Ορισμός Κατάστασης Παιχνιδιού

Η κατάσταση του παιχνιδιού περιγράφεται από την τρέχουσα διάταξη των χαρακτήρων σε έναν πίνακα 3x3. Κάθε θέση στον πίνακα μπορεί να περιέχει έναν από τους χαρακτήρες 'C', 'S', 'E' ή να είναι άδεια (συμβολίζεται ως '-'). Η κατάσταση αυτή αλλάζει μετά από κάθε κίνηση των παικτών. Η αρχική κατάσταση του πίνακα ξεκινά με όλες τις θέσεις άδειες εκτός από μία τυχαία θέση στη μεσαία σειρά που περιέχει τον χαρακτήρα 'S'.

Ορισμός Αξίας Τελικών Καταστάσεων

Η αξία των τελικών καταστάσεων στο παιχνίδι καθορίζεται από την ύπαρξη νικηφόρων ακολουθιών χαρακτήρων ή από την πλήρη κάλυψη του πίνακα με χαρακτήρες χωρίς να υπάρξει νικητής:

- **Νικηφόρα Κατάσταση**: Όταν δημιουργηθεί μια ακολουθία 'CSE' ή 'ESC' οριζόντια, κάθετα ή διαγώνια, ο παίκτης που ολοκλήρωσε την ακολουθία κερδίζει το παιχνίδι. Αυτό το γεγονός αντιπροσωπεύεται από μια θετική αξία, (+10), καθώς σημαίνει άμεση νίκη.
- **Ισοπαλία**: Εάν όλα τα κελιά του πίνακα είναι γεμάτα και δεν έχει σχηματιστεί καμία νικηφόρα ακολουθία, το παιχνίδι καταλήγει σε ισοπαλία. Αυτή η κατάσταση έχει μηδενική αξία (0).
- **Μεταβατικές Καταστάσεις**: Καταστάσεις που δεν καθορίζουν άμεσα το αποτέλεσμα του παιχνιδιού αλλά χρησιμεύουν για την ανάλυση πιθανών μελλοντικών κινήσεων και στρατηγικών. Η αξία τους καθορίζεται βάσει της πιθανότητας νίκης ή ήττας στο μέλλον, χρησιμοποιώντας τον αλγόριθμο Minimax.

Περιγραφή Κώδικα

- **Main Components**:
    - **main()**: Η μέθοδος που ξεκινά το παιχνίδι, εναλλάσσει τις κινήσεις των παικτών και ελέγχει για τερματισμό του παιχνιδιού με νίκη ή ισοπαλία.
    - **initializeBoard()**: Αρχικοποιεί τον πίνακα του παιχνιδιού.
    - **printBoard()**: Εκτυπώνει την τρέχουσα κατάσταση του πίνακα.
- **Λογική Παιχνιδιού**:
    - **findBestMove()**: Υπολογίζει την καλύτερη δυνατή κίνηση για τον MAX παίκτη χρησιμοποιώντας τον αλγόριθμο Minimax. Επιλέγει την κίνηση που μεγιστοποιεί το αποτέλεσμα βάσει της εκτίμησης των πιθανών καταστάσεων του παιχνιδιού.
    - **selectBestCharForPosition(int row, int col):** Αυτή η μέθοδος υπολογίζει και επιλέγει τον καλύτερο χαρακτήρα για να τοποθετηθεί σε μια δεδομένη θέση του πίνακα για τον παίκτη MAX. Εξετάζει τους διαθέσιμους χαρακτήρες και επιλέγει αυτόν που προσφέρει την υψηλότερη αξία αξιολόγησης βάσει της μεθόδου evaluate().

    - **selectBestCharForMin(int row, int col):** Αυτή η μέθοδος υπολογίζει και επιλέγει τον καλύτερο χαρακτήρα για τον παίκτη MIN σε μια δεδομένη θέση. Αναλύει κάθε δυνατή τοποθέτηση χαρακτήρα και επιλέγει εκείνον που προσφέρει τη χαμηλότερη αξία αξιολόγησης, επιδιώκοντας να μειώσει τις πιθανότητες νίκης του αντιπάλου.
    - **checkWin()**: Ελέγχει αν υπάρχει κερδισμένη ακολουθία στον πίνακα.
    - **checkLines()**: Βοηθητική μέθοδος που ελέγχει όλες τις δυνατές γραμμές για κερδισμένη ακολουθία.
- **Αλγόριθμος Minimax**:
    - **minimax()**: Υλοποιεί τον αλγόριθμο Minimax για να υπολογίζει την βέλτιστη στρατηγική, αξιολογώντας με τη μέθοδο της αναδρομής, πιθανές κινήσεις και τα αποτελέσματά τους.
- **Αλληλεπίδραση Παικτών**:
    - **playerMove()**: Διαχειρίζεται τις κινήσεις του ανθρώπινου παίκτη, εξασφαλίζοντας την ορθότητα των επιλογών του.
    - **isValidMove(int x, int y, char playerChar):** Ελέγχει αν η κίνηση που επιλέχθηκε από τον παίκτη είναι έγκυρη, δηλαδή αν η θέση είναι ελεύθερη και ο χαρακτήρας είναι ένας από τους 'C', 'S', 'E'.