אלגוריתם מינימקס – Minimax algorithm

האלגוריתם יוצר כל אפשרות משחק של הלוח ומנקד אותה לפי שיטה, האלגוריתם פועל בשיטת רקורסיה.
שיטת הניקוד:

הפסד = (-1) | שוויון = (0) | ניצחון = (1)

<u>דוגמה לניצחון של המחשב בעזרת האלגוריתם:</u>

בכל תור של המחשב אנחנו ננסה למקסמם את הניקוד של אותו המהלך, בכל מהלך בצורת Max הניקוד מתחיל מהערך הכי נמוך וכאן הוא יהיה (-2).
(נתחיל את הדוגמה משלב מאוחר במשחק כדי שלא ניצור יותר מידי עצי אפשרויות)

מצב הבסיס:

תור המחשב לזוז, מסומן בעיגול - O

	0	1	2
0		O	O
1	X	X	O
2		X	

```
def max():
  score = -2
  [x,y] = [None, None]
  winner=checkIfWinner()
  if winner == X:
    return [-1,0,0]
  else if winner == O:
    return [1,0,0]
   else if winner == draw:
    return [0,0,0]
  for emptySpace in Board:
    Board[emptySpace] = O
    [minScore,minX,minY] = min()
    if minScore > score:
      score = minScore
      [x,y] = [emptySpace]
    Board[emptySpace] = ""
  return [score,x,y]
```

```
def min():
  score = 2
  [x,y] = [None, None]
  winners=checkIfWinner()
  if winner == X:
    return [-1,0,0]
  else if winner == O:
    return [1,0,0]
   else if winner == draw:
    return [0,0,0]
  for emptySpace in Board:
    Board[emptySpace] = X
    [maxScore,maxX,maxY] = max()
    if maxScore < score:
      score = maxScore
      [x,y] = [emptySpace]
    Board[emptySpace] = ""
  return [score,x,y]
```

המצב הנ"ל מתפצל לשלוש אפשרויות בגלל שיש שלושה תאים פנויים, התור הוא התור של המחשב(O) ולכן אנחנו ננסה למקסם את התוצאה של כל לוח בעזרת קריאה לפונקציה ממקסמת מתוך הפונקציה הקיימת ללא החזרת ערך לבינתיים. בכל פעם שנקרא לפונקציה הזו אנחנו נבדוק ניצחנו, הפסדנו או שהתוצאה היא תיקו ואז נחזיר את הערכים לפי הבדיקה.
נכתוב את הפונקציות max ו-min בקוד על מנת להבין את הרעיון.
(המטרה של הפונקציה min היא להנמיך את התוצאה כמה שיותר ולכן היא מתחילה בספרה 2, שזו התוצאה הגרועה ביותר בשבילנו (1 זה ניצחון היריב).

<u>שניים משלושת המצבים</u> יגרמו לניצחון ויעבדו באופן הזה:

1. איפסנו את התוצאה לתוצאה הגרועה ביותר.
2. התחלנו לעבור על התאים הריקים, כשמצאנו תא ריק הזנו אליו את הסימון שלנו.
3. נקרא לפונקציה min בגלל שהלוח לא מלא.
4. הפונקציה min עובדת בצורה זהה לפונקציה max בתחילתה והיא בודקת האם יש מנצחים ומחזירה לנו את הניקוד ואת המיקומים של המשחק (הפונקציה min כתובה בעמוד הבא).
5. קיבלנו חזרה ערך אחד, לדוגמה: [1,0,0].
6. חזרנו חזרה לפונקציה max לשורה:
   [minScore=1,minX=0,minY=0] = min()
   הפונקציה בודקת האם 1 גדול מ-(-2), במידה וכן התוצאה והמיקומים נשמרים במשתנים score,x,y והפונקציה מוחקת את הסימון ששמה בתא ועוברת לנסות את האפשרויות הבאות.
7. הפונקציה סיימה לעבור על כל האפשרויות ומחזירה לנו את האפשרות הטובה ביותר.

	0	1	2
0	~~O~~	~~O~~	~~O~~
1	X	X	O
2		X	

	0	1	2
0		O	O
1	X	X	O
2		X	O

	0	1	2
0		O	O
1	X	X	O
2	O	X	

<u>במצב השלישי</u> לעומת זאת המשחק לא יסתיים, ניתן אותו כדוגמא נוספת כדי לראות מה הפונקציות עושות ומחזירות לנו.

1. נכנסנו לפונקציה max שעברה על התאים הריקים והזינה את הסימון שלנו במיקום הפנוי [0,2].
2. נקרא לפונקציה min בגלל שהלוח לא מלא.
3. הפונקציה min לא מזהה שיש מנצחים ולכן מתחילה פעולה.
4. הפונקציה נכנסה ללולאה שעוברת על התאים הריקים והתחילה לעבור עליהם, גם כאן יש לנו שתי אפשרויות:

	0	1	2
0		O	O
1	X	X	O
2	O	X	X

	0	1	2
0	X	O	O
1	X	X	O
2	O	X	

5. הפונקציה תקרא לפונקציה max על הלוח הנוכחי ותבדוק האם יש מנצחים.
6. אין לנו מנצחים ולכן נעבור על התאים הריקים בלוח ונזין בהם את הסימון שלנו:

	0	1	2
0	~~O~~	~~O~~	~~O~~
1	X	X	O
2	O	X	X

	0	1	2
0	X	O	~~O~~
1	X	X	~~O~~
2	O	X	~~O~~

7. יש לנו ניצחון על אחד הלוחות אז נחזיר חזרה לכל הערכים את המיקום המיטבי על מנת להגיע לניצחון.

לאחר כל הפעולות האלו על המחשב לבצע את הפעולה ולסיים את התור שלו.

בעמוד הבא יש תרשים מפורט של כל הפעולות שהקוד עושה ומה הערכים שהוא מקבל ומחזיר.

```
def max():
  score = -2
  [x,y] = [None, None]
  winners=checkIfWinner()      # no winner
  for emptySpace in Board:     # (0,2)
    Board[emptySpace] = O
    [minScore,minX,minY] = min()
    min():
      score = 2
      [x,y] = [None, None]
      winners=checkIfWinner()          # no winner
        for emptySpace in Board:       # (0,0)
            Board[emptySpace] = X
            [maxScore,maxX,maxY] = max()
            max():
              score = -2
              [x,y] = [None, None]
              winners=checkIfWinner() # no winner
              for emptySpace in Board: # (2,2)
                Board[emptySpace] = O
                 [minScore,minX,minY] = min()
                min():
                  winners=checkIfWinner()
                  # winners == O
                  return [1,0,0]
             max():
               [minScore,minX,minY] = [1,0,0]
               if minScore > score:       # 1 > -2
                 score = minScore         # = 1
                 [x,y] = [emptySpace]     #(2,2)
                 Board[emptySpace] = ""
               return [score,x,y]          # [1,2,2]
    min():
      [1,2,2] = min()
      if minScore > score:       # 1 > -2
        score = minScore         # = 1
        [x,y] = [emptySpace]     # (2,2)
        Board[emptySpace] = ""
       return [score,x,y]
max():
  [minScore,minX,minY] = [1,2,2]
    if minScore > score:         # 1 > -2
      score = minScore           # = 1
      [x,y] = [emptySpace]       # (2,2)
    Board[emptySpace] = ""
  return [score,x,y]
# returns an action needed to do to win!
```

	0	1	2
0		O	O
1	X	X	O
2	O	X	

	0	1	2
0	X	O	O
1	X	X	O
2	O	X	

	0	1	2
0	X	O	O
1	X	X	O
2	O	X	O

	0	1	2
0	X	O	O
1	X	X	O
2	O	X	O