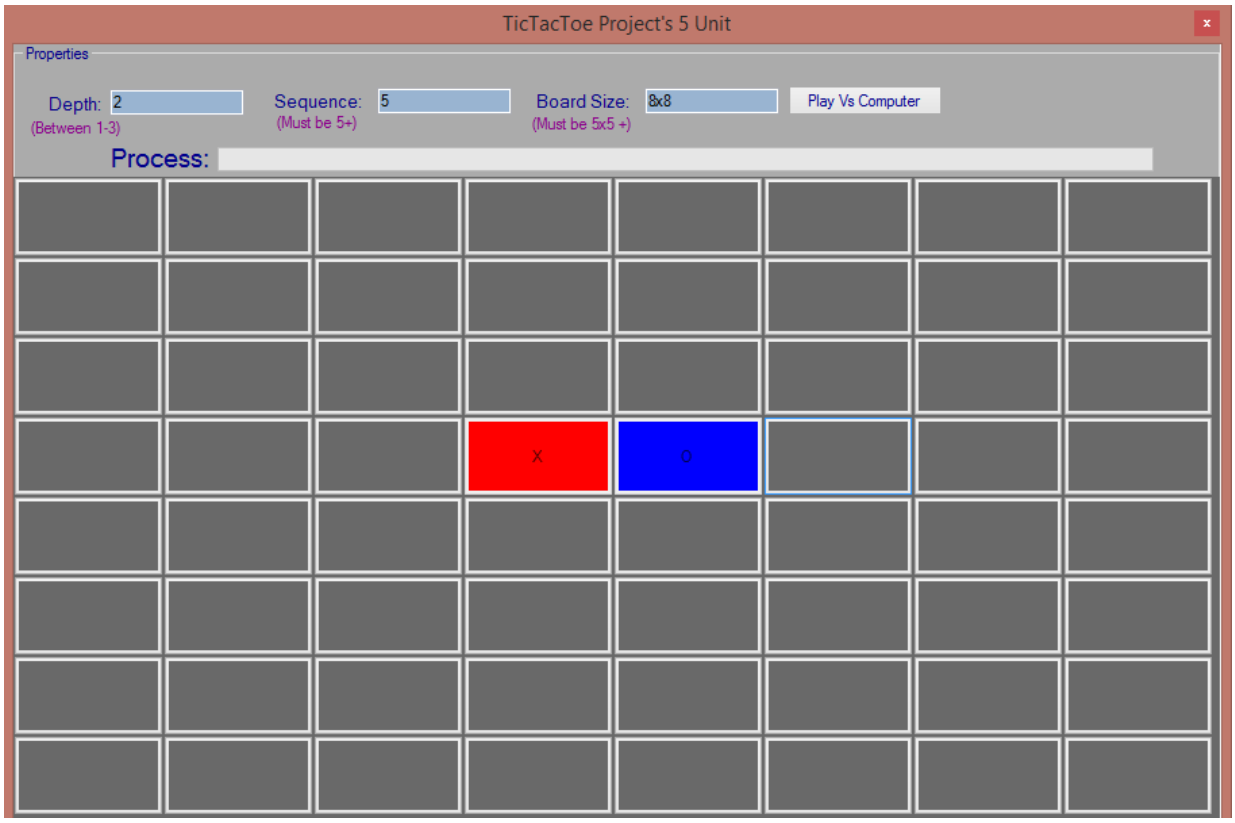
פרויקט בגרות במדעי המחשב 5 יח"ל
בית הספר זיו ומרקס, ירושלים

שם הפרויקט: איקס עיגול עם רצף ומהלכי חשיבה
מגיש: ינאי אלטר
ת"ז: 211377775
בהנחיית המורה: מאיר פנחס



הדר ב' תשע"ו

תוכן עניינים:

מבוא למשתמש

איקס עיגול הוא משחק לשני שחקנים המשוחק על לוח של 3,3 משבצות. כל שחקן בתורו מסמן סימן איקס או עיגול באחת מן המשבצות. השחקן המנצח הוא זה שהצליח למלא שורה, טור או אלכסון בסימנים שלו. אם אף שחקן לא הצליח לעשות זאת והלוח תפוס בכל המקומות, או שאף שחקן לא יכול ליצור ניצחון, יש תיקו בין השחקנים.

אם באיקס עיגול, בלוח 3,3 שני השחקנים שיחקו באופן מושלם (כל אחד עשה את המהלך הטוב ביותר), יהיה תיקו תמיד.

באיקס עיגול לוח 3,3 המהלכים פשוטים וקלים למחשבה, המשחק נהיה מסובך ברגע שמוסיפים וריאציות, כלומר, אפשרות לשינוי גודל הלוח, אפשרות לשינוי הרצף, וכדומה.

המשתמש בתוכנה זאת ישחק נגד המחשב, המחשב יהיה 'O' והמשתמש יהיה 'X'.

תוכלו לבחור את עומק החשיבה של המחשב, יש לציין שגם בחשיבה הכי נמוכה של המחשב, המחשב ישחק חכם, ומהווה אתגר למשתמש.

"משחק מושלם" הוא משחק שכל אחד מהשחקנים מבצעים את המהלכים הטובים ביותר, ובמקרה הזה מקבלים תיקו.

חוקי המשחק

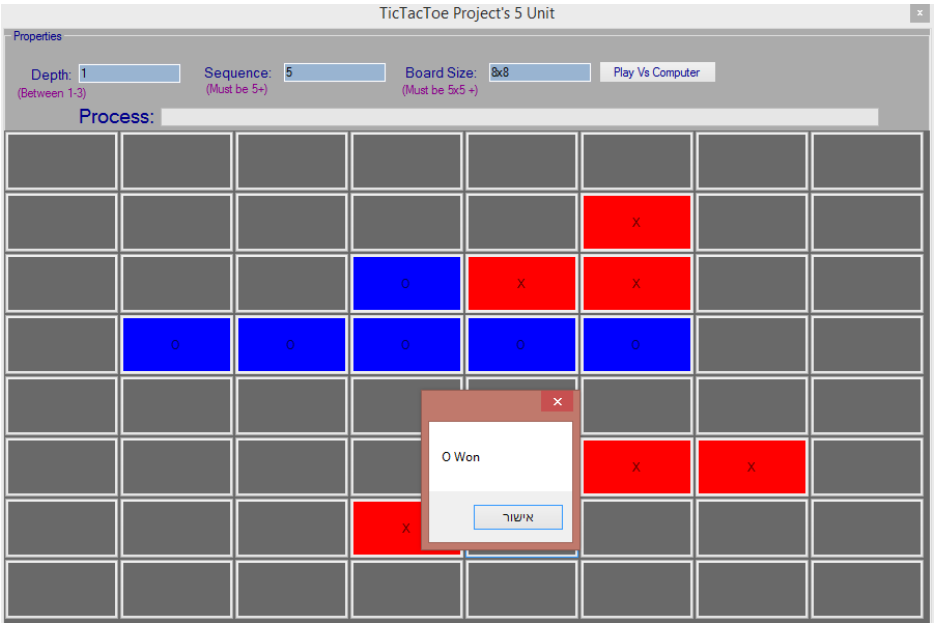
במשחק ישנם שני שחקנים. שחקן אחד הוא המחשב – עיגול, והשחקן השני הוא המשתמש – איקס. מטרת המשחק היא ליצור רצף (שהמשתמש בוחר) של איקסים, בשורה או בעמודה או באלכסון.

המשתמש ישחק נגד המחשב, אם המחשב יוצר את הרצף המבוקש, המחשב ניצח.

אם המשתמש יוצר את הרצף המבוקש, המשתמש ניצח.

המשחק מתמשך כל עוד יש אפשרות לנצח.

ברגע שאין אפשרות לנצח, התוכנה תודיע על תיקו.



(דוגמא לניצחון של המחשב ברצף של 5, לוח 5x5)

מדריך למשתמש

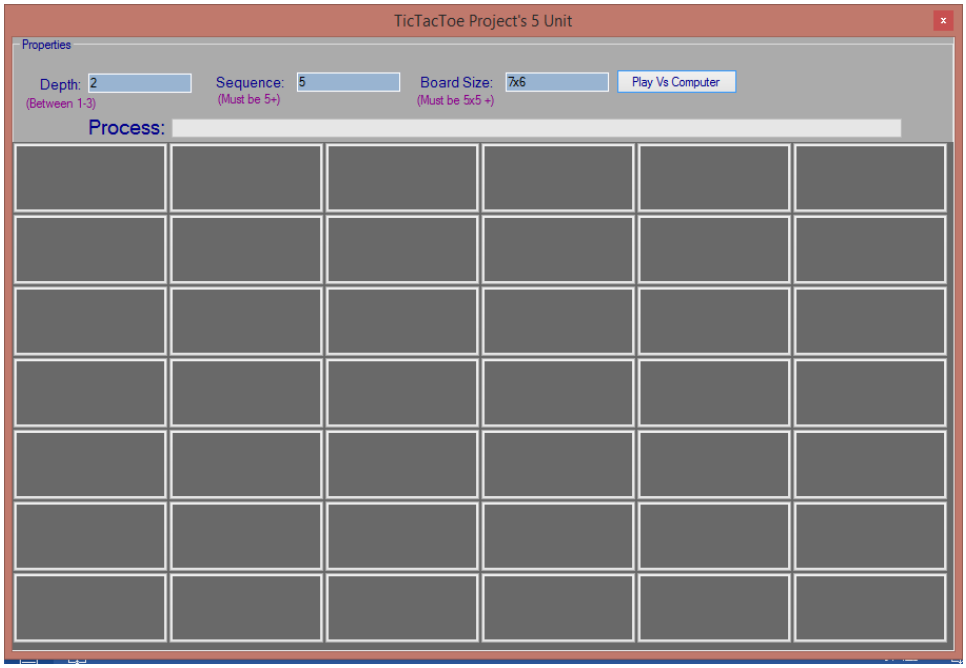
במשחק תוכלו למצוא מגוון אופציות:

בחירת גודל הלוח (Board Size): תוכלו לבחור את גודל הלוח. נניח ואתם רוצים לשחק בלוח 5x5, תוכלו לרשום בתיבה Board Size: 5x5. ניתן גם לקבוע שהשורה אינה תהיה שווה לעמודה (לדוגמא 5x6). מכיוון שבמשחק עם לוח שהשורה קטנה מ-5 או העמודה הקטנה מ-5 יש טכניקות לניצחון בטוח, לא תוכלו לבחור בגדלים אלו.

בחירת רצף המשחק (Sequence): תוכלו לבחור את רצף הסימנים כדי לנצח. שימו לב, כמובן שלא ניתן לרשום רצף יותר גבוהה מגודל הלוח. נניח וגודל הלוח הוא 6x5, מקסימום הרצף שניתן לבחור הוא 5. במקרה שתבחרו רצף גבוהה יותר מגודל הלוח, תקבלו הודעת שגיאה! בנוסף, ישנה הגבלה מינימלית של הרצף. הרצף לא יהיה קטן מ-5 מאחר וכאשר הרצף קטן מ-5 יש טכניקות לניצחון בטוח.

בחירת רמת החשיבה של המחשב (Depth): תוכלו לבחור את רמת החשיבה של המחשב, שימו לב שככל שרמת החשיבה גבוהה יותר, זמן התגובה איטי יותר. רמת חשיבה של 1 היא הנמוכה ביותר. מ-3 ומעלה המחשב מתחיל להגיב תוך דקות. 2 היא הרמה הממוצעת (וכמובן גם מאתגרת). בפרויקט יש הגבלה לעומק החשיבה (ניתן לבחור בעומק 1-3) מכיוון שבחירת עומק חשיבה גדול מ-3 יכולה לגרום למחשב להתאמץ יותר מידי (מאות אלפי פעולות).

- המשחק מתחיל כאשר אתם לוחצים על אחת מן המשבצות. מיד אחרי תוכלו לעקוב אחרי תהליך החשיבה של המחשב ולדעת מתי המחשב מסיים לחשוב ומשחק. תוכלו לראות מתחת למאפיינים את המלבן שמתמלא בצבע ירוק. ברגע שהצבע ירוק מגיע לסופו המחשב עושה את תורו. עד אז, צריך לחכות בסבלנות שהמחשב יסיים את התור.



דוגמא להצגת הלוח של גודל 7x6, רצף 5, עומק חשיבה 2.

מצבים במשחק

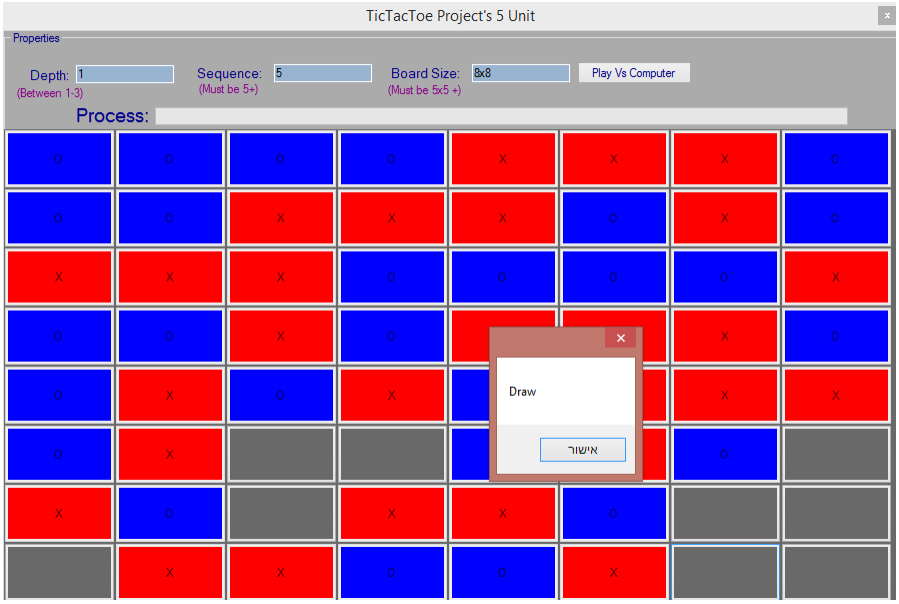
ללא הגבלת כלליות, המצבים בספר על לוח 8x8, רצף 5, עומק חשיבה 2.

מצב ניצחון למחשב:

מצב ניצחון למשתמש:

בלוח 8x8, רצף 5, עומק 2, לא נמצאה אפשרות למשתמש לנצח את המחשב.

מצב תיקו: כאשר אין אפשרות לנצח. תיקו יכול להיקבע גם אם הלוח אינו מלא (כאשר אין אפשרות ליצור ניצחון).



מבוא למתכנת

משחק האיקס עיגול נבנה בשפת C#. המשחק מבוסס על האלגוריתם MiniMax, שהוא האלגוריתם המרכזי, המחשב מקבל עומק חשיבה, יוצר משחק ווירטואלי בהתאם לעומק החשיבה, ולפיו מחשב את ניקוד הלוח. המקום שמקבל את הניקוד הגבוהה ביותר הוא המקום שיבחר על ידי המחשב ויסומן ב'O' (עיגול). כמובן שזה רק בתור של המחשב. עומק החשיבה יהיה אחראי על כמה תורות המחשב ישחק וירטואלי. אם עומק החשיבה הוא 3 למשל, המחשב ישחק משחק וירטואלי של עיגול איקס ועיגול, יבדוק את הניקוד הגבוהה ביותר שקיבל מהמשחק, וישמור את השורה והעמודה של המיקום של העיגול שקיבל את הניקוד הגבוהה ביותר.

<u>האלגוריתם הכללי פועל כך:</u>

0.0 המשתמש לוחץ על אחד מהכפתורים על מנת לסמן איקס ←

0.1 המחשב קולט את הלחיצה, מסמן את האיקס בלוח בצבע אדום והתור עובר לעיגול ←

1.0 המחשב קורא לפעולה **WinOrBlock** שבודקת האם יש אפשרות לנצח או לחסום את המשתמש. אם יש אפשרות, היא עושה זאת, אחרת, היא קוראת לפעולה **MiniMax** ←

1.1 הפעולה **MiniMax** קוראת לפעולה הרקורסיבית **MiniMaxNext** על מנת לחשב את הניקוד של הלוח לפי משחק דמיוני שהמחשב יעשה, לפי עומק החשיבה שהמשתמש בחר בהתחלה. הפעולה **MiniMaxNext** תקרא לפעולה **ScoreCalculate** שתחשב את הניקוד בלוח לפי פקודות מסוימות ←

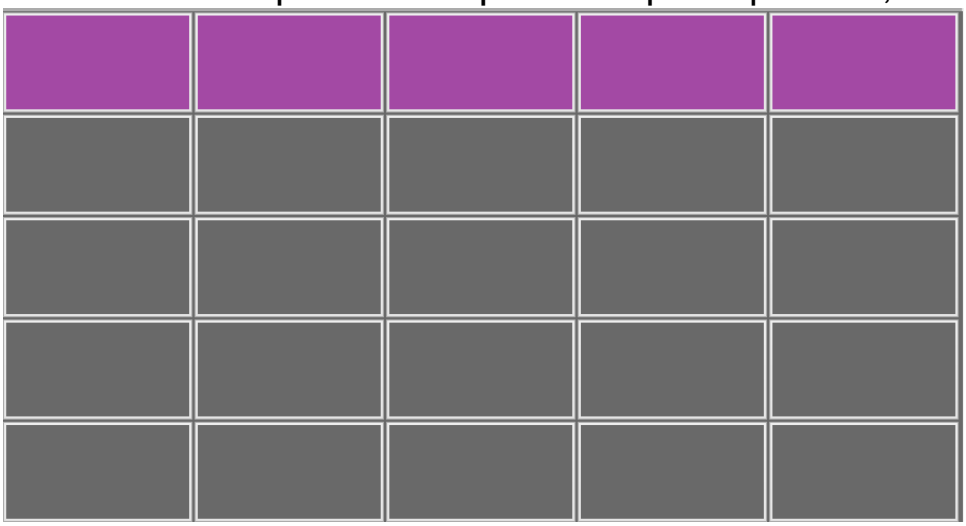
1.2 הפעולה **MiniMaxNext** תחזיר **לMiniMax** ניקוד. המחשב ישמור את המקום שקיבל את הניקוד הגבוהה ביותר ויסמן את המקום ב'O' (עיגול).

פירוט על האלגוריתם של כל פעולה בנפרד ראו בהמשך הספר.

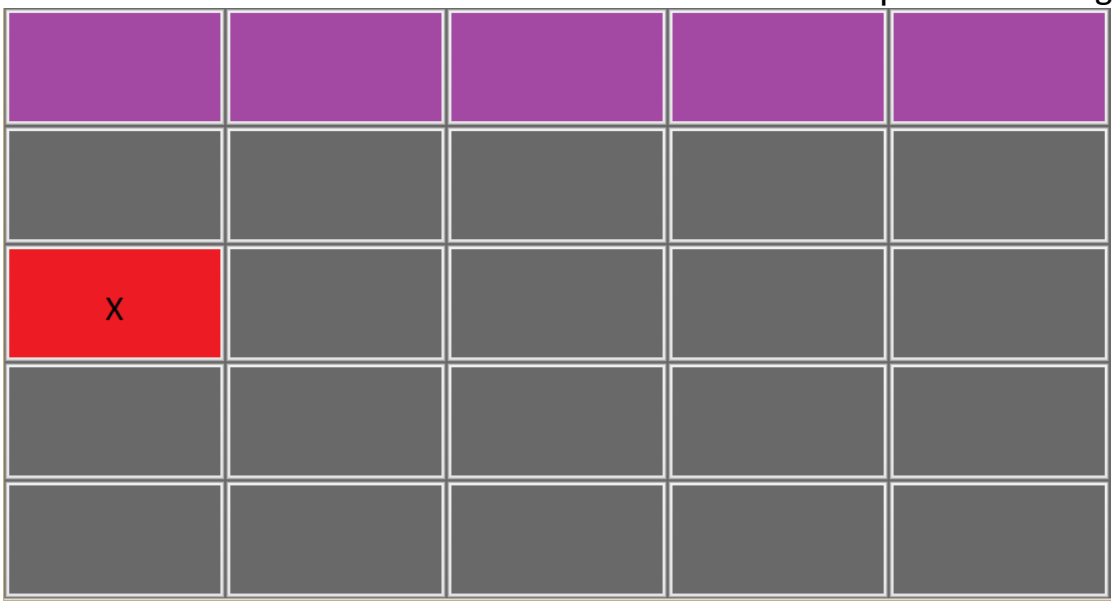
<u>מערכי מונים:</u> בפרויקט יש 4 מערכי מונים. תפקידם לייצג אפשרויות ניצחון (כל איבר במערך המונים הוא מקום שממנו אפשר ליצור ניצחון).

מערך מונה רוחב (a_XOStatus6):

האיברים במערך זה מייצגים אפשרויות לניצחון ברוחב הלוח.
כל מקום בלוח, שממנו אפשר לעלות שורה רצף פעמים מבלי לחרוג מגודל הלוח, הוא חלק ממערך זה. נתבונן בלוח 5x5 רצף 5:



גודל המערך הוא הכפתורים המסומנים בסגול (שורה אחת ו-5 עמודות במקרה זה). מכאן אפשר להסיק שבלוח 5x5 רצף 5, יש 5 אפשרויות לנצח לאורך.
ערך המערך במקום [x,y] יהיה האיקסים שיש ממקום [x,y] עד מקום [x,y+Rezef] כאשר x לא משתנה ו Rezef הוא רצף המשחק. ערך המערך הוא string שכולל רק X או O.



אז a_XOStatus2[0,0] יהיה "X" (שכן יש איקס אחד באפשרות ניצחון של המקום 0,0). אפשר להגיד את זה גם ככה: אם נרד שורה מ0,0 רצף פעמים ונספור את האיקסים, יהיה איקס אחד (במקרה הזה).

מערך מונה אלכסון ראשי (a_XOStatus3):

כל מקום, ממנו אפשר לעלות שורה ועמודה רצף פעמים, מבלי לחרוג מגודל הלוח, הוא חלק ממערך זה. במקרה של לוח 5,5 רצף 5 יש רק מקום אחד כזה והמערך מכיל איבר אחד בלבד.

נתבונן במצב הזה:

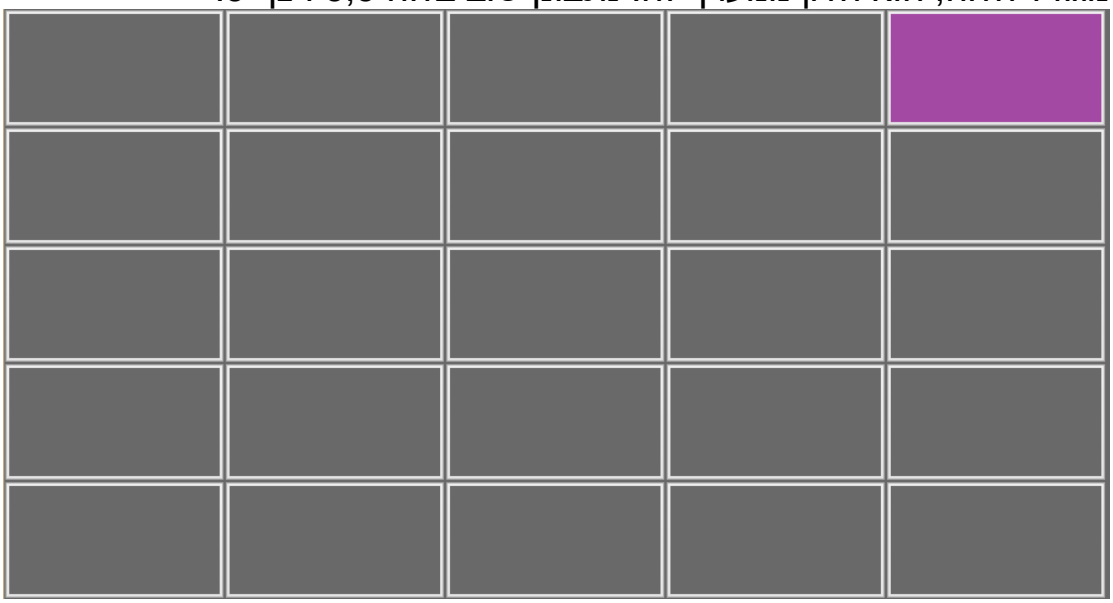
X				
	O			
		O		
			X	
				X

במצב הזה, ערך a_XOStatus3[0,0] הוא "XOOXX" (הסדר יכול להיות שונה, תלוי מה נלחץ קודם). הספירה מתחילה מ0,0, נבדק אם יש 'X', אם כן, מתווסף לערך המערך "X". כך יורדים רצף פעמים שורה ועמודה ומוסיפים איקסים או עיגולים לפי מה שנמצא.

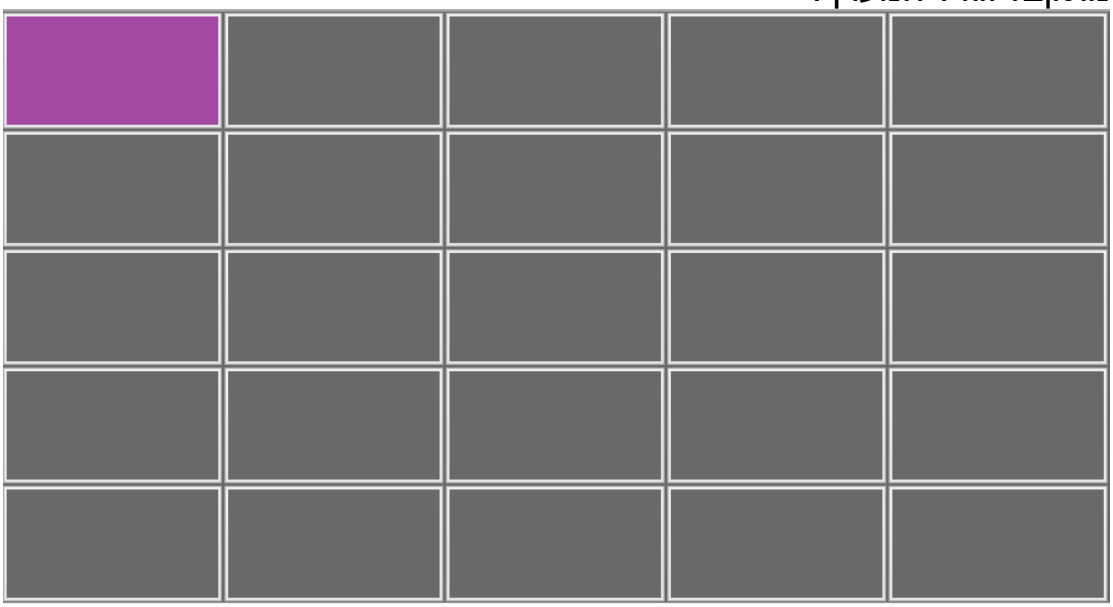
חשוב לציין שכשיש איבר במערך המונים שהערך שלו מכיל גם X וגם O הוא חסר ערך מכיוון שאי אפשר ליצור דרכו ניצחון כלשהו.

מערך מונה אלכסון משני (a_XOStatus1):

כל מקום, ממנו אפשר לעלות שורה ולרדת עמודה רצף פעמים, מבלי לחרוג מגודל הלוח, הוא חלק ממערך זה. נתבונן שוב בלוח 5,5 רצף 5:

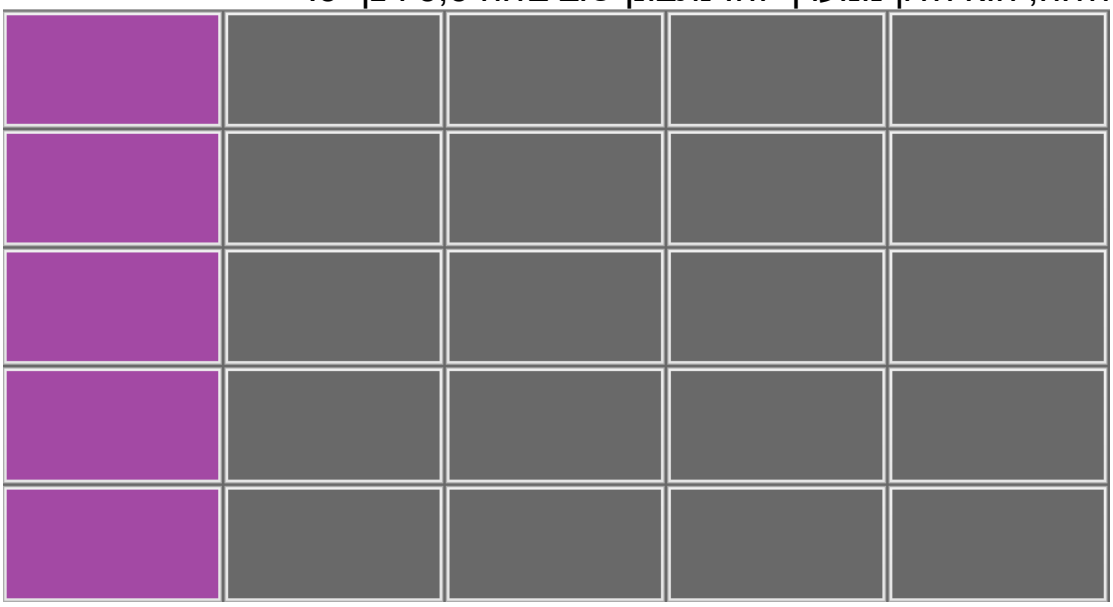


אפשר לראות כאן שרק [0,4] (שורה 0 עמודה 4) חלק ממערך זה. אך איך ייתכן שמערך יתחיל מצד ימין? במערך מונה אלכסון משני ישנה המרה. מורידים למספר העמודות שלו את מספר הרצף, מוסיפים לתוצאה אחד, ואז מתקבל גודל המערך:



מערך מונה אורך (a_XStatus2):

כל מקום, ממנו אפשר לעלות שורה ועמודה רצף פעמים, מבלי לחרוג מגודל הלוח, הוא חלק ממערך זה. נתבונן שוב בלוח 5,5 רצף 5:



אפשר לראות שמקרה זה, גודל המערך יהיה [5,1], 5 שורות ועמודה אחת.

- יש להבהיר שהצביעה בסגול וצהוב נעשתה לצורך המחשת מערך המונים. צבעים אלו אינם נמצאים בפרויקט.

5 אלגוריתמים מרכזיים:

אלגוריתם הWinOrBlock: הקריאה לפעולה מתבצעת לאחר שהמשתמש מסמן איקס על הלוח והתור עובר למחשב (לעיגול).

האלגוריתם:

0.0 עבור על כל מערך המונים a_XOStatus1
0.1 בדוק אם אתה יכול לנצח
אם כן תעשה 0.2 זאת, אחרת, המשך ←
0.3 בדוק אם למשתמש נשאר רק צעד אחד כדי לנצח,
אם כן, חסום אותו, אחרת, המשך ←
0.4 בדוק אם אתה 2 צעדים מניצחון, אם כן, תוסיף 'O' באחד מהמקומות בו אתה 2 צעדים מהניצחון אחרת, המשך ←
0.5 בדוק אם למשתמש 2 צעדים מניצחון. אם כן, חסום אותו, אחרת, המשך ←

1.0 עבור על כל מערך המונים a_XOStatus2
1.1 בדוק אם אתה יכול לנצח – אם כן תעשה 1.2 זאת, אחרת, המשך ←
1.3 בדוק אם למשתמש נשאר רק צעד אחד כדי לנצח,
אם כן, חסום אותו, אחרת, המשך ←
1.4 בדוק אם אתה 2 צעדים מניצחון, אם כן, תוסיף 'O' במקום בו אתה 2 צעדים מהניצחון, אחרת, המשך ←
1.5 בדוק אם למשתמש 2 צעדים מניצחון, אם כן, חסום אותו, אחרת, המשך ←

2.0 עבור על כל מערך המונים a_XOStatus3
2.1 בדוק אם אתה יכול לנצח – אם כן תעשה 2.2 זאת, אחרת, המשך ←
2.3 בדוק אם למשתמש נשאר רק צעד אחד כדי לנצח,
אם כן, חסום אותו, אחרת, המשך ←
2.4 בדוק אם אתה 2 צעדים מניצחון, אם כן, תוסיף 'O' במקום בו אתה 2 צעדים מהניצחון, אחרת, המשך ←
2.5 בדוק אם למשתמש 2 צעדים מניצחון, אם כן, חסום אותו, אחרת, המשך ←

3.0 עבור על כל מערך המונים a_XOStatus6
3.1 בדוק אם אתה יכול לנצח – אם כן תעשה 3.2 זאת, אחרת, המשך ←
3.3 בדוק אם למשתמש נשאר רק צעד אחד כדי לנצח,
אם כן, חסום אותו, אחרת, המשך ←
3.4 בדוק אם אתה 2 צעדים מניצחון, אם כן, תוסיף 'O' במקום בו אתה 2 צעדים מהניצחון, אחרת, המשך ←
3.5 בדוק אם המשתמש 2 צעדים מניצחון. אם כן, חסום אותו.

<u>אלגוריתם הMiniMax:</u>

הקריאה לפעולה MiniMax מתבצעת אחרי WinOrBlock אם WinOrBlock לא הצליחה לסמן 'O' (אף אחד מהתנאים שבפעולה לא התקיימו).

MiniMax מקבל פרמטר אחד:

depth – עומק החשיבה

<u>האלגוריתם:</u>

0.0 אתחל int maxScore = -1000;

1.0 עבור על כל המקומות בגודל הלוח

1.1 לכל מקום בגודל הלוח:

1.2 סמן את המקום כתפוס

1.3 הוסף נקודות דמיוניות למערכי המונים לפי המקום החדש

1.4 אתחל int score = MiniMaxNext(depth-1); כאשר depth עומק החשיבה

1.5 שקיבלת מהמשתמש

1.6 הורד נקודות דמיוניות למערכי המונים לפי המקום החדש

1.7 סמן את המקום כפנוי

1.8 בדוק, האם score > maxScore?

1.8.1 **אם כן, הגדר:**

maxScore = score;

bestI =i; כאשר i הוא השורה של המקום בו אתה נמצא, וbestI משתנה כללי שהוגדר בתחילת קוד הפרוייקט

bestJ = j; כאשר j הוא העמודה של המקום בו אתה נמצא, וbestJ משתנה כללי שהוגדר בתחילת קוד הפרוייקט.

אלגוריתם הMiniMaxNext:

הקריאה לMiniMaxNext מתבצעת בפעולה MiniMax.
MiniMaxNext מקבל 2 פרמטרים:
(int) Depth – עומק החשיבה
(char) Turn – תור, 'X' – משתמש, 'O' – מחשב.

חשוב לציין שבאלגוריתם זה, המחשב יוצא מנקודת הנחה שהמשתמש חכם ולכן הוא מבצע את המהלכים בהנחה שהמשתמש עושה את הצעד החכם ביותר, זה בא על ידי ביטוי באלגוריתם זה מכיוון שאם התור של הX, המקום שקיבל את הכי קצת נקודות של העיגול (הכי הרבה ל'X') הוא המקום שיוחזר.

האלגוריתם:

0.0 אם העומק הוא 0, החזר את ;()ScoreCalculate
1.0 אם התור של O (עיגול):
1.1 אתחל ;int maxScore = -1000
1.2 עבור על כל גודל הלוח
1.3 לכל מקום ריק:
1.4 סמן את המקום כתפוס
1.5הוסף נקודות למקום (במערי המונים)
1.6 אתחל (int scoreO = MiniMaxNext(depth-1,'X'
1.7 סמן את המקום כפנוי
1.8 הורד נקודות למקום (במערכי המונים)
1.9 בדוק scoreO > maxScore?
1.9.1 אם כן:
1.9.1.1 הגדר ;maxScore = scoreO
1.9.2 אחרת המשך ←
2 החזר את maxScore

3 אם התור של X (איקס):
3.1 אתחל ;int minScore = 1000
3.2 עבור על כל גודל הלוח
3.3 לכל מקום ריק:
3.4 סמן את המקום כתפוס
3.5 הוסף נקודות דמיוניות למקום
3.6 אתחל (int scoreX = MiniMaxNext(depth-1,'O'
3.7 סמן את המקום כפנוי
3.8 הורד נקודות דמיוניות למקום
3.9 בדוק scoreX < minScore?
3.9.1 אם כן:
3.9.1.1 הגדר ;minScore = scoreX
3.9.2 אחרת המשך ←
4.0 החזר את minScore

אלגוריתם הScoreCalculate:

הקריאה לScoreCalculate מתבצעת בMiniMaxNext כאשר עומק החשיבה הוא 0.

האלגוריתם:

0.0 אתחל score = 0;

1.0 עבור על כל גודל המערך a_XOStatus1

1.1 לכל מקום במערך:

1.2 אם a_XOStatus1 במקום זה לא מכיל איקסים, הוסף ל-score את אורך המחרוזת של a_XOStatus1 במקום זה.

2.0 עבור על כל גודל המערך a_XOStatus2

2.1 לכל מקום במערך:

2.2 אם a_XOStatus2 במקום זה לא מכיל איקסים, הוסף ל-score את אורך המחרוזת של a_XOStatus2 במקום זה.

3.0 עבור על כל גודל המערך a_XOStatus3

3.1 לכל מקום במערך:

3.2 אם a_XOStatus3 במקום זה לא מכיל איקסים, הוסף ל-score את אורך המחרוזת של a_XOStatus3 במקום זה.

4.0 עבור על כל גודל המערך a_XOStatus6

4.1 לכל מקום במערך:

4.2 אם a_XOStatus6 במקום זה לא מכיל איקסים, הוסף ל-score את אורך המחרוזת של a_XOStatus6 במקום זה.

אלגוריתם בדיקת ניצחון:

בדיקת הניצחון מתבצעת בתוך הפונקצייה addPoints.

האלגוריתם:

1.0 עבור על כל המערך a_XOStatus1

1.1 לכל מקום במערך

1.1.1 אם a_XOStatus1 במקום הזה לא מכיל איקסים ואורך המחרוזת של a_XOStatus1 במקום הזה שווה לרצף קרא לפונקציה ResetPanelWithMessagebox("O Won");

1.1.2 אם a_XOStatus1 במקום הזה לא מכיל עיגולים ואורך המחרוזת של a_XOStatus1 במקום הזה שווה לרצף קרא לפונקציה ResetPanelWithMessagebox("X Won");

2.0 עבור על כל המערך a_XOStatus2

2.1.0 לכל מקום במערך

2.1.1 אם a_XOStatus2 במקום הזה לא מכיל איקסים ואורך המחרוזת של a_XOStatus2 במקום הזה שווה לרצף קרא לפונקציה ResetPanelWithMessagebox("O Won");

2.1.2 אם a_XOStatus2 במקום הזה לא מכיל עיגולים ואורך המחרוזת של a_XOStatus2 במקום הזה שווה לרצף קרא לפונקציה ResetPanelWithMessagebox("X Won");

3.0 עבור על כל המערך a_XOStatus3

3.1.0 לכל מקום במערך

3.1.1 אם a_XOStatus3 במקום הזה לא מכיל איקסים ואורך המחרוזת של a_XOStatus3 במקום הזה שווה לרצף קרא לפונקציה ResetPanelWithMessagebox("O Won");

3.1.2 אם a_XOStatus3 במקום הזה לא מכיל עיגולים ואורך המחרוזת של a_XOStatus3 במקום הזה שווה לרצף קרא לפונקציה ResetPanelWithMessagebox("X Won");

4.0 עבור על כל המערך a_XOStatus6

4.1.0 לכל מקום במערך

4.1.1 אם a_XOStatus6 במקום הזה לא מכיל איקסים ואורך המחרוזת של a_XOStatus6 במקום הזה שווה לרצף קרא לפונקציה ResetPanelWithMessagebox("O Won");

4.1.2 אם a_XOStatus6 במקום הזה לא מכיל עיגולים ואורך המחרוזת של a_XOStatus6 במקום הזה שווה לרצף קרא לפונקציה ResetPanelWithMessagebox("X Won");

טענות כניסה ויציאה

```
//הפונקצייה בודקת אם יש אפשרות למחשב לנצח או לחסום את המשתמש ואם יש היא חוסמת או מנצחת
//
public void WinOrBlock()...
```

```
//הפונקצייה מחשבת את ניקוד הלוח לפי מערכי המונים, ומחזירה את הניקוד
//
private int ScoreCalculate()...
```

```
//ט. כניסה: הפונקצייה קולטת את עומק החשיבה והתור (איקס או עיגול)
//ט. יציאה: הפונקצייה מחזירה ניקוד (מספר שלם) הגבוהה או הנמוך ביותר שהתקבל בלולאה
//אם התור הוא איקס, יוחזר הערך הנמוך ביותר שהתקבל בלולאה
//אם התור הוא עיגול, יוחזר הערך הגבוהה ביותר שהתקבל בלולאה
public int MiniMaxNext(int depth, char Turn)...
```

```
//ט. כניסה: הפונקצייה קולטת את עומק החשיבה
//יציאה: הפונקצייה מגדירה את השורה והעמודה הכי טובים לעיגול לפי הניקוד שהיא מקבלת מהפונקצייה מינימקס
private void MiniMax(int depth)...
```

```
//הפונקצייה קוראת לפונקציית הניקוי של הפאנל וכך מנקה את הפאנל
//
private void resetPanel()...
```

```
//ט. כניסה: הפונקצייה קולטת שורה, עמודה, סימן (-1 או 1), ופסוק בוליאני - מצב אמת או דימיוני
// ט. יציאה: הפונקצייה מוסיפה איקס או עיגול למערכי המונים לפי הפרמטרים שהתקבלו
//ולפי התור
private void addPoints(int i, int j,int sign,bool dimion)
```

```
//הפונקצייה מגדירה משתנים התחלתיים וקבועים, גבולות של מערכים, מטפלת במקרי קצה
//
private void setProperties()...
```

```
//הפונקצייה מציירת על הפאנל לוח ממערך כפתורים לפי גודל השורה והעמודה
//
private void drawButtons()
```

```
//הפונקצייה בודקת אם יש ניצחון לאחד מהשחקנים
//
public void CheckWin()...
```