מבוא למדעי המחשב 67101 – סמסטר א' 2024

תרגיל 4 – Battleships

להגשה בתאריך **07/02/2024** בשעה 22:00

משחק צוללות (Battleships)

בתרגיל זה אתן תממשו וריאציה של המשחק "צוללות". משחק זה מיועד לשני שחקנים, לכל שחקן יש לוח עליו הוא ממקם בתחילת המשחק צוללות, והוא מנסה להטביע את הצוללות בלוח של היריב ע"י שליחת פצצה אל קואורדינטה כלשהי בלוח המשחק. כל שחקן משחק בתורו (לסירוגין).

מטרת המשחק היא להטביע את כל הצוללות של היריב, והשחקן שעושה זאת ראשון מנצח.

ניתן לקרוא עוד על המשחק באתר ויקיפדיה: צוללות (משחק) – ויקיפדיה

בשונה מהמשחק הקיים, במימוש שלנו יתקיימו התנאים הבאים:

1. את הצוללות ניתן להציב במאונך בלבד.
2. לצוללות מותר להיות צמודות לצוללות אחרות.
3. המשחק ישוחק בין שחקן אנושי, לבין המחשב (שיקרא גם "היריב". בחירת הפעולות עבור המחשב מומשו עבורכם).

כמו במשחק הקיים – לצוללות אסור לחפוף אחת לשנייה (משמע, צוללת אחת לא יכולה לחלוק משבצת עם צוללת אחרת).

אנו ממליצים מאוד לקרוא את כל הוראות התרגיל המפורטות במסמך זה לפחות פעם אחת לפני שאתן ניגשות לפתרון!

בנוסף, הקפידו על כתיבת קוד קריא על פי חוקי ה- **Coding Style** שהוצגו בקורס!

מילוי הלוח:

בחלק זה עליכן לכתוב קוד שימקם באופן אינטראקטיבי (באמצעות קלט מהמשתמש) את הצוללות על לוח המשחק של השחקן האנושי.

כדי להקל על התהליך, מסופק לכן קובץ עזר - **helper.py** המכיל קבועים שונים ופונקציות עזר שונות.

עליכן לייבא (באמצעות import) את הקובץ לתוך הקובץ **battleship.py** כדי שתוכלו להשתמש בקבועים ובפונקציות המוגדרות בו.

שימו לב - עליכן להשתמש ישירות בקבועים המוגדרים בקובץ זה ולא להעתיק אותם לקוד שלכן!

מותר להגדיר קבועים נוספים אך עליהם להימצא בקובץ **battleship.py** בלבד ועליכן להשתמש בקבועים המסופקים לכן בקובץ **helper.py** במקומות הנדרשים לכך.

קובץ העזר - פירוט

הקבועים המוגדרים בקובץ **helper.py** (שעליכן להשתמש בהם - ישנם עוד קבועים שאין צורך להשתמש בהם) הם:

1. NUM_ROWS: קבוע שהוא מספר שלם המייצג את מספר השורות בלוח המשחק.
2. NUM_COLUMNS: קבוע שהוא מספר שלם המייצג את מספר העמודות בלוח המשחק.
3. SHIP_SIZES: קבוע מסוג tuple המכיל את גדלי הצוללות (שימו לב שיכולות להיות צוללות שונות באותו גודל).
4. WATER: קבוע המייצג תא בלוח המשחק שיש בו מים (= תא ללא צוללת = תא ריק. כלומר – תא שיש בו מים זה כמו לומר תא ריק.)
5. SHIP: קבוע המייצג תא בלוח שיש בו צוללת.
6. HIT_WATER: קבוע המייצג תא בלוח המשחק שהכיל מים ונפגע.
7. HIT_SHIP: קבוע המייצג תא בלוח המשחק שהכיל צוללת ונפגע.

הפונקציות המוגדרות בקובץ **helper.py** (שעליכן להשתמש בהן - ישנן עוד פונקציות שאין צורך להשתמש בהן) הן:

1. **print_board(board1, board2=None):**

פונקציה זו מקבלת לוח אחד או שני לוחות ומדפיסה אותם על המסך.
אם מסופק רק לוח אחד - הפונקציה מדפיסה רק אותו. אם מסופקים שני לוחות - הפונקציה תדפיס את שניהם.
שימו לב - הפונקציה תדפיס את הלוח במלואו כפי שהוא מתקבל בפונקציה.

אם ברצונכם שהלוח יודפס בצורה אחרת (למשל – להסתיר את הספינות שעוד לא נפגעו), יש להכין לוח מתאים.

2. **get_input(msg):**

פונקציה זו מקבלת הודעה להדפסה, מדפיסה הודעה זו ומחכה לקבלת קלט מהמשתמש - שימו לב להשתמש אך ורק בפונקציה זו כשאתן מבקשות קלט מהמשתמש בתרגיל!

3. **is_int(s):**

פונקציה זו מקבלת מחרוזת ובודקת שניתן להמיר אותה ל-int. אם ניתן - הפונקציה מחזירה True. אם לא ניתן - הפונקציה מחזירה False. (ניתן, במקום, להשתמש במתודה isdigit של מחרוזות)

4. **choose_ship_location(board, size, locations):**

פונקציה זו מקבלת לוח משחק, גודל של צוללת וקבוצת מיקומים אפשריים חוקיים (מהצורה - **(row_index,column_index)**) להצבת ראש הצוללת, ומחזירה מיקום אחד להצבת הצוללת.
מיקום חוקי יהיה כל מיקום בגבולות הלוח כך שכל חלקי הצוללת במים פנויים (כלומר מכיל את הערך WATER) ובגבולות הלוח.
הפונקציה יוצאת מנקודת הנחה שכל הערכים הנשלחים אליה חוקיים, שליחת מיקום לא חוקי (או בפורמט לא מתאים) יכול לגרום לתוצאות לא חוקיות או אפילו לקריסת התוכנית.

5. **choose_torpedo_target(board, locations):**

פונקציה זו מקבלת לוח (עם ספינות מוסתרות) וקבוצת מיקומים אפשריים (מהצורה – **(row_index,column_index)**) לשם שליחת פצצה (טורפדו) לצוללת של היריב ומחזירה מיקום לשליחת טורפדו.

<u>מימוש המשחק</u>

לוחות המשחק מיוצגים ע"י רשימה דו מימדית.

כל תא ברשימה דו מימדית (לאחר אתחול הרשימה) מייצג אלמנט כלשהו בלוח:

- מיקום של צוללת מיוצג ע"י הקבוע SHIP (מיוצג עם צבע חום בלוח).
- מיקום של פגיעה מיוצג ע"י הקבוע HIT_SHIP (מיוצג עם צבע אדום בלוח).
- מיקום של החטאה מיוצג ע"י הקבוע HIT_WATER (מיוצג עם צבע טורקיז בלוח).
- מיקום ריק (ללא צוללת) בלוח של השחקן (הלוח שלנו), או מיקום לא ידוע בלוח של היריב (המחשב) מיוצג ע"י הקבוע WATER (מיוצג עם צבע כחול בלוח).

<u>הערות כלליות:</u>

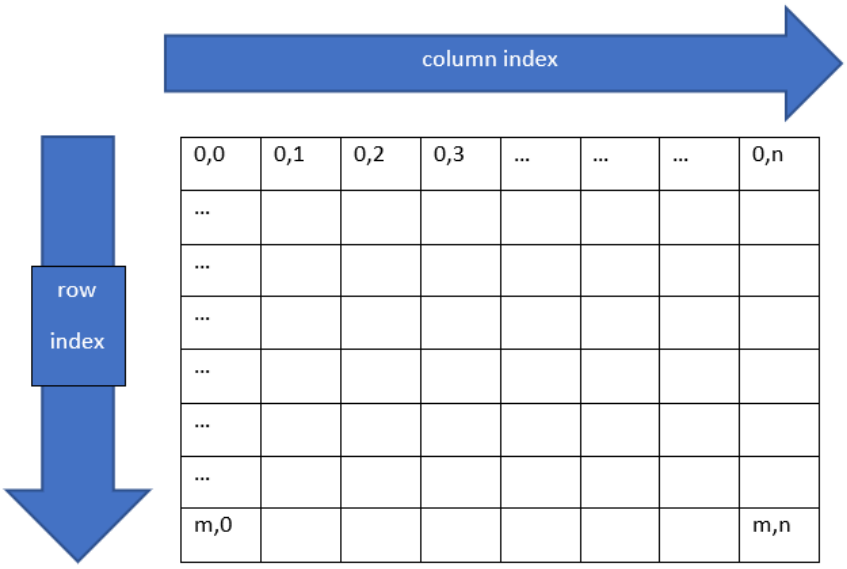
1. בכל מקום בו מצויין "לוח המשחק" (או לוח) הכוונה היא לרשימה דו מימדית מלבנית המייצגת את לוח המשחק.
2. בכל מקום בו מצויין "מיקום" או "קואורדינטה" הכוונה היא ל-tuple המכיל זוג מספרים שלמים המהווים אינדקס של שורה ואינדקס של עמודה (בהתאמה) ברשימה דו מימדית המייצגת את הלוח:

   (row_index,column_index)

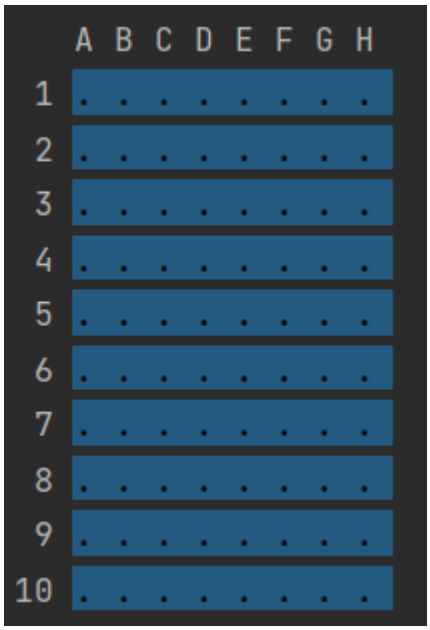
   (שימו לב שהאינדקס של השורה יבוא <u>לפני</u> האינדקס של העמודה).

   לדוגמא: (5,3) הוא התא הנמצא בשורה השישית ובעמודה הרביעית (מפני שאינדקסים בפייתון מתחילים מ-0, 5 הוא האינדקס הראשון ולכן מייצג את השורה ה5+1, ובאופן דומה 3 הוא האינדקס השני ולכן מייצג את העמודה ה3+1).
3. ייצוג הצבעים הוא פנימי לקובץ העזר, אין להשתמש/להוסיף/לשנות אותו.

4. דוגמא לייצוג של רשימה דו מימדית בעלת $m+1$ שורות ו-$n+1$ עמודות:



5. בעוד אנחנו עובדים עם רשימה דו מימדית בעלת אינדקסים מספריים, לוח המשחק החשוף לשחקן מייצג מיקום ע"י אינדקס מספרי לשורות, ואינדקס לקסיקלי (מבוסס אותיות) עבור העמודות.



הוראות:

1. ממשו את הפונקציה:

init_board(rows, columns)

הפונקציה צריכה לקבל שני פרמטרים:

a. rows - מספר השורות בלוח.

b. columns - מספר העמודות בלוח.

על הפונקציה ליצור לוח ריק בגודל השורות כפול העמודות עבור השחקן או המחשב - לוח המלא כולו באותיות המסמלות מיקום ריק. הפונקציה תחזיר את הלוח.

שימו לב :הרשימה המוחזרת היא רשימה של רשימות בעלות ערכים זהים, אבל יש לוודא בעת היצירה שלה שהרשימות הן רשימות שונות ((כלומר ששינוי רשימה אחת לא יגרור שינוי ברשימה אחרת).

2. ממשו את הפונקציה:

cell_loc(name)

הפונקציה מקבלת פרמטר יחיד בשם name, שמייצג קלט טקסטואלי מהמשתמש המתאר מיקום כלשהו בלוח ומחזירה tuple המתאר קואורדינטה מספרית.

הקלט שהשחקן מספק יהיה מהפורמט XN כאשר X מתארת עמודה כלשהי המיוצגת ע"י אות אנגלית גדולה, ו-N מתאר את מספר השורה (כשמתחילים ב- 1).

בעוד כקלט מהמשתמש אות קטנה ומספר יהיו ערכים חוקיים, הפונקציה צריכה לקבל רק אות גדולה ומספר כפרמטר חוקי.

שימו לב! הקלט הפוך מייצוג קואורדינטה (בקלט הטקסטואלי מופיעה קודם העמודה ואז השורה, ואילו בייצוג של קואורדינטה מופיע קודם האינדקס של השורה ואז האינדקס של העמודה).

לדוגמא: אם המשתמש סיפק את הקלט "**A2**" הפונקציה תחזיר את הערך (1,0) שכן העמודה **A** היא העמודה הראשונה (כלומר באינדקס 0) ושורה **2** היא השורה באינדקס 1.

- ניתן להניח שהערך שהתקבל הוא מסוג מחרוזת **ובפורמט החוקי** (כלומר תו ולאחריו מספר שלם).
- ניתן להניח שיהיו לכל היותר 26 עמודות.
- הפרמטר יכול לתאר קואורדינטה כללית (בטווח של 26 עמודות, ללא הגבלה על מספר השורות, וללא קשר ללוח המשחק הנוכחי)
  - למשל: ייתכן כי הלוח שלנו בגודל 5×5 ונקבל כפרמטר חוקי את "**F2**" שמתרגמת ל קואורדינטה (1,5).

על מנת לבצע את ההמרה בין הייצוג הטקסטואלי של אות לייצוג המספרי שלה ניתן להשתמש בפונקציה ord (ניתן לקרוא עוד על הפונקציה ord כאן).

הפונקציה ord מקבלת מחרוזת באורך 1 המייצגת אות ומחזירה ערך מספרי עבור האות הזו לפי ייצוג ASCII.

למשל הקוד:

ord("A")

יחזיר את הערך 65.

למשל הקוד:

ord("D")

יחזיר את הערך 68. על מנת לבצע המרה חזרה מערך ASCII מספרי לייצוג טקסטואלי ניתן להשתמש בפונקציה chr (ניתן לקרוא עוד על הפונקציה chr באן).

למשל הקוד:

char(68)

יחזיר את הערך "D".

3. ממשו את הפונקציה:

valid_ship(board, size, loc)

הפונקציה צריכה לקבל שלושה פרמטרים:

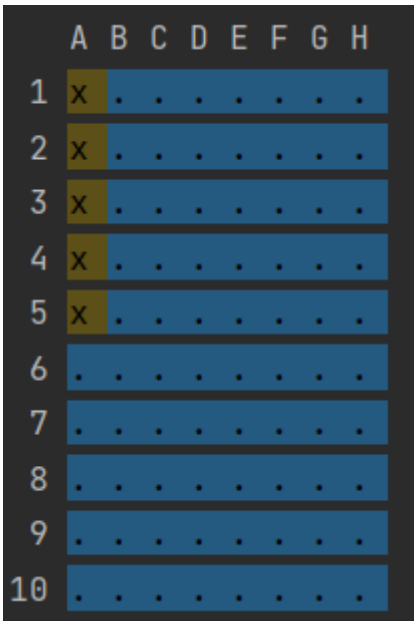
a. board – לוח משחק
b. size - גודל ספינה
c. loc – משתנה מסוג tuple שמייצג מיקום בלוח (זוג מספרים שלמים) מהצורה – **(row_index,column_index)** כאשר הראשון הוא האינדקס של השורה והשני של העמודה.

הפונקציה תבדוק האם צוללת בגודל size שמתחילה בקואורדינטה (מהצורה - **(row_index,column_index)**) יכולה להיכנס ללוח ללא התנגשות בקצות הלוח או בצוללת אחרת.

הפונקציה תחזיר True אם הצוללת יכולה להיכנס ללוח באופן חוקי, ו- False אחרת.

שימו לב שהצוללת מוכנסת במאונך, כאשר הקואורדינטה שהתקבלה מתארת את המקום הגבוה ביותר עבורה בלוח.

לדוגמא, ניתן להכניס צוללת בגודל 5 בקואורדינטה 1A באופן הבא:



שימו לב שלא נוכל להכניס עוד צוללת בקואורדינטה 2A כיוון שזו תתנגש עם הצוללת הראשונה שהוכנסה.

לא נוכל גם להכניס צוללת בגודל 5 בקואורדינטה 9A בלוח זה, כיוון שזו תגרום ליציאה מקצות הלוח.

כמו כן, שימו לב שפונקציה זו לא מבצעת שינויים בלוח, רק מבצעת בדיקה.

4. ממשו את הפונקציה:

create_player_board(rows, columns, ship_sizes)

הפונקציה צריכה לקבל שלושה פרמטרים:

a. rows – מספר השורות בלוח.

b. columns - מספר העמודות בלוח.

c. ship_sizes – רשימה או טופל (tuple) של מספרים שלמים שמתארים את אורכי הצוללות שיופיעו בלוח.

על הפונקציה לייצר לוח ריק חדש עבור השחקן, ולבקש מהמשתמש את הקואורדינטות בהן הוא מעוניין להניח צוללות. סדר גדלי הצוללות הוא לפי הסדר של ship_sizes.

כל עוד נותרו ספינות שצריך למקם, הפונקציה תדפיס את מצב הלוח הנוכחי (עם הפונקציה **print_board**) ולאחר מכן תבקש מהמשתמש (עם הפונקציה **get_input**) מיקום להנחת הספינה הבאה. יש להציג עם הבקשה הודעה מתאימה כדי שהשחקן יבין מה הוא אמור לעשות.

הקלט מהמשתמש יתקבל בעזרת הפונקציה **get_input** שמומשה עבורכם בקובץ **helper.py**, והפורמט בו הקלט צריך להתקבל הוא: **XN** כאשר X הוא אות המייצגת טור ו-N הוא מספר שלם המייצג שורה.

הצוללת תמוקם על הלוח כאשר הקואורדינטה שניתנה היא הקואורדינטה העליונה ביותר (ראו דוגמא).

אם אכן ניתן למקם את הצוללת על הלוח במיקום זה, עדכנו את הלוח והמשיכו להשמת הצוללת הבא.

אם לא, הדפיסו שוב את הלוח, ובקשו קלט נוסף מהמשתמש עבור צוללת זו עם הודעה אינפורמטיבית

(שיכולה להיות כל דבר, העיקר שתסמן שהקלט אינו חוקי).

בסוף הפונקציה יש להחזיר את לוח המשחק.

<u>הערות:</u>

a. ניתן להניח שהקלט חוקי, ושניתן יהיה להכניס את הספינות ללוח.

b. המשחק אמנם לא יהיה מעניין, אך גם רשימה ריקה היא קלט חוקי.

<u>דוגמאות:</u>

הכנסה חוקית:

```
   A B C D E F G H
 1 . . . . . . . .
 2 . . . . . . . .
 3 . . . . . . . .
 4 . . . . . . . .
 5 . . . . . . . .
 6 . . . . . . . .
 7 . . . . . . . .
 8 . . . . . . . .
 9 . . . . . . . .
10 . . . . . . . .
enter top coordinate for ship of size 5: D3
   A B C D E F G H
 1 . . . . . . . .
 2 . . . . . . . .
 3 . . . x . . . .
 4 . . . x . . . .
 5 . . . x . . . .
 6 . . . x . . . .
 7 . . . x . . . .
 8 . . . . . . . .
 9 . . . . . . . .
10 . . . . . . . .
enter top coordinate for ship of size 5:
```

הכנסה לא חוקית (קלט הגורם להתנגשות עם צוללת אחרת, ועל כן הלוח לא השתנה):

```
enter top coordinate for ship of size 5: D3
   A B C D E F G H
 1 . . . . . . . .
 2 . . . . . . . .
 3 . . . x . . . .
 4 . . . x . . . .
 5 . . . x . . . .
 6 . . . x . . . .
 7 . . . x . . . .
 8 . . . . . . . .
 9 . . . . . . . .
10 . . . . . . . .
enter top coordinate for ship of size 5: D4
not a valid location
   A B C D E F G H
 1 . . . . . . . .
 2 . . . . . . . .
 3 . . . x . . . .
 4 . . . x . . . .
 5 . . . x . . . .
 6 . . . x . . . .
 7 . . . x . . . .
 8 . . . . . . . .
 9 . . . . . . . .
10 . . . . . . . .
enter top coordinate for ship of size 5:
```

הכנסה לא חוקית (קלט שלא נמצא בפורמט, על כן הלוח לא ישתנה):

```
enter top coordinate for ship of size 5: D3
   A B C D E F G H
 1 . . . . . . . .
 2 . . . . . . . .
 3 . . . x . . . .
 4 . . . x . . . .
 5 . . . x . . . .
 6 . . . x . . . .
 7 . . . x . . . .
 8 . . . . . . . .
 9 . . . . . . . .
10 . . . . . . . .
enter top coordinate for ship of size 5: D1o
not a valid location
   A B C D E F G H
 1 . . . . . . . .
 2 . . . . . . . .
 3 . . . x . . . .
 4 . . . x . . . .
 5 . . . x . . . .
 6 . . . x . . . .
 7 . . . x . . . .
 8 . . . . . . . .
 9 . . . . . . . .
10 . . . . . . . .
```

הכנסה לא חוקית (מיקום לא חוקי):

```
enter top coordinate for ship of size 5: J0
not a valid location
   A B C D E F G H
 1 . . . . . . . .
 2 . . . . . . . .
 3 . . . x . . . .
 4 . . . x . . . .
 5 . . . x . . . .
 6 . . . x . . . .
 7 . . . x . . . .
 8 . . . . . . . .
 9 . . . . . . . .
10 . . . . . . . .
```

5. ממשו את הפונקציה:

fire_torpedo(board, loc)

הפונקציה צריכה לקבל שני פרמטרים:

a. board – לוח המשחק.

b. loc – משתנה מסוג tuple שמייצג את המיקום בלוח (באשר השורה מיוצגת ראשונה, כמו בשאר הtupleים בתרגיל). הפונקציה תבצע הפצצה על קואורדינטה מסוימת בלוח.

אם הפגיעה הייתה במים, הפונקציה תחליף בלוח את מיקום ההפצצה בלוח לקבוע המייצג מים חשופים.

אם הפגיעה הייתה בצוללת, הפונקציה תחליף בלוח את מיקום ההפצצה לקבוע המייצג פגיעה.

לאחר ביצוע השינוי הפונקציה תחזיר את הלוח.

אם התקבל מיקום שאינו חוקי, הפונקציה לא תבצע שינוי אך עדיין תחזיר את הלוח (ניתן להניח שהתקבל לוח חוקי).

6. ממשו את הפונקציה:

main()

בפונקציה זו אתם תממשו את המשחק עבור שני שחקנים, כאשר השחקן הראשון הוא שחקן אנושי, והשחקן השני הוא שחקן יריב (המחשב).

עליכם לעקוב אחר סדר הפעולות הבא:

1. יצירת הלוח עבור השחקן באמצעות הפונקציה create_player_board (ופונקציות נוספות אם יש צורך) ויצירת הלוח עבור המחשב. יש לבחור מיקומי ספינות של שחקן המחשב עם הפונקציה choose_ship_location בקובץ helper (ופונקציות נוספות אם יש צורך).

2. לאחר מכן נתחיל את המשחק עצמו, כאשר הפונקציה main אחראית על הפעלת התורים במשחק.

סיבוב אחד (מהלך שני תורים במשחק - אחד של השחקן ואחד של המחשב) מתבצע באופן הבא:

כל עוד המשחק לא הסתיים (מהלכו של סיבוב אחד):

1. הדפס את שני הלוחות בעזרת הפונקציה **print_board** שמומשה עבורכם בקובץ **helper.py**.

<u>הערות למימוש:</u>

הפונקציה מקבלת שני לוחות כפרמטרים כאשר הלוח שמתקבל בפרמטר הראשון הוא הלוח שהשחקן יצר שיודפס באופן מלא (נראה את מיקומי הצוללות) והלוח שמתקבל בפרמטר השני הוא לוח מוסתר של המחשב, בו הספינות החבויות יודפסו כמים

*הקוד שקורא לפונקציה אחראי לשלוח אליה לוח מוסתר.

2. בקשת קלט - תודפס הודעה למשתמש שתבקש ממנו להכניס מיקום לשליחת פצצת טורפדו.

(השתמשו בפונקציה **get_input** שמומשה עבורכם בקובץ **helper.py**).

2.1 אם הקואורדינטה שהתקבלה תקינה והיא מסמנת תא בלוח המחשב שעדיין חבוי נשמור את הקורדיאנטה בצד שאליה ישוגר הטורפדו בצד ~~ונעדכן את לוח המחשב בהתאם~~.

2.2 אחרת, הקואורדינטה שהתקבלה אינה תקינה, נדפיס הודעה אינפורמטיבית (שיכולה להיות כל דבר, העיקר שתצביע על כך שהקלט לא תקין) ונחזור ל1.

<u>הערות למימוש:</u>

a. ניתן לקבל אותיות קטנות בקלט תקין. בשביל להמיר אותיות קטנות לגדולות ניתן להשתמש בפונקציה upper של string (עוד מידע על הפונקציה upper בלינק).

3. המחשב בוחר מהלך לטורפדו בעזרת הפונקציה **choose_torpedo_target** בקובץ helper.py.
4. ביצוע הירי במקביל ועדכון הלוחות של המשתמש ושל המחשב.
5. במקרה ולאחר פגיעה הושמד הצי של אחד השחקנים (או שניהם!), יודפסו שני הלוחות בצורה גלויה, ולאחר מכן שאלה אם לשחק שוב יחד עם הודעה על תוצאת המשחק שהסתיימה (ניצחון למחשב/לשחקן או שניהם).
   a. הקלט החוקי במקרה זה הוא "Y" או "N".
   b. אם מתקבל קלט לא חוקי, יש להמשיך לשאול (עם הודעה מתאימה), אך ללא הדפסת הלוחות.
6. אחרת, נחזור ל-1.

<u>דוגמאות:</u>

1.<u>תחילת המשחק:</u>

בדוגמה הבאה אנו רואים את שני הלוחות לאחר שמיקמנו את כל הצוללות של השחקן.

הלוח של השחקן - שלנו - הוא הלוח השמאלי וניתן לראות בו את הצוללות לאחר שמיקמנו את הצוללת האחרונה בגודל 2 במיקום F1.

הלוח הימני הוא הלוח של היריב - המחשב - והצוללות בו לא נראות לנו:

```
enter top coordinate for ship of size 2: F1
   A B C D E F G H          A B C D E F G H
 1 x x x x x x . .        1 . . . . . . . .
 2 x x x x x x . .        2 . . . . . . . .
 3 x x x x . . . .        3 . . . . . . . .
 4 x x . . . . . .        4 . . . . . . . .
 5 x x . . . . . .        5 . . . . . . . .
 6 . . . . . . . .        6 . . . . . . . .
 7 . . . . . . . .        7 . . . . . . . .
 8 . . . . . . . .        8 . . . . . . . .
 9 . . . . . . . .        9 . . . . . . . .
10 . . . . . . . .       10 . . . . . . . .
```

<u>דוגמא לקלט חוקי:</u>

הקלט שהכנסנו הוא D5 ולכן אנחנו מנסים לפגוע בצוללת שנמצאת בקואורדינטה (4,3).

הפגיעה לא הצליחה כי לא הייתה שם צוללת של היריב.

המחשב בחר בקוארדינטה G1 שגם בה אין צוללת.

```
Choose target: D5
   A B C D E F G H          A B C D E F G H
 1 x x x x x x o .        1 . . . . . . . .
 2 x x x x x x . .        2 . . . . . . . .
 3 x x x x . . . .        3 . . . . . . . .
 4 x x . . . . . .        4 . . . . . . . .
 5 x x . . . . . .        5 . . . o . . . .
 6 . . . . . . . .        6 . . . . . . . .
 7 . . . . . . . .        7 . . . . . . . .
 8 . . . . . . . .        8 . . . . . . . .
 9 . . . . . . . .        9 . . . . . . . .
10 . . . . . . . .       10 . . . . . . . .
```

דוגמא לקלט לא חוקי (הכנסנו בתור קלט את המחרוזת B90 שהוא קלט לא חוקי וקיבלנו הדפסה של invalid target):

```
Choose target: B90
   A B C D E F G H          A B C D E F G H
 1 x x x x x x o .        1 . . . . . . . .
 2 x x x x x x . .        2 . . . . . . . .
 3 x x x x . . . .        3 . . . . . . . .
 4 x x . . . . . .        4 . . . . . . . .
 5 x x . . . . . .        5 . . . o . . . .
 6 . . . . . . . .        6 . . . . . . . .
 7 . . . . . . . .        7 . . . . . . . .
 8 . . . . . . . .        8 . . . . . . . .
 9 . . . . . . . .        9 . . . . . . . .
10 . . . . . . . .       10 . . . . . . . .
invalid target
Choose target:
```

הוראות הגשה

- עליכן להגיש את הקובץ ex4.zip (בלבד) בקישור ההגשה של תרגיל 4 דרך אתר הקורס (moodle). ההגשה היא עד המועד הנקוב בראש הקובץ.
- ex4.zip צריך לכלול את הקובץ הבא בלבד:
  - battleship.py
- שימו לב שאין להגיש את הקובץ **helper.py**, ושהבדיקות יתבצעו עם קובץ עזר בעל ערכים שונים.

הנחיות כלליות בנוגע להגשה

- הנכם רשאים להגיש תרגילים דרך מערכת ההגשות באתר הקורס מספר רב של פעמים במועד ההגשה המקורי. ההגשה האחרונה בלבד היא זו שקובעת ושתיבדק.
- לאחר הגשת התרגיל, ניתן ומומלץ להוריד את התרגיל המוגש ולוודא כי הקבצים המוגשים הם אלו שהתכוונתם להגיש וכי הקוד עובד על פי ציפיותיכם.
  - באחריותכם לוודא כי – **PDF** הבדיקות נראה כמו שצריך.
- קראו היטב את קובץ נהלי הקורס לגבי הנחיות נוספות להגשת התרגילים.
- **הגשות מחדש** - ניתן יהיה להגיש את התרגיל מחדש, לאחר מועד ההגשה המקורי, פעם אחת בלבד על מנת לתקן את הציון, כל עוד ההגשה היא לפני סיום הסמסטר.

בהצלחה!