תיאור מצב

נתאר מצב אמצעות אובייקט בשם Board

לBoard יהיו המאפיינים הבאים:  
**מערך דו מדדי (board)** בגודל של הלוח (8X8) בכל תא במערך יהיה או **דיסקית (piece)** לבנה או דיסקית שחורה או 0 לסמל מקום פנוי.  
**משתנה (turn)** שמציין תור מי לשחק עכשיו  
**משתנים (white\_pieces , black\_pieces)** שמציינים כמה דיסקיות יש לכל שחקן.

**משתנה(turns\_played)** הסופר כמה תורות שוחקו מתחילת המשחק.   
  
מצב התחלתי של הלוח ייראה כך:   
( [ [ 0, 0, 0, 0, 0, 0, 0, 0] ,

[ 0, 0, 0, 0, 0, 0, 0, 0] ,

[ 0, 0, 0, 0, 0, 0, 0, 0] ,

[ 0, 0, 0, ל, ש, 0, 0, 0] ,

[ 0, 0, 0, ש, ל, 0, 0, 0] ,

[ 0, 0, 0, 0, 0, 0, 0, 0] ,

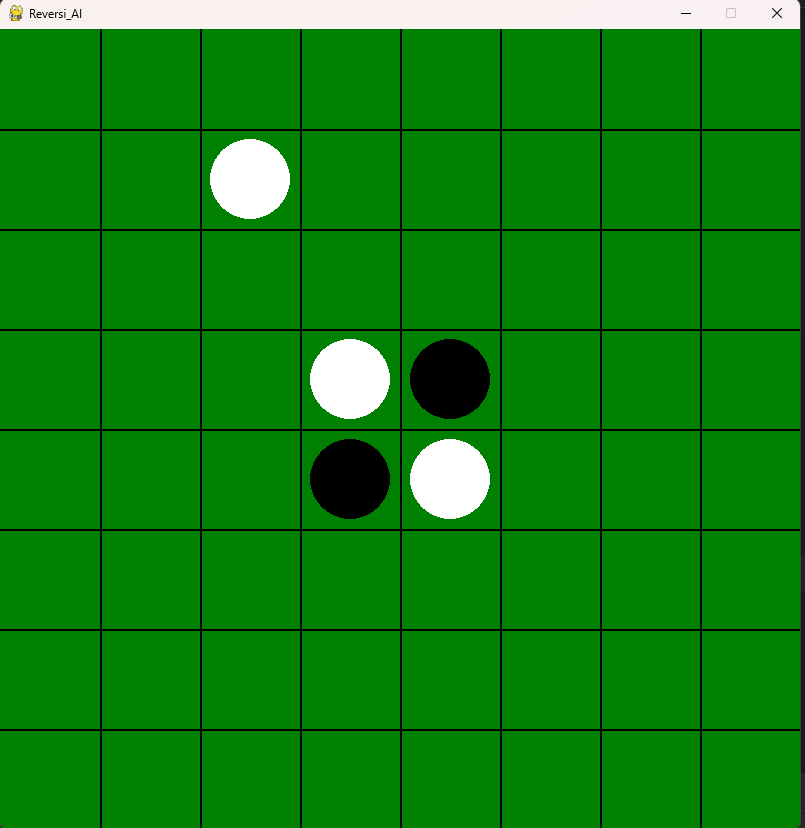
[ 0, 0, 0, 0, 0, 0, 0, 0] ,

[ 0, 0, 0, 0, 0, 0, 0, 0] ] , לבן, 2, 2, 0)

**שאלה 1**

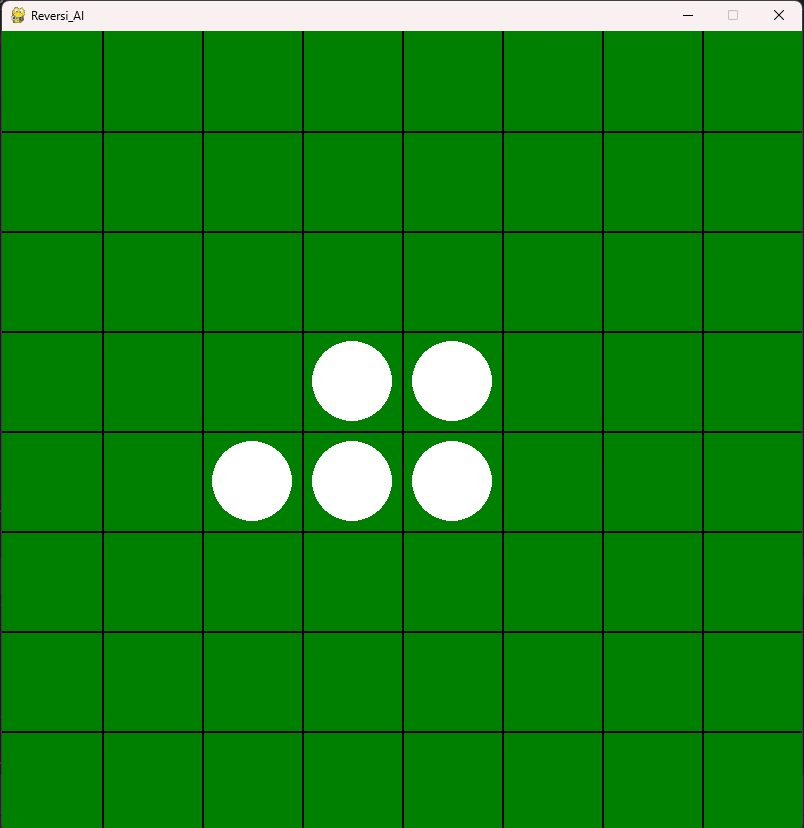
א) גודל מרחב המצבים   
ללא התחשבות במצבים תקינים\לא תקינים   
בכל אחד מ64 המקומות על הלוח יכול להיות דיסקית שחורה לבנה או שאין דיסקית לכן ניתן לתאר בעזרת מרחב המצבים 364

כמובן שבתיאור זה יש הרבה מצבים לא חוקיים , לגביי מצבים חוקיים ולא חוקיים נדבר בהמשך.

ב)

**מצב בלתי חוקי:**ניתן להניח דיסקיות רק בצמוד

לדיסקיות הקיימות



**מצב בלתי נגיש:**

אין אף פעולה אחד שתביא  
למצב הזה (הנחת דיסקית אחת  
בתחילת המשחק כך שכל שאר  
הדיסקיות ייעשו לבנות גם כן.)

מודל מעברים ראשוני **(next\_moves)**:

נעבור משבצת משבצת בלוח, אם המשבצת ריקה **(piece == 0)** נבדוק האם ניתן להניח בה דיסקית **(is\_legal)** הבדיקה תתצבע בצורה הבא:

נגדיר כיוונים אליהם ניתן ללכת מהמשבצת. **(moves)**

את moves הגדרתי כתכונה של הלוח, למרות שזה לא מרגיש מאוד "נכון"

אלכסון שמאלה למעלה, למעלה, אלכסון ימין למעלה,

ימינה, שמאלה,

אלכסון שמאל למטה, למטה, אלכסון ימין למטה. 

עבור כל אחד מהכיוונים נבדוק אם במשבצת הבאה בכיוון יש דיסקית בצבע של היריב(**opponent\_found)**  אם כן נמשיך לטייל עד אשר נפגוש את:  
קצה הלוח, משבצת ריקה,

דיסקית בצבע שלנו כאשר המשתנה opponent\_fond =False

אם אחד מהמקרים האלו קרה, הכיוון אינו חוקי, ונבדוק את הכיוון הבא.  
אם פגשנו דיסקית בצבע שלנו לאחר שפגשנו דיסקית של היריב   
נחזיר אמת, ואז נוסיף את המשבצת(כלומר את המהלך לשים משבצת בצבע שלנו במשבצת זו) לרשימת המלהכים החוקיים.  
אם עברנו על כל הכיוונים עבור משבצת זו ואף כיווון אינו חוקי נחזיר false ונתקדם למשבצת הבאה.  
**בתום התהליך יש לנו רשימה של כל המהלכים החוקיים עבור המצב הנוכחי.**

**ג.**  כן, כמובן שבכל הרצה מתקבלת אותו תוצאה, שום דבר אינו משתנה, תמיד ייבחרו אותם המהלכים.  
לבן בוחר את המהלך הראשון שלו ומביא תמיד לאותו מצב עבור השחור, והוא בתגובה בוחר את המהלך הראשון שלו וכן הלאה.

**שאלה 2**

פונקציית היוריסטיקה.  
כפי שניתן לראות בקוד, ניסיתי שלל פונקציות יוריסטיקה שונות עם שלל משקלים שונים עבור כל אלמנט ביוריסטיקה.  
אתאר בקצרה את התהליך:  
1) יוריסטיקה נאיבית-> מספר הדיסקיות שלי פחות מספר הדיסקיות של היריב.

נקרא בקוד weak heuristic

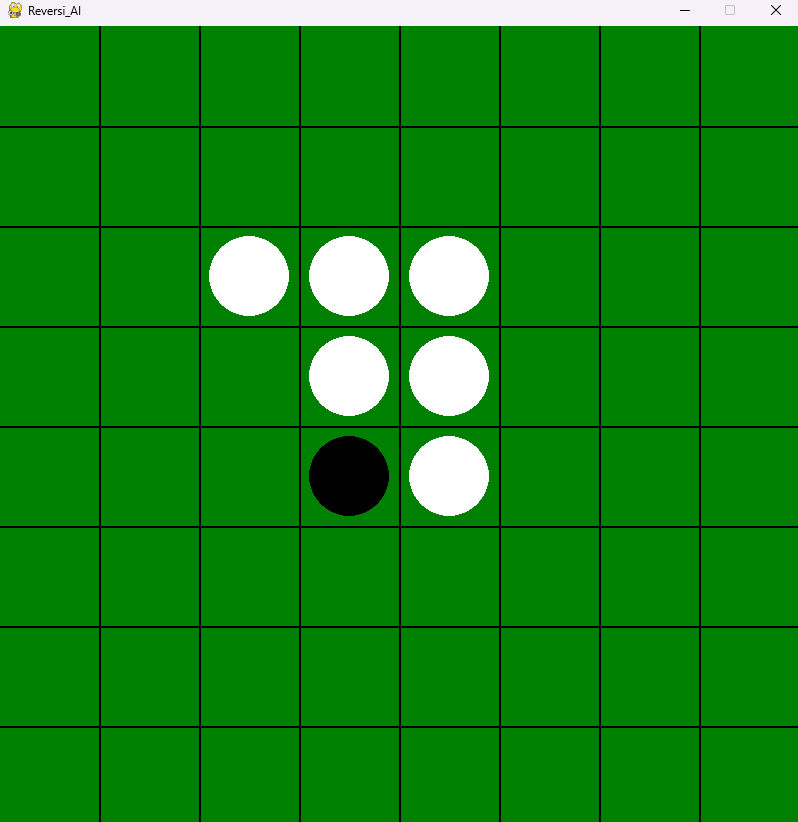
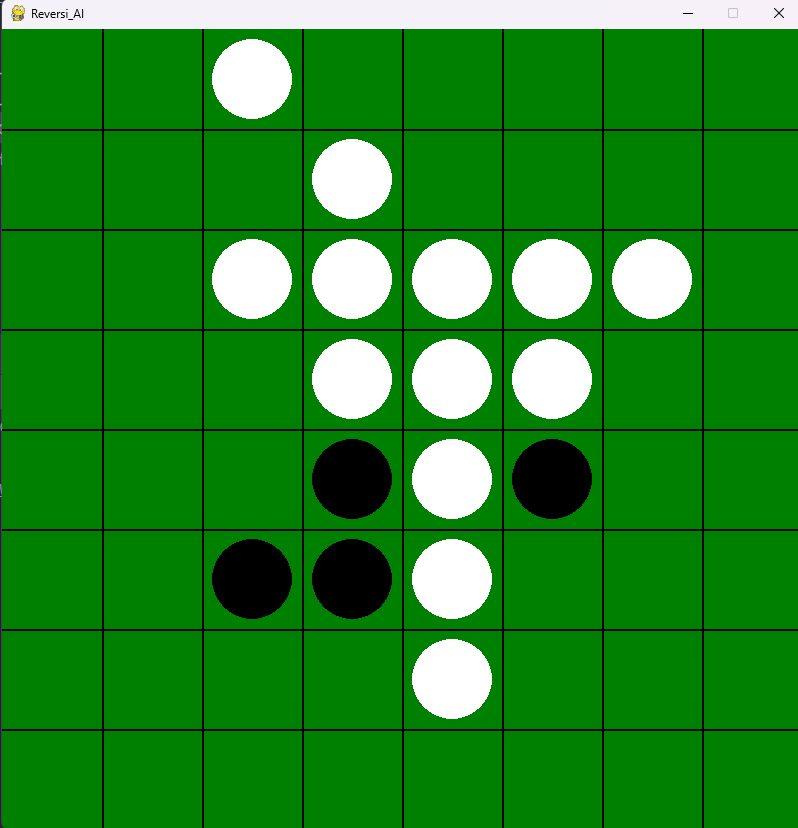
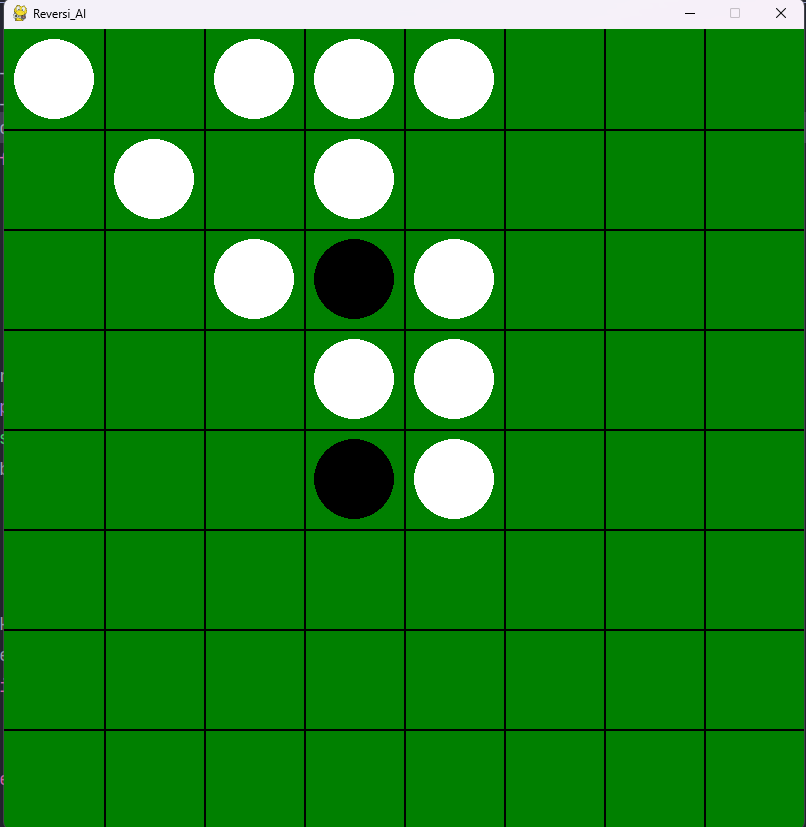
2) נוסיף משקל שונה למיקומים אסטרטגים בלוח, פינה שווה הכי הרבה (מקסימום שליטה, אי אפשר להפוך), קיר שווה גם יותר (מיקום טוב וקשה להפוך),  
משבצת פנימית (דרכה ניתן להגיע לקיר , חשוב בעיקר בתחחילת המשחק).

גם פה , מצב שלי פחות מצב היריב.

3) משבצת "מסוכנת" משבצת שמאפשרת ליריב לתפוס בעתיד פינה ולכן מורידה את הציון שהמצב מקבל עבורי  
4) הגבלת מספר התנועות שהיריב יכול לעשות (mobility)   
  
5) שילובים ושינויים שונים עם משקלים שונים לאלמנטים הביאו אותי לבחירה ב-  
**board\_heuristic.**   
היא נותת משקל גדול לפינה משקל שלילי כבד יחסית למשבצת מסוכנות ומשקל בינוני גם לקיר. סוכמת את ציון הדיסקיות שלי פחות ציון הדיסקיות של היריב. מנרמלת על ידי הפחתה של הציון המינימלי וחלוקה בטווח התוצאות.

MAX\_SCORE = 476 עם המשקלים שנתתי בסוף  
לכן MIN\_SCORE = -476 בעצם מצב שבו כל הלוח שלי\היריב.

חישוב הנירמול 



דוגמאות:

6 לבנות פחות אחד שחור 7 דיסקיות לבנות 12 לבנות   
 ואז נרמול 2 שחורות 4 שחורות   
 סה"כ 0.505252 3 לבנות קיר (10) 1 לבנות קיר(10)

1 לבן פינה (50) 0.5189075  
 0.589285

א) כן כשאר חוזרים על ההרצה מתקבלת אותה התוצאה. שום דבר לא משתנה בין המשחקים. השחקן הראשון (לבן) מנתח את המצב ההתחלתי שהווא אותו הדבר בכל משחק ותמיד ייבחר את אותו המהלך, לכן השחקן השני תמיד בוחן את אותו המצב וכן הלאה.

ב) כפי שרשמתי , רשמתי כמה וכמה פונקציות יוריסטיקה והרצי אותן אחד מול השני וראיתי מי מנצח הכי הרבה במצבים שונים של הלוח ומהמצב ההתחלתי.

ג) ניתן לראות את הניסיון של changing\_heuristic שבוחרת יוריסטיקה ספציפית  
למשל שמתמקדת בלשחק בפנים עבור 70% מהמשחק, ובסוף המשחק מתמקדת בלהגדיל את מספר הדיסקיות.

**שאלה 3**

א) כן כשאר חוזרים על ההרצה מתקבלת אותה התוצאה. שום דבר לא משתנה בין המשחקים. השחקן הראשון (לבן) מנתח הפעם שני צעדים קדימה את המצב ההתחלתי שהווא אותו הדבר בכל משחק ותמיד ייבחר את אותו המהלך, לכן השחקן השני תמיד בוחן את אותו המצב וכן הלאה.

ב) אם בכל מהלך(יש שישים) בוחנים לכל אחד מהמהלכים (כתוב להניח 10) את כל המהלכים שיש ליריב (שוב 10) אז בעצם מסתכלים על 10\*10 =100 מהלכים במהלך המשחק. לכן סה"כ 60\*100 = 6000.

ג) אם בכל מהלך נסתכל קדימה עד סוף המשחק אז למהלך הראשון לבד היה בערך עשר בחזקת 60 מהלכים לראות ואז לשני 10 בחזקת 59 וכו וכו

כלומר

**הסבר על הקבצים הנוספים**

יש בנוסף גרסא שבה משחקים מול הAI בעזרת GUI בקובץ בשם main.py  
בתוך תיקיית עזרים בשם reversi יש:

constants עבור קבועים  
helpers עבור פונקציות עזר   
board לייצוג ופעולות לוח  
piece לייצוג ופעולות על דיסקית

\_\_init\_\_ בשביל לעשות מהתיקייה import