תרגיל בית 3 – MDP ומבוא ללמידה

עברו על כלל ההנחיות לפני תחילת התרגיל.

הנחיות כלליות:

- תאריך ההגשה: לחלק א' של התרגיל (MDP) – עד ליום האחרון של הסמסטר - **08/04/2024** ב23:59
  לחלק ב' של התרגיל (מבוא ללמידה) – עד לסוף מועדי א' - **17/05/2024** ב23:59
- את המטלה יש להגיש **בזוגות בלבד.**
- יש להגיש מטלות מוקלדות בלבד. פתרונות בכתב יד לא ייבדקו.
- ניתן לשלוח שאלות בנוגע לתרגיל בפיאצה בלבד.
- המתרגל האחראי על תרגיל זה: **דניאל אלגריסי.**
- בקשות דחיה מוצדקות (מילואים, אשפוז וכו') יש לשלוח למתרגל האחראי (**ספיר טובול**) בלבד.
- במהלך התרגיל ייתכן שנעלה עדכונים, למסמך הנ"ל – תפורסם הודעה בהתאם.
- העדכונים הינם מחייבים, ועליכם להתעדכן עד מועד הגשת התרגיל.
- שימו לב, העתקות תטופלנה בחומרה.
- התשובות לסעיפים בהם מופיע הסימון ✍️ צריכים להופיע בדוח.
- לחלק הרטוב מסופק שלד של הקוד.
- אנחנו קשובים לפניות שלכם במהלך התרגיל ומעדכנים את המסמך הזה בהתאם. גרסאות עדכניות של המסמך יועלו לאתר. הבהרות ועדכונים שנוספים אחרי הפרסום הראשוני יסומנו כאן בצהוב. ייתכן שתפורסמנה גרסאות רבות – אל תיבהלו מכך. השינויים בכל גרסה יכולים להיות קטנים.

שימו לב שאתם משתמשים רק בספריות הפייתון המאושרות בתרגיל (מצוינות בתחילת כל חלק רטוב)
לא יתקבל קוד עם ספריות נוספות

מומלץ לחזור על שקפי ההרצאות והתרגולים הרלוונטיים לפני תחילת העבודה על התרגיל.

חלק א' – MDP (44 נק')

רקע

בחלק זה נעסוק בתהליכי החלטה מרקובים, נתעניין בתהליך עם **אופק אינסופי** (מדיניות סטציונרית).

חלק א' - חלק היבש ✍️

1. בתרגול ראינו את משוואת בלמן כאשר התגמול ניתן עבור המצב הנוכחי בלבד, כלומר $R: S \to \mathbb{R}$, למתן תגמול זה נקרא "תגמול על הצמתים" מכיוון שהוא תלוי בצומת שהסוכן נמצא בו.
בהתאם להגדרה זו הצגנו בתרגול את האלגוריתמים Value iteration ו-Policy Iteration למציאת המדיניות האופטימלית.

כעת, נרחיב את ההגדרה הזו, לתגמול המקבל את המצב הנוכחי והמצב אליו הגיע הסוכן, כלומר: $R: S \times S \to \mathbb{R}$, למתן תגמול זה נקרא "תגמול תוצאתי". לצורך שלמות ההגדרה, נגדיר שאם לכל $a \in A$ מתקיים - $P(s'|s,a) = 0$ אז $R(s,s') = -\infty$.

א. (1 נק') התאימו את הנוסחה של התוחלת של התועלת מהתרגול, עבור התוחלת של התועלת המתקבלת במקרה של " תגמול תוצאתי ", אין צורך לנמק.

$$U(s) = E[\sum_{t=0}^{inf} \gamma^t R(s_t, s_t')]$$

ב. (1 נק') כתבו מחדש את נוסחת משוואת בלמן עבור המקרה של " תגמול תוצאתי ", אין צורך לנמק.

$$U^{\pi}(s) = \sum_{s'} P(s'|s,\pi(s))[R(s_t, s_t') + \gamma^t U(s')]$$

בסעיפים הבאים התייחסו גם למקרה בו $\gamma = 1$, והסבירו מה לדעתכם התנאים שצריכים להתקיים על הסביבה\mdp על מנת שתמיד נצליח למצוא את המדיניות האופטימלית.

ג. (2 נק') נסחו את אלגוריתם Value Iteration עבור המקרה של " תגמול תוצאתי ".

נשים לב כי אם גאמא שווה 1, אז תנאי העצירה יהיה רק כאשר דלתא שווה ממש ל0, כלומר כאשר אין הפרש בכלל בין 2 איטרציות עוקבות, ואנו מתכנסים למדיניות האופטימלית.

$local\ variables: U, U' - utility\ vectors,$

$\delta = 0 - \max change\ in\ the\ utility\ of\ any\ state\ in\ any\ iteration$

$repeat:$

$U \leftarrow U', \delta \leftarrow 0$

$for\ each\ state\ s\ in\ S\ do:$

$U'[s] \leftarrow \max_{a \in A(s)} \sum_{s'} P(s'|s,a)[R(s,s') + \gamma U(s')]$

$\delta \leftarrow \max(\delta, |U'(s) - U(s)|)$

$until\ \delta < \frac{\varepsilon(1-\gamma)}{\gamma}\ or\ (\delta = 0\ and\ \gamma = 1)$

$return\ U$

ד. (2 נק') נסחו את אלגוריתם Policy Iteration עבור המקרה של " תגמול תוצאתי ".

נשים לב כי במקרה של אלגוריתם זה, הערך של גאמא לא משפיע על תנאי העצירה, כלומר גם במקרה שבו הוא 1 וגם בכזה שהוא שונה מ1, האלגוריתם יפעל אותו הדבר ונתכנס למדיניות האופטימלית, שכן מרחב המצבים והפעולות סופי.

$local\ variables: U -\ utility\ vectors, \pi = 0 - policy$

$repeat:$

$U \leftarrow Policy - Evaluation(\pi, U, mdp)$

$unchanged \leftarrow true$

$for\ each\ state\ s\ in\ S\ do:$

$if\ \max_{a \in A(s)} \sum_{s'} P(s'|s,a)[R(s,s') + \gamma U(s')] >$

$$\sum_{s'} P(s'|s,\pi(s))[R(s,s') + \gamma U(s')]:$$

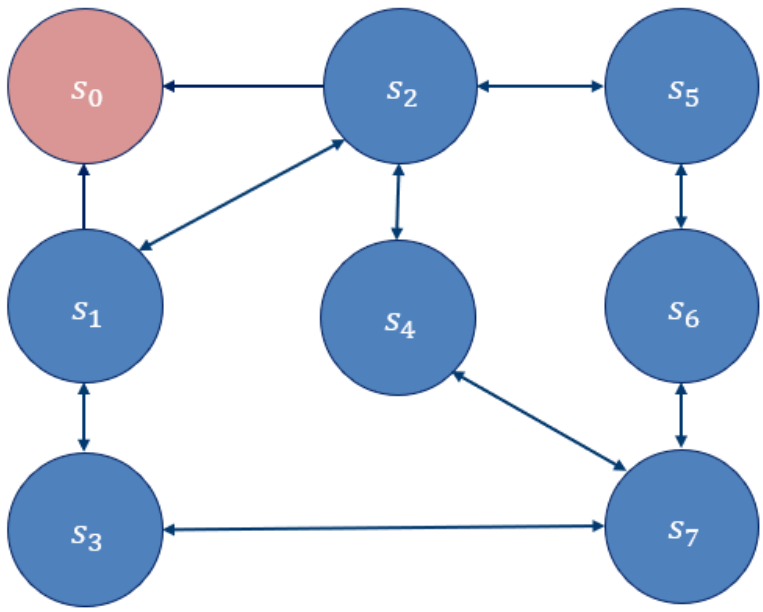
$\pi(s) \leftarrow \operatorname{argmax}_{a \in A(s)} \sum_{s'} P(s'|s,a)[R(s,s') + \gamma U(s')]$

$unchanged \leftarrow false$

$until\ unchanged$

$return\ \pi$

נתון הגרף הבא:



נתונים:

- $\gamma = 0.5$ (Discount factor).
- אופק אינסופי.
- $S = \{s_0, s_1, s_2, s_3, s_4, s_5, s_6, s_7\}$ – קבוצת המצבים – מתארים את מיקום הסוכן בגרף.
- $S_G = \{s_0\}$ – קבוצת המצבים הסופיים.
- קבוצת הפעולות לכל מצב (על פי הגרף), לדוגמא: $A(s_3) = \{\uparrow, \rightarrow\}$.
- תגמולים ("תגמול על פעולה "):

$$\forall s \in S, s' \in S \backslash S_G: \quad R(s, s') = -1, \qquad R(s_1, s_0) = 5, \qquad R(s_2, s_0) = 7$$

- מודל המעבר הוא דטרמיניסטי, כלומר כל פעולה מצליחה בהסתברות אחת.

ה. (יבש 2 נק') הרץ את האלגוריתם Value iteration שכתבת על הגרף הנתון. ומלא את הערכים בטבלה הבאה, כאשר $\forall s \in S \backslash S_G : U_0(s) = 0$. (ייתכן שלא צריך למלא את כולה).

	$U_0(s_i)$	$U_1(s_i)$	$U_2(s_i)$	$U_3(s_i)$	$U_4(s_i)$	$U_5(s_i)$	$U_6(s_i)$	$U_7(s_i)$	$U_8(s_i)$
s_1	0	5	5	5	5				
s_2	0	7	7	7	7				
s_3	0	-1	1.5	1.5	1.5				
s_4	0	-1	2.5	2.5	2.5				
s_5	0	-1	2.5	2.5	2.5				
s_6	0	-1	-1.5	0.25	0.25				
s_7	0	-1	-1.5	0.25	0.25				

ו. (יבש 2 נק') הרץ את האלגוריתם Policy iteration שכתבת על הגרף הנתון. ומלא את הערכים בטבלה הבאה, כאשר המדיניות ההתחלתית π_0 מופיעה בעמודה הראשונה בטבלה. (ייתכן שלא צריך למלא את כולה).

	$\pi_0(s_i)$	$\pi_1(s_i)$	$\pi_2(s_i)$	$\pi_3(s_i)$	$\pi_4(s_i)$	$\pi_5(s_i)$	$\pi_6(s_i)$	$\pi_7(s_i)$	$\pi_8(s_i)$
s_1	↓	↑	↑	↑	↑				
s_2	↓	←	←	←	←				
s_3	→	→	↑	↑	↑				
s_4	↑	↑	↑	↑	↑				
s_5	←	←	←	←	←				
s_6	↑	↑	↑	↑	↑				
s_7	↑	↑	↑	↖	↖				

ז. (יבש 2 נק') חיזרי על הסעיף הקודם. הפעם עם **אופק סופי כאשר** $\boldsymbol{N = 2}$ (שימי לב, המדיניות לא חייבת להסתיים במצב מסיים, ישנם מצבים שלא יכולים להגיע למצב מסיים עם אופק זה. ישנם צמתים עם מספר תשובות נכונות, נקבל את כולם).

	$\pi_0(s_i)$	$\pi_1(s_i)$	$\pi_2(s_i)$	$\pi_3(s_i)$	$\pi_4(s_i)$	$\pi_5(s_i)$	$\pi_6(s_i)$	$\pi_7(s_i)$	$\pi_8(s_i)$
s_1	↓	↑	↑						
s_2	↓	←	←						
s_3	→	→	↑						
s_4	↑	↑	↑						
s_5	←	←	←						
s_6	↑	↑	↑						
s_7	↑	↑	↑						

ח. (1 נק') ללא תלות בשינוי של הסעיף הקודם. אם $\gamma = 0$, מה מספר המדיניות האופטימליות הקיימות? נמקו.

בשני האלגוריתמים שמימשנו, במקרה שבו גאמא היא 0, לא נתחשב בutility של המצב הבא אליו עוברים, שכן המכפלה שלו תהיה ב0.

בהיבט מדיניות, נתכנס בסופו של האלגוריתם למדיניות יחידה, מS1 וS2-, לS0, שכן כל שאר המצבים לא משפיעים על המדיניות האופטימלית.

מספר המדיניות יהיה לפי מספר הצעדים של כל צומת בגרף לצומת היעד, כלומר:

$$N(s_1) * N(s_2) * N(s_3) * N(s_4) * N(s_5) * N(s_6) * N(s_7)$$
$$= 1 * 1 * 2 * 2 * 2 * 2 * 3 = 48$$

ט. (1 נק') ללא תלות בשנוי של הסעיף הקודם, הסבירי מה היה קורה אם

$$R(s_1, s_2) = R(s_2, s_1) = 2, \qquad \gamma = 1$$

בתשובתך, התייחסי גם לערכי התועלות של כל צומת וגם לשינוי במדיניות, אין צורך לחשב.

במקרה זה נקבל מעגל חיובי, שכן בכל איטרציה התועלות של s1,s2 יגדלו, לאחר מספר צעדים נגיע למצב שבו התועלות שלהן גדולות מזו של צומת המטרה ולכן הן לא ישאפו להגיע אליה (ל-s0), אלא כל פעם ישאפו להגיע לאחת מהצמתים של s1 or s2, ומשם לבחור כל פעם באחת עם התועלת הגדולה יותר בהתאם למאיפה הגענו ומה התועלת של כל אחת מהן.

חלק ב' - היכרות עם הקוד

חלק זה הוא רק עבור היכרות הקוד, עבורו עליו במלואו וודאו כי הינכם מבינים את הקוד.

mdp.py – **אתם לא צריכים לערוך כלל את הקובץ הזה**.

בקובץ זה ממומשת הסביבה של ה-mdp בתוך מחלקת MDP. הבנאי מקבל:

- board - המגדיר את המצבים האפשריים במרחב ואת התגמול לכל מצב, תגמול על הצמתים בלבד.
- terminal_states – קבוצה של המצבים הסופיים (בהכרח יש לפחות מצב אחד סופי).
- transition_function – מודל המעבר בהינתן פעולה, מה ההסתברות לכל אחת מארבע הפעולות האחרות. ההסתברויות מסודרות לפי סדר הפעולות.
- gamma – discount factor המקבל ערכים - $\gamma \in (0,1)$.
  בתרגיל זה לא נבדוק את המקרה בו $\gamma = 1$.

הערה: קבוצת הפעולות מוגדרת בבנאי והיא קבועה לכל לוח שיבחר.

למחלקת MDP יש מספר פונקציות שעשויות לשמש אתכם בתרגיל.

- print_rewards() – מדפיסה את הלוח עם ערך התגמול בכל מצב.
- print_utility(U) – מדפיסה את הלוח עם ערך התועלת U לכל מצב.
- print_policy(policy) – מדפיסה את הלוח עם הפעולה שהמדיניות policy נתנה לכל מצב שהוא לא מצב סופי.
- step(state, action) – בהינתן מצב נוכחי state ופעולה action מחזיר את המצב הבא באופן דטרמיניסטי. עבור הליכה לכיוון קיר או יציאה מהלוח הפונקציה תחזיר את המצב הנוכחי state.

חלק ג' – רטוב

כל הקוד צריך להיכתב בקובץ mdp_implementation.py

מותר להשתמש בספריות:

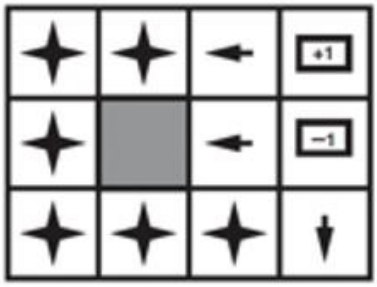
All the built-in packages in python, numpy, matplotlib, argparse, os, copy, typing, termcolor, random

עליכם לממש את הפונקציות הבאות:

- (רטוב 6 נק'): value_iteration(mdp, U_init, epsilon) – בהינתן ה-mdp, ערך התועלת ההתחלתי U_init, וחסם העליון לשגיאה מהתוחלת של התועלת האופטמילי epsilon מריץ את האלגוריתם value iteration ומחזיר את U המתקבל בסוף ריצת האלגוריתם. **TODO**
- (רטוב 4 נק'): get_policy(mdp, U) – בהינתן ה-mdp וערך התועלת U (המקיים את משוואת בלמן) מחזיר את המדיניות (במידה וקיימת יותר מאחת, מחזיר אחת מהן). **TODO**
- (רטוב 4 נק'): policy_evaluation(mdp, policy) – בהינתן ה-mdp, ומדיניות policy מחזיר את ערכי התועלת לכל מצב. **TODO**
- (רטוב 6 נק'): policy_iteration(mdp, policy_init) - בהינתן ה-mdp, ומדיניות התחלתית policy_init, מריץ את האלגוריתם policy iteration ומחזיר מדיניות אופטימלית. **TODO**

למימוש הפונקציות הבאות ניתן להשתמש באיזה ספריות שתרצו.

- (רטוב 5 נק'): get_all_policies(mdp, U, ...) – בהינתן ה-mdp, וערך התועלת U (המקיים את משוואת בלמן) מדפיס\מציג את כל המדיניות המקיימות ערך זה בלוח בודד (יש לבצע ויזואליזציה להצגת כל המדיניות), לדוגמא:



הפונקציה מחזירה את מספר המדיניות (policies) השונות הקיימות המקיימות את U. **TODO**

- (רטוב 5 נק'): get_policy_for_different_rewards(mdp, ...) – בהינתן ה-mdp מדפיס\מציג את המדיניות האופטימלית כתלות ב-r (ערכי התגמול לכל מצב שאינו סופי). **TODO**

דוגמא חלקית של פתרון אפשרי:

✍️ בנוסף לקוד עליכם לצרף להגשה היבשה את התצוגות של הפונקציות על הסביבה שניתנה בתרגיל.

```
Policies per reward range:

 -5 <= R(s) < -1.46
| R     | R     | R     | +1    |
| U     | WALL  | R     | -1    |
| R     | R     | R     | U     |

 -1.46 < R(s) < -1.3
| R     | R     | R     | +1    |
| U     | WALL  | U     | -1    |
| R     | R     | R     | U     |

 -1.3 < R(s) < -0.61
| R     | R     | R     | +1    |
| U     | WALL  | U     | -1    |
| R     | R     | U     | U     |

 -0.61 < R(s) < -0.6
| R     | R     | R     | +1    |
| U     | WALL  | U     | -1    |
| U     | R     | U     | U     |

 -0.6 < R(s) < -0.37
| R     | R     | R     | +1    |
| U     | WALL  | U     | -1    |
| U     | R     | U     | U     |

 -0.37 < R(s) < -0.05
| R     | R     | R     | +1    |
| U     | WALL  | U     | -1    |
| U     | R     | U     | L     |

 -0.05 < R(s) < -0.04
| R     | R     | R     | +1    |
| U     | WALL  | U     | -1    |
| U     | L     | U     | D     |

 -0.04 < R(s) < 0.01
| R     | R     | R     | +1    |
| U     | WALL  | U     | -1    |
| U     | L     | U     | D     |
```

```
 0.01 < R(s) < 0.09
| R     | R     | R     | +1    |
| U     | WALL  | L     | -1    |
| U     | L     | U     | D     |

 0.09 < R(s) < 0.1
| UDRL  | UDRL  | UDRL  | +1    |
| UDRL  | WALL  | L     | -1    |
| UDRL  | UDRL  | UDRL  | D     |

 0.1 < R(s) < 0.11
| UDRL  | UDRL  | L     | +1    |
| UDRL  | WALL  | L     | -1    |
| UDRL  | UDRL  | UDRL  | D     |

 0.11 < R(s) <= 5
| UDRL  | UDRL  | L     | +1    |
| UDRL  | WALL  | L     | -1    |
| UDRL  | UDRL  | UDRL  | D     |

Ranges: [-1.46, -1.3, -0.61, -0.6, -0.37, -0.05, -0.04, 0.01, 0.09, 0.1, 0.11]
Done!
```

עבור מצבים סופיים וקירות (WALL), הערך שצריך לחזור בתאים אלו עבור טבלאות המדיניות הוא None. כל ערך אחר לא יתקבל כתשובה.

main.py – דוגמת הרצה לשימוש בכל הפונקציות.

בתחילת הקובץ אנו טוענים את הסביבה משלושה קבצים:
board, terminal_states, transition_function
ויוצרים מופע של הסביבה (mdp).

- שימו לב, שכרגע הקוד ב-main לא יכול לרוץ מכיוון שאתם צריכים להשלים את הפונקציות הרלוונטיות ב-mdp_implementation.py .
- בנוסף, על מנת לראות את הלוח עם הצבעים עליכם להריץ את הקוד בIDE לדוגמה PyCharm.

הוראות הגשה

- ✓ הגשת התרגיל תתבצע אלקטרונית בזוגות בלבד.
- ✓ הקוד שלכם ייבדק (גם) באופן אוטומטי ולכן יש להקפיד על הפורמט המבוקש. הגשה שלא עומדת בפורמט לא תיבדק (ציון 0).
- ✓ המצאת נתונים לצורך בניית הגרפים אסורה ומהווה עבירת משמעת.
- ✓ הקפידו על קוד קריא ומתועד. התשובות בדוח צריכות להופיע לפי הסדר.
- ✓ יש להגיש קובץ zip יחיד בשם AI3_<id1>_<id2>.zip (ללא סוגריים משולשים) שמכיל:

- קובץ בשם AI_HW3.PDF המכיל את תשובותיכם לשאלות היבשות.
- קבצי הקוד שנדרשתם לממש בתרגיל **ואף קובץ אחר:**
  - קובץ – utils.py
  - בחלק של עצי החלטה – ID3_experiments.py, ID3.py
  - בחלק של mdp - mdp_implementation.py

אין להכיל תיקיות בקובץ ההגשה, הגשה שלא עומדת בפורמט לא תיבדק.

נספח MDP:

<u>דוגמת הרצה (שימו לב שהרצה זו השתמשה במודל הסתברותי שונה משלכם)</u>

יצירת הסביבה:

```
mdp = MDP(board=board_env,
          terminal_states=terminal_states_env,
          transition_function=transition_function_env,
          gamma=1.0)
```

הדפסת הלוח עם התגמולים לכל מצב:

```
print('@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@')
print("@@@@@@ The board and rewards @@@@@@")
print('@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@')
mdp.print_rewards()
```

פלט:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@ The board and rewards @@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
| -0.04 | -0.04 | -0.04 | +1    |
| -0.04 | WALL  | -0.04 | -1    |
| -0.04 | -0.04 | -0.04 | -0.04 |
```

Value iteration:

```
print('@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@')
print("@@@@@@@@@@ Value iteration @@@@@@@@@@")
print('@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@')

U = [[0, 0, 0, 0],
     [0, 0, 0, 0],
     [0, 0, 0, 0]]
print("\nInitial utility:")
mdp.print_utility(U)
print("\nFinal utility:")
U_new = value_iteration(mdp, U)
mdp.print_utility(U_new)
print("\nFinal policy:")
policy = get_policy(mdp, U_new)
mdp.print_policy(policy)
```

פלט:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@ Value iteration @@@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

Initial utility:
| 0.0   | 0.0   | 0.0   | 0.0   |
| 0.0   | WALL  | 0.0   | 0.0   |
| 0.0   | 0.0   | 0.0   | 0.0   |


Final utility:
| 0.812 | 0.868 | 0.918 | 1.0   |
| 0.762 | WALL  | 0.66  | -1.0  |
| 0.705 | 0.655 | 0.611 | 0.388 |


Final policy:
| RIGHT | RIGHT | RIGHT | +1    |
| UP    | WALL  | UP    | -1    |
| UP    | LEFT  | LEFT  | LEFT  |
```

:Policy iteration

```python
print('@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@')
print("@@@@@@@@@@ Policy iteration @@@@@@@@@")
print('@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@')

print("\nPolicy evaluation:")
U_eval = policy_evaluation(mdp, policy)
mdp.print_utility(U_eval)

policy = [['UP', 'UP', 'UP', 0],
          ['UP', 'WALL', 'UP', 0],
          ['UP', 'UP', 'UP', 'UP']]
print("\nInitial policy:")
mdp.print_policy(policy)
print("\nFinal policy:")
policy_new = policy_iteration(mdp, policy)
mdp.print_policy(policy_new)

print("Done!")
```

פלט:

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@@@@@@@@@@ Policy iteration @@@@@@@@@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

Policy evaluation:
| 0.812 | 0.868 | 0.918 | 1.0   |
| 0.762 | WALL  | 0.66  | -1.0  |
| 0.705 | 0.655 | 0.611 | 0.388 |


Initial policy:
| UP    | UP    | UP    | +1    |
| UP    | WALL  | UP    | -1    |
| UP    | UP    | UP    | UP    |


Final policy:
| RIGHT | RIGHT | RIGHT | +1    |
| UP    | WALL  | UP    | -1    |
| UP    | LEFT  | LEFT  | LEFT  |

Done!
```