1ο Σετ Ασκήσεων: Συνδυαστικά Παιχνίδια

Μπάλλας Μιχαήλ
Α.Μ. 1072599
up1072599@upnet.gr

Σούρλας Ζήσης
Α.Μ. 1072477
sourlas.zisis@upnet.gr

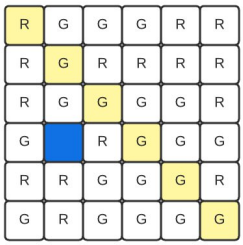
20/04/2023

Θεωρητικό Μέρος

Παρακάτω θα εξετάσουμε όλες τις υποπαρτίδες της συγκεκριμένης εκδοχής του NIM με το πολύ 5 κελιά ελεύθερα. Θα εξετάσουμε τις υποπαρτίδες κατά αύξουσα σειρά ελεύθερων κελιών. Στα παραδείγματα τα κενά κελιά χρωματίζονται με μπλε χρώμα, τα κατειλημμένα με λευκό χρώμα και γράμμα εκτός από αυτά της διαγωνίου που χρωματίζονται με κίτρινο. Σε κάθε περίπτωση υποθέτουμε ότι ο παίχτης που είναι η σειρά του να παίξει είναι ο κόκκινος παίκτης.

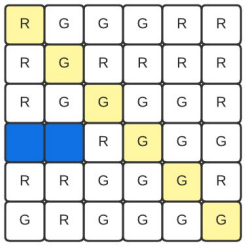
1 κενό κελί

1. Στην περίπτωση που έχουμε μόνο ένα κενό κελί είτε αυτό είναι στην διαγώνιο είτε όχι, τότε για τον κόκκινο παίκτη είναι παρτίδα νίκης καθώς αφού κάνει την κίνηση του δε θα υπάρχει άλλη κίνηση για τον πράσινο παίκτη.

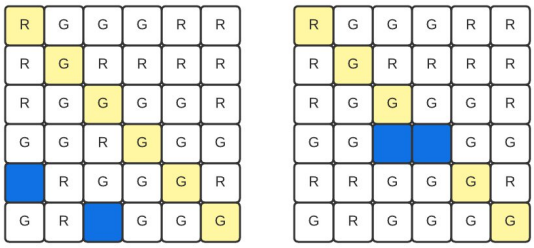


2 κενά κελιά

1. Η πρώτη περίπτωση είναι κανένα από τα δυο κενά κελιά να μην περιέχονται στην διαγώνιο και να μπορούν να συμπληρωθούν με μια κίνηση. Τότε για τον κόκκινο παίκτη είναι παρτίδα νίκης για τον ίδιο λόγο με την προηγούμενη περίπτωση.

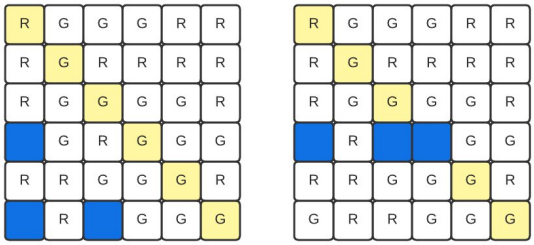


2. Η δεύτερη περίπτωση είναι ένα από τα δυο κελιά να περιέχονται στη διαγώνιο ή να μην είναι διαδοχικά. Σε κάθε περίπτωση δεν μπορούμε να τα συμπληρώσουμε με μια κίνηση. Τότε για τον κόκκινο παίκτη είναι παρτίδα ήττας καθώς όποια κίνηση και να κάνει η υποπαρτίδα που θα δώσει στον αντίπαλο θα είναι αυτή με ένα κελί ελεύθερο δηλαδή μια παρτίδα νίκης.

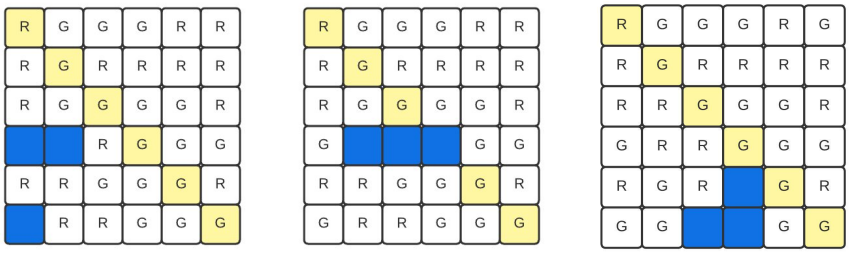


3 κενά κελιά

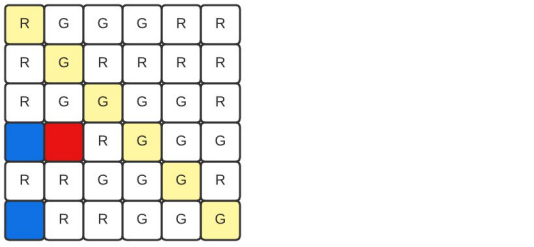
1. Η πρώτη περίπτωση είναι τα κενά κελιά να μπορούν να συμπληρωθούν μόνο κατά μόνας. Για τον κόκκινο παίκτη είναι παιχνίδι νίκης καθώς παίζοντας τυχαία σε ένα από τα κελιά θα δώσει στον αντίπαλο μια παρτίδα ήττας του τύπου 2.2.



2. Η δεύτερη περίπτωση είναι τα δύο κενά κελιά να μπορούν να συμπληρωθούν με μια κίνηση και το τρίτο ξεχωριστά. Για τον κόκκινο παίκτη είναι παρτίδα νίκης. Παίζει μια κίνηση «σπάζοντας» τα δύο κελιά που μπορούν να συμπληρωθούν με μια κίνηση δίνοντας έτσι παρτίδα ήττας στον αντίπαλο (περίπτωση 2.2).

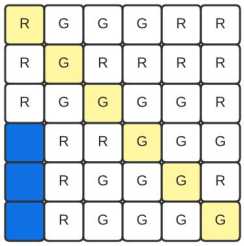


Δηλαδή παίζει :



Από αυτό το σημείο είναι προφανές ότι ό,τι και να παίξει ο παίκτης με τα πράσινα ο κόκκινος θα κερδίσει.

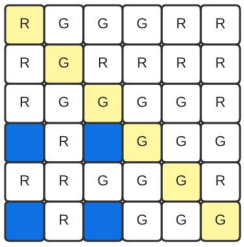
3. Η τελευταία περίπτωση είναι να μπορεί με μία κίνηση να συμπληρώσει και τα τρία κελιά. Τότε ο παίκτης που παίζει τώρα έχει μπροστά του παρτίδα νίκης καθώς μπορεί να συμπληρώσει και τα τρία κελιά με μία κίνηση.



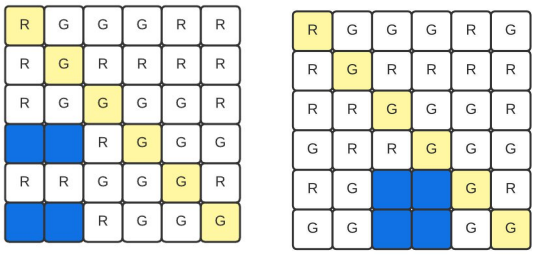
Από την παραπάνω ανάλυση προκύπτει ότι όλα τα παιχνίδια με τρία κενά κελιά είναι παρτίδες νίκης.

4 κελιά κενά

1. Η πρώτη περίπτωση είναι το κάθε κελί να μπορεί να συμπληρωθεί μόνο κατά μόνας. Αυτή η παρτίδα είναι παρτίδα ήττας για τον κόκκινο.(Δίνει αναγκαστικά μια παρτίδα της περίπτωσης 3.1 στον πράσινο που είναι παρτίδα νίκης για αυτόν).

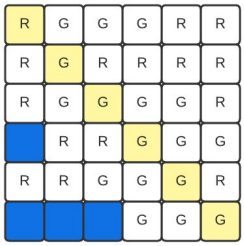


2. Η δεύτερη περίπτωση είναι τα ελεύθερα κελιά να μπορούν ανά δυο να συμπληρώνονται με μια κίνηση. Σε αυτό το σενάριο για τον κόκκινο παίκτη έχουμε παρτίδα ήττας καθώς ό,τι και να κάνει δίνει παιχνίδι νίκης στον πράσινο.



Αναλυτικότερα ο κόκκινος παίκτης έχει την επιλογή να «σπάσει» μια δυάδα ή να συμπληρώσει δύο κελιά. Άρα στον αντίπαλο θα δώσει μια παρτίδα των περιπτώσεων 3.2 ή 2.1 που και οι δύο είναι παρτίδες νίκης για τον πράσινο.

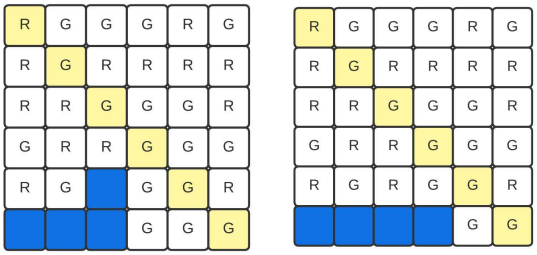
3. Η τρίτη περίπτωση είναι τα τρία ελεύθερα κελιά να μπορούν να συμπληρωθούν ταυτόχρονα και το άλλο ξεχωριστά. Για τον κόκκινο αυτό αποτελεί παρτίδα νίκης.

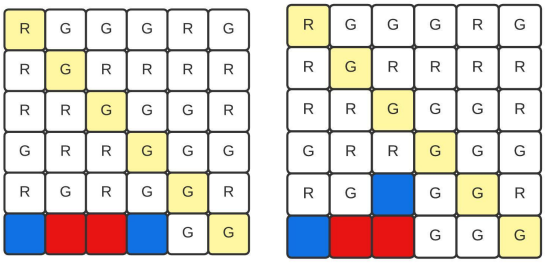


Αναλυτικότερα θα σπάσει την τριάδα συμπληρώνοντας τα δυο κελιά με μια κίνηση και θα δώσει μια παρτίδα της περίπτωσης 2.2 (παρτίδα ήττας) στον πράσινο.

Ειδικές περιπτώσεις:

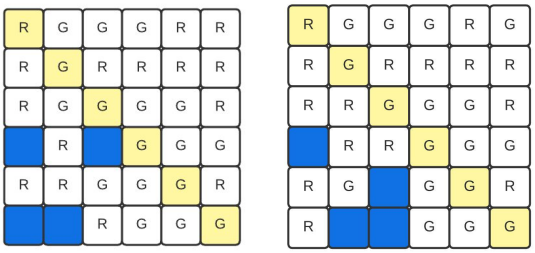
Οι παρακάτω δύο περιπτώσεις μπορούν να αναγνωστούν ως υποπεριπτώσεις της 4.2 (παρτίδα ήττας) ή της 4.3 (παρτίδα νίκης). Ωστόσο αν αναγνωστούν ως η περίπτωση 4.3 και παιχτούν όπως θα δείξουμε είναι παρτίδες νίκης.





Όπως φαίνεται η παρτίδα που προκύπτει για τον πράσινο παίκτη είναι και για τις δύο περιπτώσεις παρτίδα ήττας (περίπτωση 2.2).

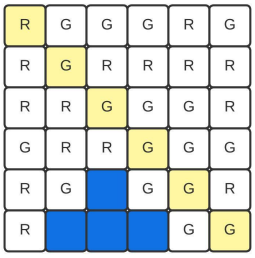
4. Η τελευταία περίπτωση είναι τα δύο ελεύθερα κελιά να μπορούν να συμπληρωθούν ταυτόχρονα και τα άλλα δύο κατά μόνας. Αυτή η παρτίδα είναι παρτίδα νίκης για τον κόκκινο παίκτη.



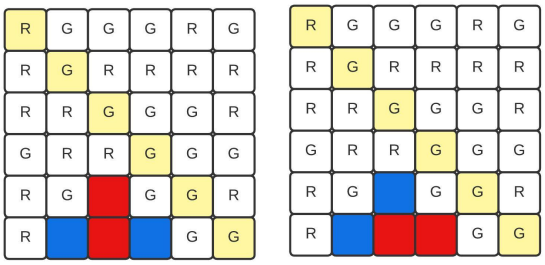
Αναλυτικότερα ο κόκκινος παίκτης θα συμπληρώσει τα δύο κελιά με μια κίνηση και θα αφήσει στον πράσινο παρτίδα ήττας (περίπτωσης 2.2).

Ειδική περίπτωση:

Η παρακάτω περίπτωση μπορεί να αναγνωσθεί ως μια παρτίδα της περίπτωσης 4.3 ή της 4.4. Ανεξαρτήτως όμως του πώς θα αναγνωσθεί είναι παρτίδα νίκης για τον κόκκινο.



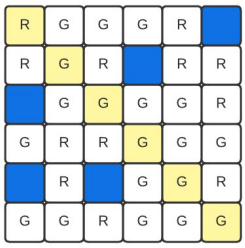
Ο κόκκινος μπορεί να παίξει μία από τις δύο στρατηγικές:



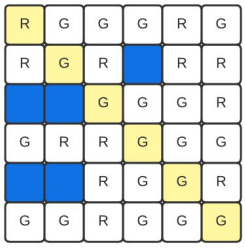
Και στις δύο περιπτώσεις δίνει στον πράσινο μια παρτίδα της περίπτωσης 2.2 δηλαδή παρτίδα ήττας.

5 κελιά κενά

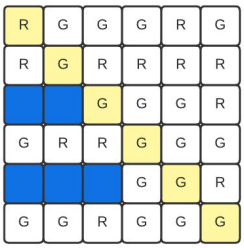
1. Η πρώτη περίπτωση είναι το κάθε ελεύθερο κελί να μπορεί να συμπληρωθεί μόνο κατά μόνας. Για τον κόκκινο παίκτη αποτελεί παρτίδα νίκης γιατί παίζοντας μια οποιαδήποτε κίνηση δίνει στον πράσινο παρτίδα ήττας(περίπτωση 4.1)



2. Η δεύτερη περίπτωση είναι τα τέσσερα κελιά να μπορούν να συμπληρωθούν σε δυάδες και αυτό που έμεινε κατά μόνας. Για τον κόκκινο παίκτη αποτελεί παρτίδα νίκης διότι συμπληρώνοντας το απομονωμένο κελί δίνει παιχνίδι ήττας στον πράσινο(περίπτωση 4.2)

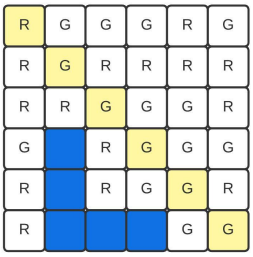


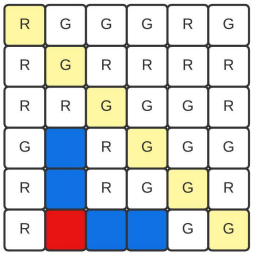
3. Η τρίτη περίπτωση είναι να μπορεί να συμπληρωθεί μια τριάδα και μια δυάδα. Αυτή η παρτίδα είναι παρτίδα νίκης για τον κόκκινο αφού μπορεί να δώσει παρτίδα ήττας στον πράσινο(περίπτωση 4.2) παίζοντας ένα εκ των ακριανών της τριάδας.



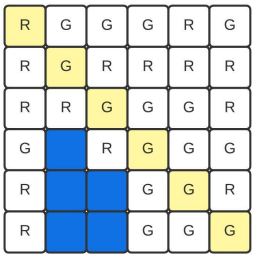
Ειδικές περιπτώσεις:

Η παρακάτω περίπτωση μπορεί είναι υποπερίπτωση της 5.3 στην οποία ωστόσο δεν είναι σαφές ποια είναι η τριάδα και ποια η δυάδα. Σε αυτήν την περίπτωση προκύπτει παιχνίδι ήττας για τον αντίπαλο αν ο κόκκινος παίκτης παίξει το κελί που βρίσκεται σε κάθε περίπτωση στην άκρη της τριάδας.

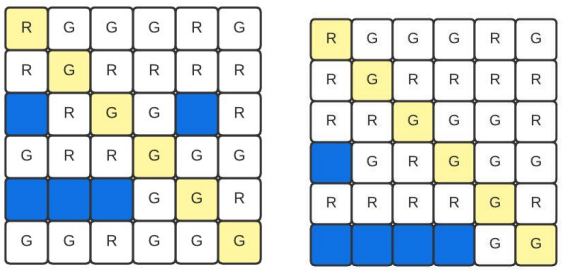




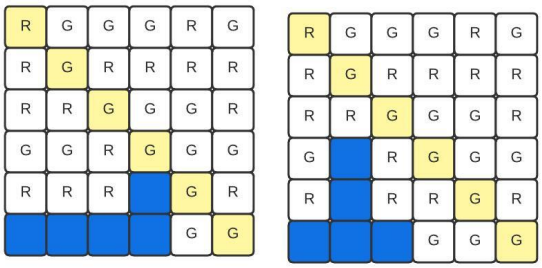
Η παρακάτω περίπτωση μπορεί να αναγνωσθεί ως υποπερίπτωση της 5.2 ή της 5.3. Σε κάθε περίπτωση όμως αποτελεί παρτίδα νίκης για τον κόκκινο.

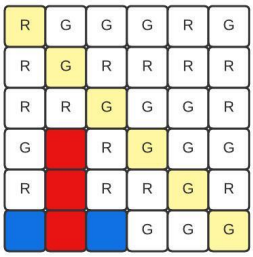


4. Τέταρτη περίπτωση είναι να υπάρχει μια τριάδα και τα υπόλοιπα κελιά να μπορούν να συμπληρωθούν μόνο κατά μόνας. Αυτή είναι παρτίδα νίκης για τον κόκκινο αφού μπορεί να δώσει την παρτίδα της περίπτωσης 2.2 στον πράσινο που είναι παρτίδα ήττας παίζοντας την τρίαδα.

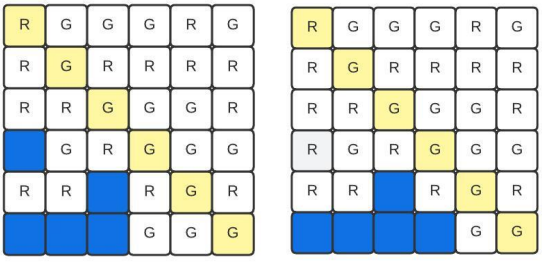


Ειδικές περιπτώσεις:
Οι παρακάτω περιπτώσεις μπορούν να αναγνωστούν είτε ως υποπεριπτώσεις της 5.3 είτε της 5.4. Εάν αναγνωστούν ως υποπεριπτώσεις της 5.3 τότε δεν είναι σίγουρο ότι μπορούν να οδηγήσουν σε παιχνίδι νίκης. Εάν αναγνωστούν ως υποπεριπτώσεις της 5.4 τότε αρκεί απλά να παιχτεί η τριάδα όπως ήδη εξηγήσαμε για να δοθεί παιχνίδι ήττας στον αντίπαλο.

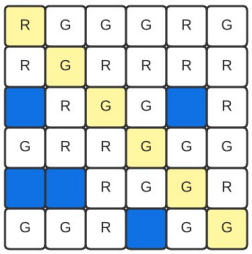




Οι παρακάτω περιπτώσεις μπορούν να αναγνωστούν ως υποπεριπτώσεις της περίπτωσης 5.2 ή της 5.4. Εάν αναγνωστούν ως υποπεριπτώσεις της 5.2 τότε δεν είναι σίγουρο ότι μπορούν να οδηγήσουν σε παιχνίδι νίκης. Εάν αναγνωστούν ως υποπεριπτώσεις της 5.4 τότε αρκεί απλά να παιχτεί η τριάδα όπως ήδη εξηγήσαμε για να δοθεί παιχνίδι ήττας στον αντίπαλο.

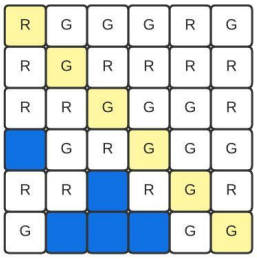


5. Τελευταία περίπτωση για τα πέντε ελεύθερα κελιά είναι να μπορεί να συμπληρωθεί μόνο μια δυάδα και όλα τα υπόλοιπα κελιά κατά μόνας. Αυτή είναι και πάλι παρτίδα νίκης για τον κόκκινο παίκτη καθώς μπορεί να δώσει στον πράσινο την παρτίδα 4.1 η οποία είναι παρτίδα ήττας παίζοντας ένα από τα δύο κελιά της δυάδας.



Ειδική περίπτωση:

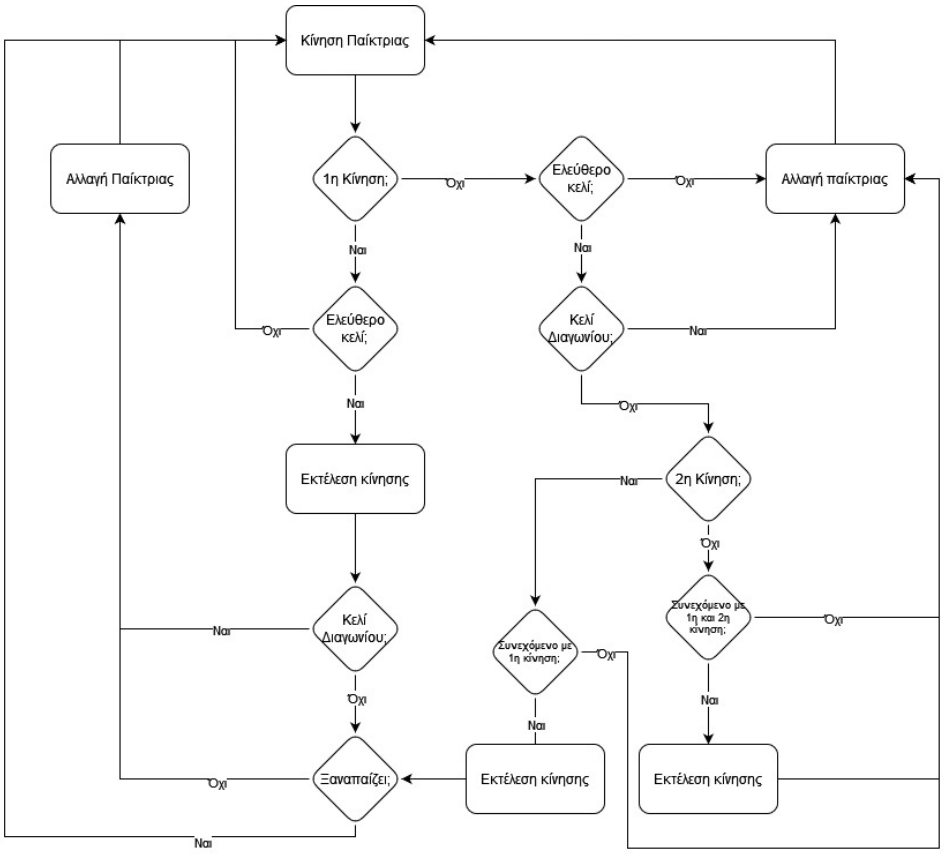
Η παρακάτω περίπτωση μπορεί να αναγνωσθεί ως υποπερίπτωση της περίπτωσης 5.5 ή 5.4. Αν αναγνωσθεί ως υποπερίπτωση της 5.4 αρκεί να παιχτεί η τριάδα για να δοθεί παιχνίδι ήττας στον αντίπαλο. Αν αναγνωσθεί ως υποπερίπτωσης της 5.5 τότε πρέπει να παιχτεί το κελί της δυάδας που συμμετέχει στην τριάδα κατά την άλλη ανάγνωση για να δοθεί το παιχνίδι ήττας 4.1 στον αντίπαλο. Σε κάθε περίπτωση μιλάμε για παιχνίδι νίκης.



Από την παραπάνω ανάλυση συμπεραίνουμε ότι όλα τα παιχνίδια με 5 ελεύθερα κελιά είναι παιχνίδια νίκης.

Προγραμματιστικό Μέρος 1: Υλοποίηση παιχνιδιού

Για την υλοποίηση του απλού παιχνιδιού (άνευ κάποιας στρατηγικής από τη μεριά του υπολογιστή) ακολουθήθηκε το παρακάτω flowchart το οποίο υλοποιήθηκε με τη χρήση των συναρτήσεων που περιγράφονται στη συνέχεια και με μικρή προσθήκη κώδικα στο «κύριο» τμήμα του προγράμματος.



Συναρτήσεις:

play(board, N, player)

Είσοδοι:

board: Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.
N: Προσδιορίζει το μέγεθος του πίνακα.
player: Το γράμμα του παίχτη που παίζει (R ή G).

Έξοδος:

Πίνακας το πολύ τριών θέσεων που περιέχει τις κινήσεις που έπαιξε ο παίκτης (τους αριθμούς των κελιών).

Περιγραφή:

Η συνάρτηση εσωτερικά περιέχει έναν counter κινήσεων ώστε να γνωρίζει το πρόγραμμα ποια κίνηση του παίκτη εκτελείται (1η, 2η ή 3η), έναν πίνακα που διατηρεί τις κινήσεις του παίκτη και ένα flag που ρυθμίζει την εκτέλεση του βρόγχου που αποτελεί το κύριο σώμα της συνάρτησης. Εντός του βρόγχου διαβάζεται από το πληκτρολόγιο η τιμή του κελιού το οποίο επιθυμεί να παίξει ο παίκτης και γίνονται οι απαραίτητοι έλεγχοι εγκυρότητας (αριθμός μεταξύ 1 και N*N). Ανάλογα με την τιμή του μετρητή κινήσεων η συνάρτηση εκτελεί

διαφορετικούς ελέγχους. Για την πρώτη κίνηση καλεί την ***firstMoveChecks*** από την οποία λαμβάνει δύο bolean τιμές, μια για τον αν η κίνηση ήταν έγκυρη και μία για τον αν η κίνηση είναι στη διαγώνιο και πρέπει να εξέλθει από το βρόγχο καθώς ο παίκτης δε μπορεί να ξαναπαίξει. Στην περίπτωση που η κίνηση είναι έγκυρη αλλά όχι στη διαγώνιο, αυξάνεται ο μετρητής κινήσεων και ο παίκτης ρωτάται αν επιθυμεί να ξαναπαίξει καλώντας την ***continuePlayingGame***. Στην περίπτωση που η κίνηση δεν είναι έγκυρη ο βρόγχος ξαναεκτελείται χωρίς να πειραχτεί ο αριθμός κινήσεων ώστε να γίνει η πρώτη κίνηση που είναι απαραίτητη. Όταν πρόκειται για την δεύτερη κίνηση καλείται η **secondMoveChecks** η οποία ελέγχει την εγκυρότητα της κίνησης. Αν είναι έγκυρη τότε ο παίκτης ρωτάται αν επιθυμεί να συνεχίσει και αυξάνεται ο μετρητής κινήσεων ενώ σε διαφορετική περίπτωση το πρόγραμμα εξέρχεται από τον βρόγχο και τελειώνει η εκτέλεση της συνάρτησης. Στην περίπτωση της τρίτης κίνησης καλείται η **thirdMoveChecks** και το πρόγραμμα εξέρχεται από το βρόγχο και η συνάρτηση τερματίζει.

firstMoveChecks(board,N,move,player)

Είσοδοι:

board*:* Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.
N: Προσδιορίζει το μέγεθος του πίνακα.
move: Πίνακας με τις κινήσεις του παίκτη.
player: Το γράμμα του παίχτη που παίζει (R ή G).

Έξοδοι:

Επιστρέφει δύο boolean τιμές. Η πρώτη είναι True/False ανάλογα με το αν η πρώτη από τις κινήσεις που δίνεται ως όρισμα είναι έγκυρη ενώ η δεύτερη ανάλογα με το αν η κίνηση είναι πάνω στη διαγώνιο ή όχι.

Περιγραφή:

Η συνάρτηση αρχικοποιεί τις εξόδους της σε False. Ελέγχει αν η πρώτη θέση του πίνακα κινήσεων (πρώτη κίνηση) αντιστοιχεί σε κενό κελί ή όχι. Αν όχι εκτελεί την κίνηση γράφοντας το γράμμα του παίχτη στην κατάλληλη θέση του πίνακα παιχνιδιού που δίνεται ως όρισμα, αυξάνει την πρώτη θέση του συγκεκριμένου πίνακα (μετρητής συνολικών κινήσεων) κατά 1 και κάνει True την έξοδο εγκυρότητας. Καλώντας την ***getRowAndColumn*** βρίσκει την γραμμή και τη στήλη της κίνησης του παίκτη. Αν αυτές ισούται (κελί διαγωνίου) τότε κάνει True την αντίστοιχη έξοδο.

secondMoveChecks(board,N,move,player)

Είσοδοι*:*

board*:* Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.
N: Προσδιορίζει το μέγεθος του πίνακα.
move: Πίνακας με τις κινήσεις του παίκτη.
player: Το γράμμα του παίχτη που παίζει (R ή G).

Έξοδος:

Επιστρέφει True/False ανάλογα με τον αν η δεύτερη κίνηση του παίκτη είναι έγκυρη ή όχι.

Περιγραφή:

Η συνάρτηση αρχικοποιεί την έξοδό της σε False. Αν η δεύτερη θέση του πίνακα κινήσεων (δεύτερη κίνηση) γίνεται σε ελεύθερο κελί και αν το κελί δεν είναι στη διαγώνιο (βλ. firstMoveChecks για έλεγχο διαγωνίου) καλείται η ***isSequential*** και εφόσον επιστρέψει True, η κίνηση εκτελείται (βλ. firstMoveChecks) και η έξοδος της συνάρτησης γίνεται True.

thirdMoveChecks(board,N,move,player)

Είσοδοι*:*

board*:* Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.
N: Προσδιορίζει το μέγεθος του πίνακα.
move: Πίνακας με τις κινήσεις του παίκτη.
player: Το γράμμα του παίχτη που παίζει (R ή G).

Έξοδος:

Επιστρέφει True/False ανάλογα με τον αν η τρίτη κίνηση του παίκτη είναι έγκυρη ή όχι.

Περιγραφή:

Η συνάρτηση αρχικοποιεί την έξοδό της σε False. Αν η τρίτη θέση του πίνακα κινήσεων (τρίτη κίνηση) γίνεται σε ελεύθερο κελί και αν το κελί δεν είναι στη διαγώνιο (βλ. firstMoveChecks για έλεγχο διαγωνίου) καλείται η ***isSequential2Cells*** και εφόσον επιστρέψει True, η κίνηση εκτελείται (βλ. firstMoveChecks) έξοδος γίνεται True.

Περιγραφή:

Καλώντας την **getRowAndColumn** παίρνει τη γραμμή και τη στήλη στην οποία βρίσκονται οι δύο κινήσεις. Ελέγχει εάν είναι ίδιες οι γραμμές ή οι στήλες. Αν οι κινήσεις βρίσκονται στην γραμμή ελέγχει αν οι στήλες τους διαφέρουν κατά ένα και αν ναι επιστρέφει True αλλιώς False. Αντίστοιχα αν βρίσκονται στην ίδια στήλη. Στην περίπτωση που βρίσκονται και σε διαφορετική γραμμή και σε διαφορετική στήλη επιστρέφει False.

isSequential(prevMove,nextMove, N)

Είσοδοι:

prevMove: Ο αριθμός που αντιστοιχεί στο κελί της πρώτης κίνηση του παίκτη.
nextMove: Ο αριθμός που αντιστοιχεί στο κελί της δεύτερης κίνησης του παίκτη.
N: Προσδιορίζει το μέγεθος του πίνακα.

Έξοδος:

True/False ανάλογα με το αν τα κελιά των ορισμάτων είναι διαδοχικά ή όχι.

Περιγραφή:

Η συνάρτηση καλώντας την **getRowAndColumn** παίρνει τη γραμμή και τη στήλη των δύο κινήσεων που δέχεται ως ορίσματα. Εξετάζει αν οι δύο κινήσεις βρίσκονται στην ίδια γραμμή ή στην ίδια στήλη. Αν βρίσκονται στην ίδια γραμμή εξετάζει αν οι στήλες τους διαφέρουν κατά 1. Αν ναι επιστρέφει True αλλιώς False. Αντίστοιχα αν βρίσκονται στην ίδια στήλη. Αν δε βρίσκονται ούτε στην ίδια γραμμή ούτε στην ίδια στήλη επιστρέφει False.

isSequential2Cells(move1,move2,move3, N)

Είσοδοι:

move1: Ο αριθμός που αντιστοιχεί στο κελί της πρώτης κίνηση του παίκτη.
move2: Ο αριθμός που αντιστοιχεί στο κελί της δεύτερης κίνησης του παίκτη.
move3: Ο αριθμός που αντιστοιχεί στο κελί της τρίτης κίνησης του παίκτη.
N: Προσδιορίζει το μέγεθος του πίνακα.

Έξοδος:

True/False ανάλογα με το αν τα κελιά των ορισμάτων είναι διαδοχικά ή όχι.

Περιγραφή:

Η συνάρτηση αρχικοποιεί την έξοδο της σε False. Καλώντας την **getRowAndColumn** παίρνει τις γραμμές και τις στήλες στις οποίες έχουν γίνει οι κινήσεις. Αφού ο συγκεκριμένος έλεγχος γίνεται για τρεις κινήσεις αυτομάτως θεωρείται ότι οι δύο πρώτες είναι διαδοχικές. Η συνάρτηση ελέγχει εάν οι δύο πρώτες κινήσεις βρίσκονται στην ίδια στήλη ή στην ίδια γραμμή. Αν βρίσκονται στην ίδια γραμμή ελέγχει εάν και η τρίτη κίνηση βρίσκεται στην ίδια γραμμή και αν ναι ελέγχει αν η στήλη της διαφέρει κατά ένα με κάποια από τις στήλες των άλλων δύο κινήσεων. Αν ναι τότε η έξοδος γίνεται True. Αντίστοιχα στην περίπτωση που βρίσκονται στην ίδια στήλη.

Προγραμματιστικό Μέρος 2: Στρατηγικές για τον Υπολογιστή

Για την υλοποίηση των στρατηγικών του υπολογιστή υλοποιήθηκαν οι ζητούμενες συναρτήσεις όπως αυτές περιγράφονται στη συνέχεια καθώς και μερικές βοηθητικές συναρτήσεις.

Συναρτήσεις

getComputerMove_random(board, N, player)

Είσοδοι:

board: Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.
N: Προσδιορίζει το μέγεθος του πίνακα.
player: Το γράμμα του παίχτη που παίζει (R ή G).

Έξοδοι:

Η συνάρτηση δεν επιστρέφει τίποτα.

Περιγραφή:

Αρχικοποιεί τον πίνακα moves[] ο οποίος αρχικά είναι κενός και κρατάει τις ενέργειες του υπολογιστή σε κάθε κίνηση. Έπειτα αρχικοποιεί καλώντας την **getEmptyCells** τον πίνακα emptyCells ο οποίος περιέχει τα κενά κελιά. Εκτελεί έλεγχο και σε περίπτωση που ο πίνακας αυτός είναι κενός ,η μεταβλητή finish γίνεται true και δεν εκτελείται κάτι άλλο. Αλλιώς ο πίνακας move παίρνει στην θέση move[0] μια τυχαία τιμή από τα κενά κελιά. Αρχικοποιείται η μεταβλητή numMove η οποία παίρνει τιμές ανάλογα με το πόσες ενέργειες θέλουμε να πραγματοποιηθούν(0->μια ενέργεια,1->δύο ενέργειες,2->τρεις ενέργειες). Έπειτα καλώντας την **firstmoveRandCheck** εκτελείται η πρώτη ενέργεια. Αν η μεταβλητή numMove είναι 0 ή η finish είναι true η συνάρτηση σταματάει σε αυτήν την ενέργεια. Αλλιώς ξανά ορίζει τον πίνακα emptyCells και γίνεται έλεγχος μέσω μιας τυχαίας μεταβλητής colOrRow για το αν η δεύτερη ενέργεια γίνει σε ίδια στήλη η γραμμή. Έτσι για κάθε περίπτωση δημιουργείται ένας πίνακας sameColEmpty ή sameRowEmpty και καλώντας την **secondmoveRandCheck** (με όρισμα έναν από τους δύο παραπάνω πίνακες στο αντίστοιχο πεδίο) πραγματοποιεί την δεύτερη ενέργεια. Αντίστοιχα γίνεται ο έλεγχος για την μεταβλητή numMove και finish. Σε περίπτωση που περάσει και αυτόν τον έλεγχο πραγματοποιείται η τρίτη ενέργεια αφού ξανά οριστεί ο πίνακας των κενών κελιών και έπειτα ο πίνακας sameColEmpty ή SameRowEmpty αναλόγως με την τυχαία μεταβλητή που ορίστηκε πριν την εκτέλεση της δεύτερης ενέργειας, καλώντας την **thirdmoveRandCheck** και ολοκληρώνεται η εκτέλεσή της.

getComputerMove_firstfit(board, N, player, moves=0)

Είσοδοι:

board: Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.
N: Προσδιορίζει το μέγεθος του πίνακα.
player: Το γράμμα του παίχτη που παίζει (R ή G).
moves: Ο μέγιστος αριθμός κινήσεων που μπορεί να γίνουν. Προαιρετική είσοδος. Προεπιλεγμένη τιμή 0.

Έξοδοι:

Η συνάρτηση δεν επιστρέφει τίποτα.

Περιγραφή:

Αν λάβει τιμή 0 στο όρισμα moves (προεπιλογή), επιλέγει τυχαία πόσες κινήσεις θα κάνει. Ακολουθεί την ίδια διαδικασία με την **getComputerMove_random** με μόνη διαφορά ότι κατά την επιλογή κελιού από τον πίνακα κενών κελιών δεν επιλέγει τυχαία αλλά επιλέγει το πρώτο κελί του πίνακα.

getComputerMove_copycat(board, N, player, opponentMove)

Είσοδοι:

board: Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.
N: Προσδιορίζει το μέγεθος του πίνακα.
player: Το γράμμα του παίχτη που παίζει (R ή G).
opponentMove: Πίνακας με το πολύ τρεις τιμές που αναπαριστούν τις κινήσεις του αντιπάλου.

Έξοδοι:

Η συνάρτηση δεν επιστρέφει τίποτα.

Περιγραφή:

Η συνάρτηση αρχικά ελέγχει αν ο πίνακας opponentMove είναι κενός. Εάν ισχύει κάτι τέτοιο τότε ο υπολογιστής κάνει την πρώτη κίνηση στο παιχνίδι και κατά συνέπεια δεν υπάρχει κίνηση του αντιπάλου για να παίξει τη συμμετρική της. Σε αυτή τη περίπτωση καλεί την **getComputerMove_random** και παίζει τυχαία. Σε διαφορετική περίπτωση ακολουθεί μια σειρά από ελέγχους.

Πρώτον ελέγχει αν το κελί της πρώτης κίνησης του αντίπαλου βρίσκεται στη διαγώνιο. Αν ναι τότε υπολογίζει τη γραμμή και τη στήλη που πρέπει να παίξει βάσει της εκφώνησης (N-k+1) και καλώντας την **getMove** παίρνει το κελί στο οποίο πρέπει να παίξει. Εξετάζει αν το κελί είναι ελεύθερο και αν είναι εκτελεί την κίνηση διαφορετικά διατρέχει όλη τη διαγώνιο και

αποθηκεύει σε ένα πίνακα τα κενά κελιά της. Αν ο πίνακας είναι κενός τότε καλώντας την **getEmptyCells** παίρνει όλα τα κενά κελιά του παιχνιδιού και παίζει στο πρώτο. Αν ο πίνακας δεν είναι κενός τότε επιλέγει τυχαία ένα κελί από αυτόν και παίζει σε αυτό.

Αν η κίνηση του αντιπάλου δεν είναι στη διαγώνιο τότε διατρέχει όλο τον πίνακα με τις κινήσεις του αντιπάλου και για κάθε μία λαμβάνει καλώντας την **getRowAndColumn** την γραμμή και τη στήλη της, χρησιμοποιώντας τες αντεστραμμένες για να υπολογίσει καλώντας την **getMove** την κίνηση που ιδανικά πρέπει να κάνει. Για την κίνηση που υπολόγισε ελέγχει αν γίνεται σε ελεύθερο κελί (καθώς αυτό είναι το μόνο πιθανό κώλυμα) και την αποθηκεύει σε ένα πίνακα. Εάν κάποια από τις υποψήφιες κινήσεις γίνεται σε μη κενό κελί τότε καλεί την **getComputerMove_firstfir** και παίζει αναλόγως. Εάν είναι όλες σε ελεύθερα κελιά τότε εκτελεί αυτές τις κινήσεις.

getEmptyCells(board,N)

Είσοδοι:

board*:* Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.
N: Προσδιορίζει το μέγεθος του πίνακα.

Έξοδος:

Πίνακας με τους αριθμούς των κενών κελιών του παιχνιδιού.

Περιγραφή:

Διατρέχει τον πίνακα που δέχεται ως όρισμα και αποθηκεύει τους αριθμούς των κενών κελιών του σε έναν καινούργιο πίνακα τον οποία και επιστρέφει.

getEmptyCellNum(board,N)

Είσοδοι:

board*:* Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.
N: Προσδιορίζει το μέγεθος του πίνακα.

Έξοδος:

Το πλήθος των κενών κελιών.

Περιγραφή:

Αφαιρεί από το συνολικό πλήθος των κελιών του παιχνιδιού (N*N) τον αριθμό των κινήσεων που έχουν γίνει (πρώτο κελί του πίνακα παιχνιδιού).

getMove(row, column, N)

Είσοδοι:

row: Μια γραμμή του πίνακα παιχνιδιού.

column: Μια στήλη του πίνακα παιχνιδιού.
N: Προσδιορίζει το μέγεθος του πίνακα.

Έξοδος:

Ο αριθμός του κελιού που αντιστοιχεί στη συγκεκριμένη γραμμή και στήλη.

Περιγραφή:

Υπολογίζει τον αριθμό του κελιού ως εξής:

$$move = N * (row - 1) + column$$

firstMoveRandChecks(board, N,move,player)

Είσοδοι:

board: Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.
N: Προσδιορίζει το μέγεθος του πίνακα.
move: Πίνακας με τις ενέργειες του παίκτη σε κάθε κίνηση.
player: Το γράμμα του παίκτη που παίζει

Έξοδος:

Επιστρέφει μια Boolean μεταβλητή endflag η οποία αν γίνει true θα σημαίνει ότι το κελί που διάλεξε ο υπολογιστής βρίσκεται πάνω στη διαγώνιο και θα πρέπει να ολοκληρώσει την κίνηση του δίνοντας σειρά αυτόματα στον αντίπαλο.

Περιγραφή:

Παίζει την κίνηση με το γράμμα που του αντιστοιχεί (G ή R) στο κελί του πίνακα board[**move[0]**],αυξάνει τον counter των κινήσεων(board[0]) και ελέγχει αν αυτό το κελί βρίσκεται στην διαγώνιο ώστε να κάνει true την μεταβλητή endflag.

secondMoveRandChecks(emptyCells,move,board,player)

Είσοδοι:

emptyCells: Πίνακας που περιέχει τους αριθμούς των κενών κελιών της παρτίδας.
move: Πίνακας με τις ενέργειες του παίκτη σε κάθε κίνηση.
board: Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.
player: Το γράμμα του παίκτη που παίζει

Έξοδος:

Επιστρέφει μια Boolean μεταβλητή finish η οποία αν γίνει true θα σημαίνει ότι το κελί που διάλεξε ο υπολογιστής βρίσκεται πάνω στη διαγώνιο και θα πρέπει να ολοκληρώσει την κίνηση του δίνοντας σειρά αυτόματα στον αντίπαλο.

Περιγραφή:

Για όλο το εύρος του πίνακα που έχει αποθηκευμένα κελιά ελέγχει αρχικά να μην είναι στη διαγώνιο και μετά αν είναι διαδοχικά με το κελί move[0], την πρώτη ενέργεια της κίνησης(ο έλεγχος αυτός γίνεται καλώντας την **isSequential**).Αν ισχύουν αυτές οι δύο προϋποθέσεις εκτελεί κίνηση στο συγκεκριμένο

κελί και σταματάει. Αλλιώς κάνει true την μεταβλητή finish και σταματάει την κίνηση.

thirdMoveRandChecks(emptyCells,move,board,player)

Είσοδοι:

emptyCells: Πίνακας που περιέχει τους αριθμούς των κενών κελιών της παρτίδας.
move: Πίνακας με τις ενέργειες του παίκτη σε κάθε κίνηση.
board: Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.
player: Το γράμμα του παίκτη που παίζει

Έξοδος:

Επιστρέφει μια Boolean μεταβλητή finish η οποία αν γίνει true θα σημαίνει ότι το κελί που διάλεξε ο υπολογιστής βρίσκεται πάνω στη διαγώνιο και θα πρέπει να ολοκληρώσει την κίνηση του δίνοντας σειρά αυτόματα στον αντίπαλο.

getComputerMove_winmove(board,N,player)

Είσοδοι:

board: Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.
N: Προσδιορίζει το μέγεθος του πίνακα.
move: Πίνακας με τις ενέργειες του παίκτη σε κάθε κίνηση.
player: Το γράμμα του παίκτη που παίζει.

Έξοδος:

Η συνάρτηση δεν επιστρέφει τίποτα.

Περιγραφή:

Η συγκεκριμένη συνάρτηση καλείται όταν στο παιχνίδι έχουν απομείνει το πολύ 5 κελιά και είναι η σειρά του υπολογιστή να παίξει. Σκοπός είναι να εκτελέσει την κίνηση (αν υπάρχει) που θα οδηγήσει σε σίγουρη ήττα τον αντίπαλο.

Αρχικά η συνάρτηση καλεί την **getEmptyCells** λαμβάνοντας μια τη λίστα των κενών κελιών του παιχνιδιού και την **getEmptyCellNum** λαμβάνοντας τον αριθμό των κενών κελιών. Ανάλογα με αυτό τον αριθμό εκτελεί διαφορετικές ενέργειες.

Στην περίπτωση που το ελεύθερο κελί είναι ένα τότε απλώς καλεί την **getComputerMove_random** και παίζει “τυχαία” το μοναδικό κελί που έχει απομείνει κερδίζοντας την παρτίδα.

Στην περίπτωση των δύο κενών κελιών καλεί την **winMoveCheck2Cells** ώστε να ελέγξει αν πρόκειται για παρτίδα ήττας ή νίκης. Από το θεωρητικό κομμάτι γνωρίζουμε ότι παρτίδας νίκης με δύο κενά κελιά υπάρχει μόνο όταν αυτά είναι συνεχόμενα και όχι στη διαγώνιο. Συνεπώς όταν αναγνωρίζει παρτίδα νίκης η συνάρτηση παίζει τα δύο αυτά κελιά. Αν αναγνωρίσει παρτίδα ήττας καλεί την **getComputerMove_random** και παίζει τυχαία.

Στην περίπτωση των τριών κενών κελιών δεν χρειάζεται να γίνει κάποιος έλεγχος για το αν πρόκειται για παρτίδα ήττας καθώς όπως προκύπτει από το θεωρητικό μέρος οι παρτίδες με 3 κενά κελιά είναι πάντα παρτίδες νίκης. Επιπλέον αφού η παρτίδα του ενός κενού κελιού είναι παρτίδα νίκης δε πρόκειται να παίξει ποτέ δύο συνεχόμενα κενά κελιά καθώς αυτά θα οδηγούσαν σε παρτίδα νίκης για τον αντίπαλο. Οι μόνες περιπτώσεις που μπορεί να οδηγήσουν σε παρτίδα ήττας για τον αντίπαλο είναι να παίξει και τα τρία κελιά μαζί εάν είναι συνεχόμενα ή να παίξει μόνο ένα κελί. Ωστόσο ακόμα και η περίπτωση των τριών συνεχόμενων κελιών μπορεί να παιχτεί με ένα κελί και να οδηγήσει σε παρτίδα ήττας τον αντίπαλο, παίζοντας το μεσαίο κελί. Συνεπώς η συνάρτηση κάνει τα εξής: δημιουργεί ένα αντίγραφο του πίνακα παιχνιδιού και για κάθε κελί στη λίστα τον κενών κελιών εκτελεί την κίνηση σε αυτό το κελί στον προσωρινό πίνακα. Έπειτα καλεί την **winMoveCheck2Cells** ώστε να ελέγξει αν η παρτίδα που προκύπτει είναι παρτίδα ήττας. Αν είναι τότε εκτελεί την κίνηση στον κανονικό πίνακα και παύει να εξετάζει τα υπόλοιπα κενά κελιά.

Στην περίπτωση των 4 κενών κελιών καλείται η **winMoveCheck4Cells** η οποία ελέγχει αν πρόκειται για παρτίδα ήττας ή νίκης και εκτελεί την κίνηση στη δεύτερη περίπτωση. Αν επιστρέψει false που σημαίνει ότι δεν εκτελέστηκε κάποια κίνηση (παρτίδα ήττας) καλείται η **getComputerMove_random** και παίζει τυχαία.

Στην περίπτωση των 5 κενών κελιών, όπως προκύπτει από το θεωρητικό μέρος, δεν υπάρχει παρτίδα ήττας συνεπώς δεν χρειάζεται κάποιος τέτοιος έλεγχος. Από τη στιγμή που τα παιχνίδια 3 κελιών είναι πάντα παιχνίδια νίκης, δεν υπάρχει επίσης περίπτωσης να γίνει κίνηση με 2 κελιά και να οδηγήσει σε παρτίδα ήττας τον αντίπαλο. Συνεπώς η συνάρτηση εξετάζει δύο περιπτώσεις. Πρώτον το να παίξει 3 συνεχόμενα κελιά και δεύτερον το να παίξει 1 κελί. Για την πρώτη περίπτωση διατρέχει όλους τους πιθανούς συνδυασμούς των κενών κελιών ψάχνοντας για τις πιθανές τριάδες συνεχόμενων και όχι επί της

διαγωνίου κελιών. Αν βρει κάποια τέτοια τριάδα εκτελεί την αντίστοιχη κίνηση σε ένα προσωρινό πίνακα και καλώντας την **winMoveCheck2Cells** ελέγχει αν η παρτίδα που προκύπτει είναι παρτίδα ήττας. Αν αυτό ισχύει τότε εκτελεί την κίνηση στον κανονικό πίνακα παιχνιδιού και σταματά την εκτέλεση της συνάρτησης. Αν δε βρει κάποια κατάλληλη τριάδα ελέγχει τη δεύτερη περίπτωση. Για κάθε κελί των κενών κελιών εκτελεί την κίνηση σε ένα προσωρινό πίνακα παιχνιδιού και καλώντας την **winMoveCheck4Cells** ελέγχει αν αυτή οδηγεί σε παιχνίδι ήττας τον αντίπαλο. Όταν βρει μια τέτοια περίπτωση εκτελεί την κίνηση στον κανονικό πίνακα παιχνιδιού και τερματίζει την εκτέλεση της συνάρτησης.

winMoveCheck2Cells(board, N)

Είσοδοι:

board: Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.

N: Προσδιορίζει το μέγεθος του πίνακα.

Έξοδος:

Η συνάρτηση επιστρέφει True/False ανάλογα με το αν η παρτίδα που αναπαριστά ο πίνακας board είναι παρτίδα νίκης ή όχι.

Περιγραφή:

Η συνάρτηση βρίσκει τα κενά κελιά καλώντας την **getEmptyCells.** Έπειτα ελέγχει αν τα δύο κενά κελιά είναι διαδοχικά και δεν βρίσκονται στη διαγώνιο. Αν ισχύει κάτι τέτοιο επιστρέφει True αλλιώς False.

winMoveCheck4Cells(board, N,player)

Είσοδοι:

board: Πίνακας ο οποίος αναπαριστά την τρέχουσα κατάσταση της παρτίδας.

N: Προσδιορίζει το μέγεθος του πίνακα.

player: Το γράμμα του παίκτη που παίζει.

Έξοδος:

Η συνάρτηση επιστρέφει True/False ανάλογα με το αν η παρτίδα που αναπαριστά ο πίνακας board είναι παρτίδα νίκης ή όχι.

Περιγραφή:

Όπως προκύπτει από το θεωρητικό μέρος οι παρτίδες με 3 κελιά και 1 κελί ελεύθερο είναι πάντα παρτίδες νίκης. Συνεπώς η συνάρτηση αναγκαστικά θα δοκιμάσει μόνο κινήσεις με 2 κελιά οι οποίες πιθανόν να οδηγήσουν σε παρτίδα ήττας. Η συνάρτηση βρίσκει τα κενά κελιά καλώντας την **getEmptyCells.** Έπειτα για κάθε πιθανό ζεύγος κενών κελιών ελέγχει αν και τα δύο κελιά δεν βρίσκονται στη διαγώνιο και αν είναι διαδοχικά. Αν ισχύει κάτι τέτοιο εκτελεί μια κίνηση με αυτά τα δύο κελιά σε ένα προσωρινό αντίγραφο του board και καλώντας την **winMoveCheck2Cells** ελέγχει αν το παιχνίδι που προκύπτει

είναι παρτίδα ήττας. Αν είναι τότε εκτελεί την κίνηση στον πίνακα board και σταματάει την εκτέλεσή της επιστρέφοντας True. Αν εν τέλει δεν εκτελέσει κάποια κίνηση η συνάρτηση επιστρέφει False πράγμα που σημαίνει ότι η παρτίδα που της δόθηκε είναι παρτίδα ήττας.