מגישים: שניר תמם (204594154), אופיר יזרעאלב (318755147)

<u>הסבר כללי על הקוד</u>

ראשית, הפונקציה get_P מחשבת את מטריצת המעברים, את המצבים הטרמינליים ואת המעברים האפשריים. כעת, מריצים את policy_iteration:

- מאתחלים את ה-policy להיות אקראית עם init_policy ואת ה-value להיות 0 עם init_value.
- בלולאה, מחשבים את הערך החדש של ה-value לפי ה-policy האחרון בעזרת policy_evaluation ואז את ה-policy החדש לפי ה-value בעזרת policy_improvement.
- אם ה-policy לא השתנה, עוצרים את הלולאה.

המדיניות שהתקבלה בתום הלולאה הינה המדיניות האופטימלית!

<u>שיטות מרכזיות</u>

במטלה זו השתמשנו בשיטה מרכזית אחת – policy iteration. אלגוריתם זה משמש למציאת מדיניות אופטימלית בסביבות תהליכי החלטה מרקוביים. באלגוריתם זה מעדכנים בלולאה את המדיניות לפי ערכי מצבים עדכניים ואת ערכי המצבים לפי מדיניות עדכנית, עד אשר המדיניות לא משתנה בעת העדכון – ומכיוון שהאלגוריתם מתכנס למדיניות אופטימלית, המדיניות בשלב זה הינה האופטימלית.

<u>סימולציות</u>

<u>סימולציה 1</u>

Total steps: 7

Total reward: +14

1. 1,0, 0,0, 4,0, move_north, -1
2. 0,0, 0,0, 4,0, pick_up_passenger, -1
3. 0,0, 0,0, 4,0, move_south, -1
4. 1,0, 1,0, 4,0, move_south, -1
5. 2,0, 2,0, 4,0, move_south, -1
6. 3,0, 3,0, 4,0, move_south, -1
7. 4,0, 4,0, 4,0, drop_off_passenger, +20

<u>סימולציה 2</u>

Total steps: 10

Total reward: +11

1. 4,0, 0,0, 4,0, move_north, -1
2. 3,0, 0,0, 4,0, move_north, -1
3. 2,0, 0,0, 4,0, move_north, -1
4. 1,0, 0,0, 4,0, move_north, -1
5. 0,0, 0,0, 4,0, pick_up_passenger, -1
6. 0,0, 0,0, 4,0, move_south, -1
7. 1,0, 1,0, 4,0, move_south, -1
8. 2,0, 2,0, 4,0, move_south, -1
9. 3,0, 3,0, 4,0, move_south, -1

10. 4,0, 4,0, 4,0, drop_off_passenger, +20

סימולציה 3

Total steps: 9

Total reward: +12

1. 4,3, 4,3, 0,0, pick_up_passenger, -1
2. 4,3, 4,3, 0,0, move_north, -1
3. 3,3, 3,3, 0,0, move_north, -1
4. 2,3, 2,3, 0,0, move_west, -1
5. 2,2, 2,2, 0,0, move_west, -1
6. 2,1, 2,1, 0,0, move_west, -1
7. 2,0, 2,0, 0,0, move_north, -1
8. 1,0, 1,0, 0,0, move_north, -1
9. 0,0, 0,0, 0,0, drop_off_passenger, +20