­תרגיל מס׳ 2: פתרון בעיית N-Puzzle בעזרת אלגוריתם A\*

מצורפים הקבצים הבאים:

search.py – מממש את אלגוריתם החיפוש.

frontier.py – מממש ערמה.

state.py – מממש מצב של N-Puzzle בגודל nXn. הפונקציה ההיריסטית מחזירה 0 לכל מצב.

עליכם לעבור על הקבצים המצורפים, להבין אותם ולבצע את המשימות הבאות:

1. שנו את מבנה הנתונים כך שישמור את מספר המצבים הכולל שאוחסנו במבנה ואת מספר המצבים המקסימלי שאוחסנו בו-זמנית במבנה.
2. הריצו search(3) ו-search(4), מה מסקנתכם?
   * הריצות של search(4) לא מסתיימים, בגלל עומס החישוב.
3. שנו את הפונקציה ההיריסטית כך שתחזיר את מספר האריחים שאינם במקומם.
4. הריצו שנית את search(3) ו-search(4), מה מסקנתכם?
   * עכשיו, הsearch(4) כן מסתיים, אבל זה עדיין לוקח כמה זמן למצוא פטרון. וגם שיפר את המהירות של search(3).
5. שנו את הפונקציה ההיריסטית כך שתחזיר את סכום מרחקי מנהטן של האריחים ממקומם במצב המטרה.
6. הריצו שוב את search(3) ו-search(4), מה מסקנתכם?
   * בשני המצבים (3 ו4), הקוד מריץ מאוד יותר מהירה. עכשיו שמתמשים עםheuristic טובה, A\* הוא לא רק אופטמלי, אבל גם מהירה. זה בגלל שA\* יכול לצמצים את מרחב החיפוש, ותעדף נתיבים שסביר שיגיע להמטרה.