGameMainMenu)

Η κλάση αυτή αποτελείται από τέσσερις(4) μεθόδους, την συνάρτηση __init__, root_center(όπως και στην MyApp), την ok και GetGameSetupParams.
Η πρώτη δημιουργεί ένα Frame στο οποίο δημιουργούνται widgets, όπως Labels, Entries και Buttons. Ο παίκτης έχει την δυνατότητα να ορίσει το όνομα, το οποίο θα αντιστοιχεί σε κάθε πλευρά(μαύρα πιόνια-άσπρα πιόνια). Όσον αφορά την αποθήκευση των επιλογών του και την ομαλή συνέχεια του παιχνιδιού, υπάρχει το αντίστοιχο "Button" ("b").
Η root_center δέχεται ως ορίσματα τις συντεταγμένες του παραθύρου που επιθυμούμε να φτιάξουμε. <<Ζητάει>> το μάκρος και το ύψος της οθόνης οποιουδήποτε υπολογιστή και βρίσκει το επιτρεπτό κενό, που πρέπει να έχει το παράθυρο("ChessBoard") από τις άκρες του υπολογιστή και ως προς τους δυο κάθετους άξονες, ώστε το κέντρο του παραθύρου και του υπολογιστή να ταυτίζονται. Αυτή η μέθοδος ήταν τόσο χρήσιμη, που την χρησιμοποιήσαμε σε όλες τις κλάσεις που δημιουργήσαμε.
Αν πατηθεί το "Button b" ή "Enter" του πληκτρολογίου , τότε δέχεται τα καινούργια ονόματα και χρώματα πιονιών

που έχει ο καθένας, μέσω της .get(), αλλά αν δεν υπάρξει κάποια μεταβολή παραμένουν τα ίδια και τα δέχεται αυτομάτως το πρόγραμμα. Το παιχνίδι περιμένει μέχρι να γίνει το εναρκτήριο κλικ ουσιαστικά, που καθορίζει αν θα διατηρηθούν τα ήδη προεπιλεγμένα χρώματα και παικτών. Αυτή ήταν η δουλειά της ήδη-στην-__init__-ορισμένης self.count, η οποία ελέγχει αν πατήθηκε το κουμπί.

Από την πλευρά της η λίστα datas, ενισχύεται με τα self.p1n, self.p2n, self.p1c, self.p2c, που είναι οι επιλογές που έκαναν οι παίκτες.
Έπειτα, δημιουργείται ένα αρχείο("GAME_OPTIONS.txt"), το οποίο περιέχει τις παραπάνω αναφερόμενες επιλογές, που θα χρησιμοποιήσουμε στην επόμενη κλάση("MyApp").
Τέλος, καλείται η μέθοδος "GetGameSetupParams" και καταστρέφουμε το συγκεκριμένο παράθυρο(το αρχικό μενού), ενώ <<λαμβάνει δράση>> η "main" και μας κατευθύνει κατευθείαν στο παιχνίδι.

Το τέλος του παιχνιδιού

Το τελικό στάδιο του παιχνιδιού διαμορφώνεται σύμφωνα με τις μεθόδους game_ends, announcement, clicked, menu, the_end, end_round, next_round, restart_game και exit_game.
Το τέλος του παιχνιδιού καθορίζεται από δύο επιλογές του παίκτη:
1η) Να πατήσει το , το οποίο τον οδηγεί σε ένα παράθυρο (1), που του παρέχει την δυνατότητα να αποχωρήσει ή να ξεκινήσει το παιχνίδι πάλι από την αρχή(από το αρχικό μενού, η λειτουργία του φαίνεται σε προηγούμενη σελίδα).
2η) Να τερματίσει τον γύρο(ως νικητής), όπου μέσω της announcement(παίρνει ως όρισμα το κείμενο του νικητή)

αναγράφεται το όνομα του κερδισμένου και στην συνέχεια πατώντας το κουμπί "Skip", κατευθύνεται στο ίδιο, αλλά ανανεομένο παράθυρο (2), το οποίο παρουσιάζει ομοιότητες με το προηγούμενο, μόνο που τώρα έχει το προνόμιο να αποφασίσει άμα θα ξεκινήσει ο επόμενος γύρος.

Με την μέθοδο game_ends και τις μεταβλητές "userscore", "aiscore", αναδεικνύεται ο νικητής. Οι 5000 πόντοι, τους οποίους όποιος συγκεντρώσει πρώτος φεύγει με το έπαθλο της νίκης, δεν επιλέχθηκαν τυχαία. Ο βασιλιάς αξίζει 5000 πόντους, οπότε όποιος κατατροπώσει πρώτος τον βασιλία του άλλου νικάει.Επίσης, αν ο ένας παίκτης πάρει υπό την κατοχή του όλα τα σκακιστικά κομμάτια του αντιπάλου, εκτός όμως του βασιλιά, δεν εξασφαλίζει την επικράτησή του, αφού το άθροισμα των πόντων όλων των πιονιών είναι μικρότερο του 5000.
Η συνάρτηση "clicked" ελέγχει με την βοήθεια της μεταβλητής "count_click" αν πατηθεί το "Skip". Αν πατηθεί μία φορά <<καταστρέφεται>> το μήνυμα του νικητή, η σκακιέρα(κανβάς)(με την μέθοδο "destroy") και καλείται η end_round, δημιουργώντας το παράθυρο (2), το οποίο εκτελεί τις εντολές που αναφέρθηκαν πιο πάνω. Αν δεν πατηθεί το "Skip", το αρχικό παράθυρο(με την σκακιέρα, το σκορ και τα χρονόμετρα) αναμένει να πατηθεί το "Skip". Σε αυτό συμβάλει η εντολή wait_waindow.
Τέλος, οι μέθοδοι next_round, restart_game, exit_game, δίνουν την επιλογή της εκκίνησης ενός νέου γύρου, ορίζοντας το αντικείμενο "app" της κλάσης MyApp, μιας επανέναρξης ή της εξόδου απο το παιχνίδι.

Η κίνηση των πιονιών

CHESS

Για την κίνηση των πιονιών αποφασίσαμε να φτιάξουμε τις μεθόδους white_play, black_play, targetisvalid, userattack, aiattack και get_possmoves.
Στην white_play, όσον αφορά την κίνηση των πιονιών, με την βοήθεια της μεταβλητής self.counter, ενεργοποείται η κίνηση των λευκών πιονιών όταν είναι μονός ο αριθμός των <<κλικ>> και όταν είναι ζυγός, εκτελείται. Αν δεν πραγματοποιηθεί κάποια κίνηση ή υπάρξει λανθασμένη ενέργεια ο παίκτης ειδοποιείται με τα αντίστοιχα μηνύματα("try again" ," invalid move") και έχει την δυνατότητα να ξαναπροσπαθήσει. Πιο αναλυτικά, ελέγχει αν οι συντεταγμένες που δίνονται με τα μονά <<κλικ>> είναι θέσεις πιονιών και ορίζει σε μια μεταβλητή την εικόνα του πιονιού αυτού. Ακόμη, παρουσιάζει στον παίκτη τις δυνατές κινήσεις, που μπορεί να πραγματοποιήσει, χρωματίζοντας τα αντίστοιχα τετραγωνάκια, καλώντας την "colorpossmoves". Έπειτα, στο επόμενο <<κλικ>>, διαγράφονται τα χρωματισμένα τετραγωνάκια και στην συνέχεια με την βοήθεια της συνάρτησης "targetisvalid", ελέγχει αν μπορεί να μεταβεί το πιόνι στην θέση αυτή.
Καλώντας την "targetisvalid" <<ενεργοποιείται>> και η συνάρτηση "aiattack", η οποία έχει σαν αποτέλεσμα τηη διαγραφή του πιονιού, σε περίπτωση που ύπηρχε πιόνι του αντιπάλου στην θέση εκείνη. Η "targetisvalid" μας δίνει ένα αποτέλεσμα "True" or "False", ανάλογα με το αν μπορεί να μεταβεί ή όχι το πιόνι εκεί. Η κίνηση του πιονιού ωστόσο θα εκτελεστεί αν και μόνο αν το αποτέλεσμα της συνάρτησης είναι "True", αλλά και οι συντεταγμένες της θέσης που θέλει να μεταβεί το πιόνι περιέχονται στην λίστα "greenrectli", η οποία ανακλαείται από την μέθοδο "get_possmoves"

Σύμφωνα με τα παραπάνω, η μέθοδος "black_play" λειτουργεί σύμφωνα με την ίδια λογική, με την μόνη διαφορά να είναι ότι με το κάλεσμά της καλείται η "userattack".

CHESS

Προφανώς, με την "userattack" επιτίθενται τα μαύρα σκακιστικά αντικείμενα, ενώ η "aiattack" αναφέρεται στα λευκά πιόνια.

Όταν το στρατιωτάκι φθάσει στην αντίπαλη πλεύρα στην τελευταία επίτρεπτή σειρά εμφανίζεται ενά νέο παράθυρο(self.window), το οποίο δίνει στον παίκτη την δυνατότητα να «μετατρέψει» το στρατιωτάκι του(Pawn) σε βασιλισσα, ιππότη, πύργο, αξιωματικό ή να παραμείνει στρατιωτάκι. Αρχικά οι συναρτήσεις white_play και black_play εξετάζουν αν έχει φθάσει το στρατιωτάκι στην απέναντι τελευταία σειρά της σκακίερας. Με το που αναγνωρίσει ότι υπάρχει στρατιωτάκι σε αυτές τις θέσεις καλείτε για τα άσπρα η συνάρτηση pawn_change_white και αντίστοιχα για τα μαύρα η pawn_change_black. Οι συγκεκριμένες συναρτήσεις δημιουργούν ένα νέο παράθυρο στο οποίο ο χρήστης έχει στην διάθεση του 5 κουμπιά (κάθε κουμπί(Button) για τις αντίστοιχες παραπάνω επιλογές του). Το κάθε κουμπί καλεί μια συνάρτηση, η οποία ελέγχει αν ήδη έχουν δημιουργηθεί καλα «νέα» πιόνια στην σκακιέρα με την ίδια διαδικασία, θέτωντας όριο μεχρι 5 του ίδιου είδους. Αν δεν έχουν δημιουρθεί αλλα πιόνια, «κατασκευάζονται» με την συναρτηση αυτή πιόνια, τα οποια στην εικόνα, στην κίνηση και στην επίθεση ταυτίζονται με τα αρχίκα. Έπειτα καταστρέφεται το παράθυρο και συνεχίζεται το παιχνίδι.

Προβλήματα που αντιμετωπίσαμε και ολοκλήρωση ομαδικής εργασίας

Το μείζον πρόβλημα με την ομαδική εργασία ήταν αναφορικά με την κίνηση των πιονιών. Αρχικά δεν γνωρίζαμε πως ακριβώς να την επιτύχουμε. Ωστόσο, με αρκετό ψάξιμο και αφού κατανοήσαμε τον ευφυή τρόπο ελέγχου επιτρεπτής κίνησης του <<αλόγου>>, καταλάβαμε

πως θα πρέπει να διαμορφωθούν οι μέθοδοι πιόνια.
Ένα ακόμη θέμα, το οποίο αντιμετωπίστηκε την κυριολεκτικά τελευταία στιγμή, ήταν η επιθυμία μας να εμφανίζει τις πιθανές κινήσεις των πιονιών πάνω στην σκακιέρα, χρωματίζοντας τα αντίστοιχα τετραγωνάκια.
Οι συναντήσεις δεν ήταν πρόβλημα. Συναντιόμασταν στο σπίτι του Αλέξανδρου ή της Σμαράγδας, ήμαστε όλοι συνεπείς στα ραντεβού και με συνεχή μόχθο και ατελείωτες ώρες προγραμματισμού, διασφαλίστηκε η εύρυθμη λειτουργία του κώδικα.

Χρήσιμες πηγές πληροφοριών

https://commons.wikimedia.org/wiki/Category:PNG_chess_pieces/Standard_transparent
Την συγκεκριμένη πηγή την χρησιμοποιήσαμε για την λήψη των εικονιδίων (Gif), τις οποίες τοποθετήσαμε στον καμβά με την μέθοδο της tkinter την PhotoImage.