**Colegiul Național “Decebal” Deva**

**ALGORITM DE A.I. X ȘI 0**

**LUCRARE DE ATESTAT LA INFORMATICĂ**

**Elev**: **Profesor coordonator:**

Șichet Paul – Cristian Faur Haynal

Clasa a XII-a A

2021

**Argument**

Inteligența artificială este probabil domeniul informaticii care s-a dezvoltat cel mai mult în ultima perioadă. Aplicațiile sale încep să fie prezente în tot mai multe împrejurări, de la mașini cu pilot automat, algoritmi de recunoaștere facială, jocuri video, la roboți care reușesc să reproducă cu fidelitate gestica umană sau chiar verdicte medicale pe baza analizelor unor poze mai eficiente decât ale specialiștilor.

Am ales această temă deoarece este un domeniu în care mi-ar plăcea să lucrez în viitor, și totodată pentru a profita de oportunitatea de a pune în practică cunoștințele de informatică dobândite pe parcursul liceului. În plus, am reușit să aflu mai multe despre ce înseamnă concret dezvoltarea de jocuri video, și cum trebuie un programator să abordeze un astfel de proiect.

Programul a fost creat în JavaScript, și este rulat în editorul p5js, o aplicație specializată pentru grafică și animații.

**Algoritmul Minimax**

**Minimax** este un algoritm recursiv care generează toate situațiile posibile de pe tabla de joc, alegând la fiecare pas mutarea care maximizează șansa de câștig a calculatorului. În Minimax cei 2 jucători se numesc minimizator și maximizator. Minimizatorul încearcă să obțină cel mai mic scor posibil, iar maximizatorul cel mai mare scor posibil. Fiecare stare a tablei de joc are asociată o valoare numerică. Într-o anumită stare, dacă maximizatorul este în avantaj, valoarea tinde să fie pozitivă. Invers, dacă minimizatorul este avantaj, valoarea tinde să fie negativă. Valorile tablei sunt calculate printr-o metodă specifică fiecărui tip de joc.

**Arborele de joc și parcurgerea sa**

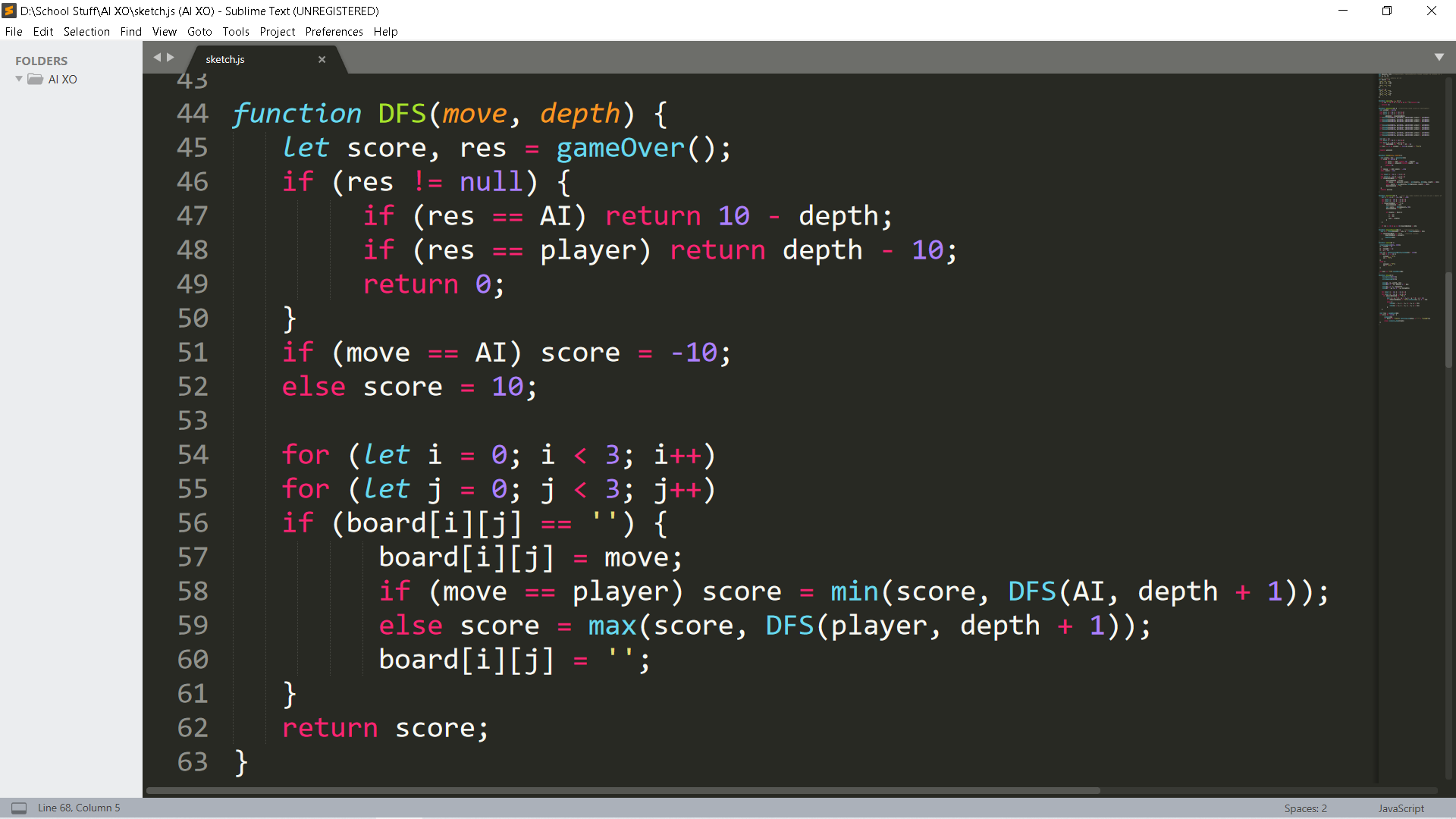
Algoritmul generează un arbore „de joc”, în care nodurile reprezintă stările tablei, iar fiii unui nod toate posibilitățile de a muta ale jucătorului aflat în momentul respectiv la mutare. Dintre aceste mutări, se alege nodul cu valoare maximă dacă AI-ul este la mutare, respectiv valoarea minimă dacă jucătorul mută. Valoarea unui nod se stabilește astfel:

1) Dacă nodul este frunză, înseamnă că jocul a luat sfârșit și valoarea sa este valoarea asociată rezultatului obținut.

2) Dacă nodul nu este frunză, atunci valoarea sa se calculează prin programare dinamică pe arbore, fiind maximul/minimul (după cum cere situația) dintre toate frunzele din subarborele său.

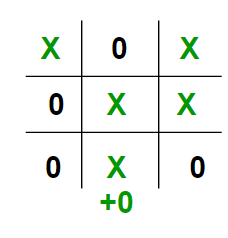
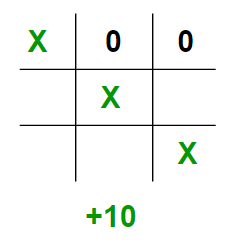
Este important de menționat că aceste mutări sunt fictive, nu se petrec în realitate pe tabla de joc, sunt o simulare a calculatorului care evaluează toate scenariile posibile, pornind de la ce se află la momentul actual pe tabla de joc(starea curentă a tablei reprezintă de fapt rădăcina arborelui).

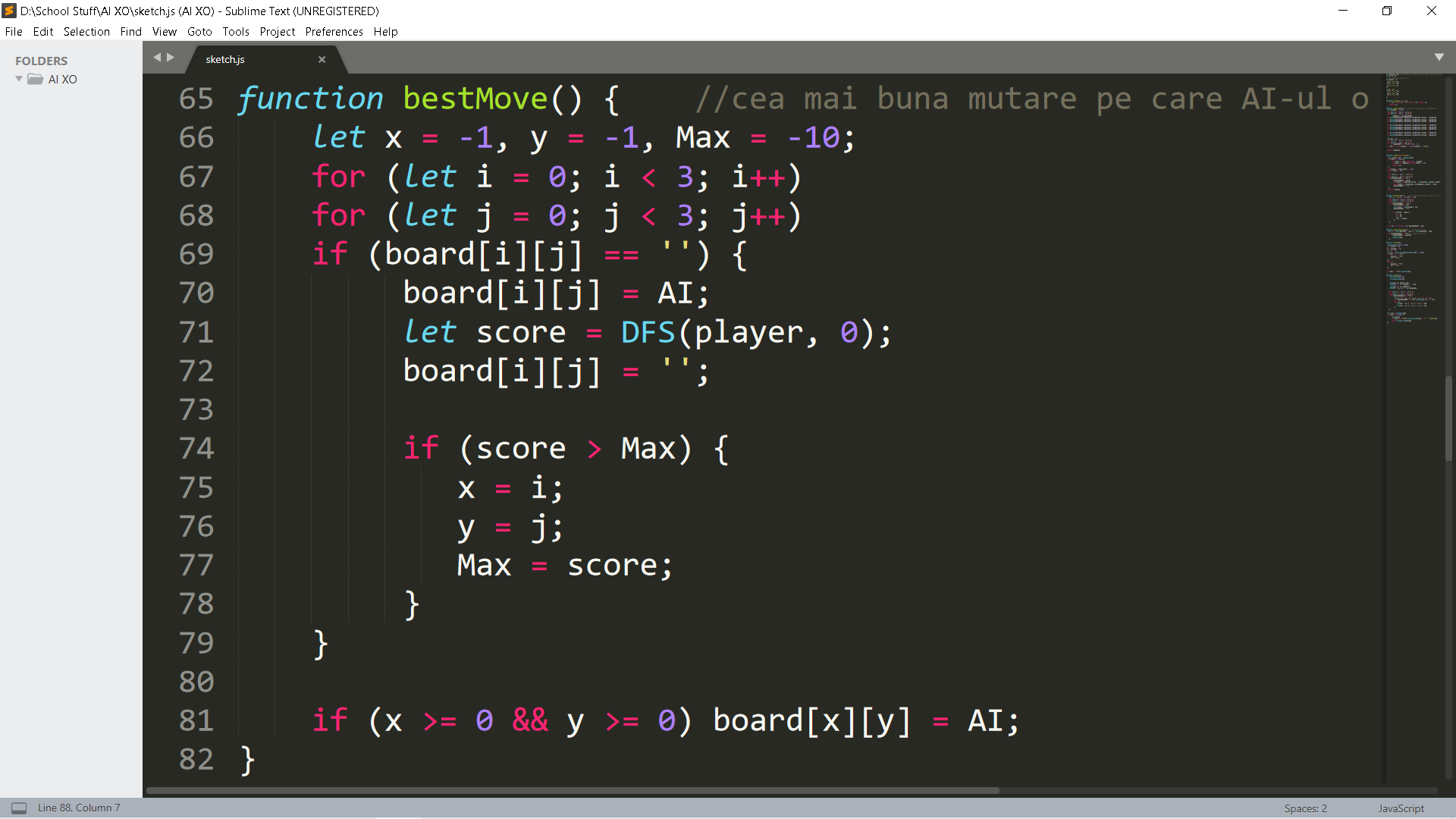
Parcurgerea DFS pe arborele de joc pentru a calcula scorul cel mai bun



**Funcția de evaluare**

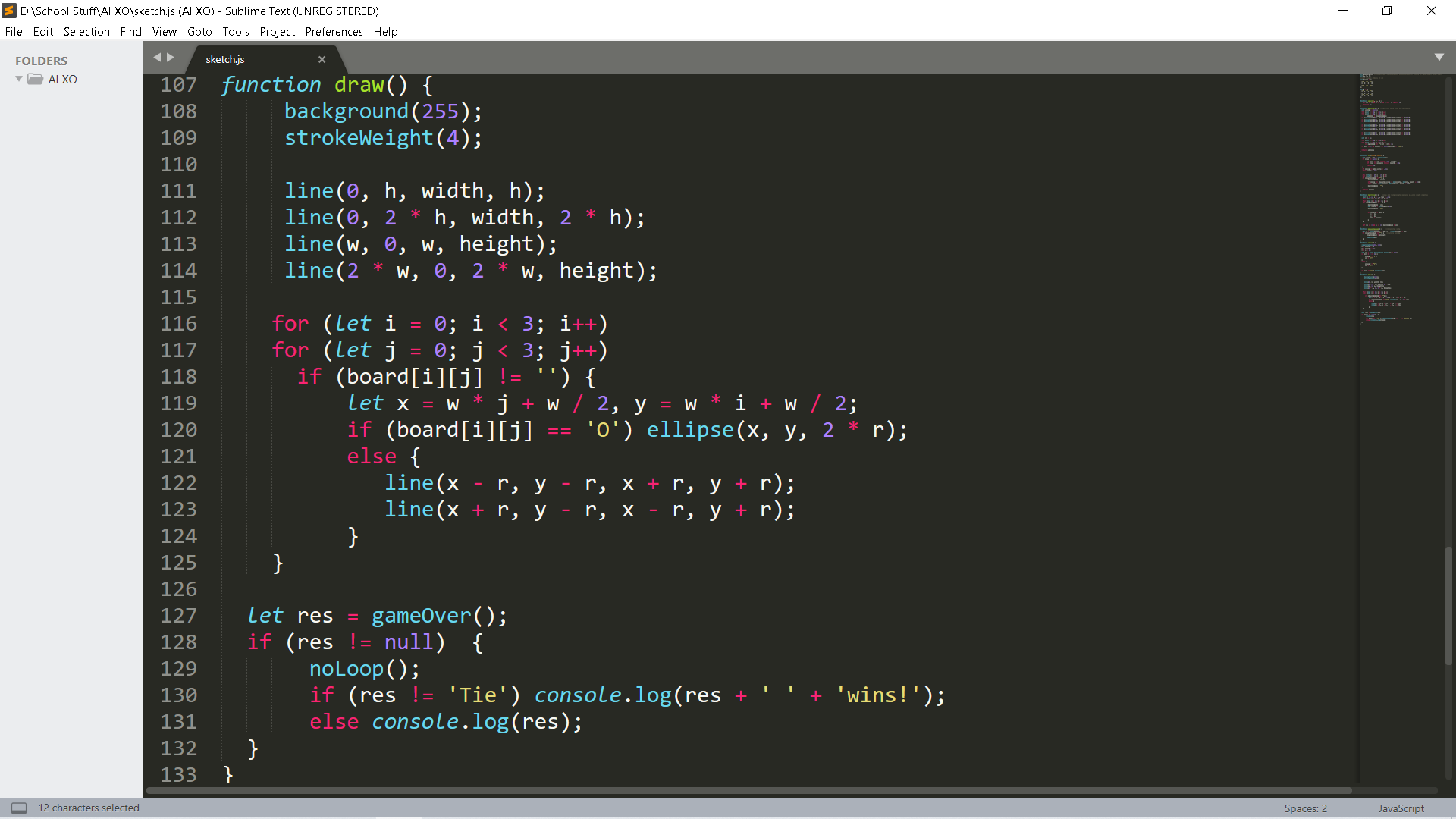
Pentru simplitate, am ales ca victoria jucătorului să aibă valoarea -10, remiza 0, iar victoria AI-ului 10. Calculatorul este pe post de maximizator, iar jucătorul minimizator. Funcția de evaluare calculează pentru fiecare mutare posibilă pe care o poate efectua AI-ul(are loc pe tabla de joc), prin apelarea funcției DFS, scorul acestei mutări. Se realizează maximul dintre aceste valori și pe ecran este marcată mutarea calculatorului.

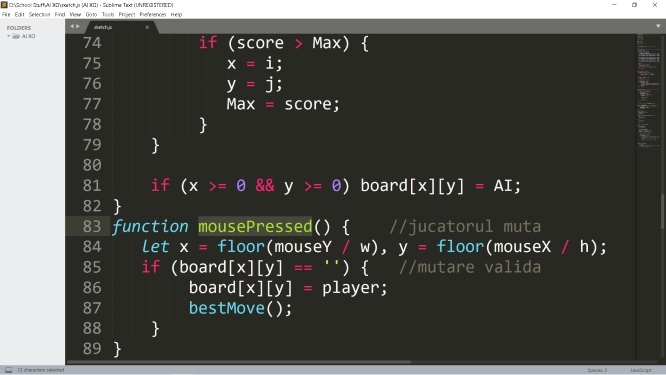




**Bucla de joc și interacțiunea cu jucătorul**

Bucla de joc este reprezentată de funcția draw, funcție principală a editorului p5js, folosit pentru a desena tabla de joc, care are sfârșitul jocului drept condiție de ieșire. Interacțiunea cu jucătorul este realizată prin intermediul funcției mousePressed(de asemnea, funcție specifică editorului; nu are nevoie de apelare, este apelată automat în momentul apăsării mouse-ului). În cadrul acestei funcții este reprezentată pe tablă mutarea jucătorului și, următorul la mutare fiind calculatorul, este apelată funcția de evaluare, pentru a stabili mutarea optimă a AI-ului.





**Bibliografie**

* https://www.geeksforgeeks.org/minimax-algorithm-in-game-theory-set-1-introduction/
* https://www.geeksforgeeks.org/minimax-algorithm-in-game-theory-set-2-evaluation-function/?ref=rp
* https://www.geeksforgeeks.org/minimax-algorithm-in-game-theory-set-3-tic-tac-toe-ai-finding-optimal-move/