תרגיל בית 2 - פתרון בעיות בתנאי ידע חלקי

מבוא

מטרת התרגיל היא להתנסות בפתרון בעיות סוקובן בתנאי ידע חלקי.

עקרון המשחק נשאר דומה לזה שראיתם בתרגיל הראשון: על השחקן למלא את כל היעדים בארגזים במבוך תוך שימוש במספר תזוזות מינימלי. בשונה מהתרגיל הראשון, שבו היה לסוכן ידע מלא על מצב העולם, הפעם הידע שיש לסוכן על העולם יהיה חלקי בלבד.

הבעיה

כקלט הסוכן יקבל את המצב ההתחלתי של המשחק. זוהי מטריצה בגודל NxM שמייצגת חלקית את לוח המשחק כשהמצאות הקרח מוסתרת מהסוכן. כמו בתרגיל הקודם, לסוכן מותר לזוז ימינה, שמאלה, למעלה ולמטה. כמו כן, הסוכן יכול להזיז את הארגזים ביחד עם התזוזה באותו אופן כמו שעשה זאת בתרגיל הראשון.

ההבדל היחיד, אך משמעותי, שבתרגיל זה אין באפשרותכם לדעת איפה נמצא הקרח בלי להפעיל פעולה שתוכל לחשוף אותו. ולכן, אין ביכולתכם לעשות תוכנית לפתרון כל התרגיל מראש, ותצטרכו להחליט על הפעולה הבאה לאחר קבלת תצפיות על מצב העולם אחרי הפעלת פעולה קודמת.

תיאור משימה

לתרגיל זה מצורף קוד בדיקה שמחזיק את המצב האמיתי של העולם (שאין באפשרות הסוכן שלכם לגשת אליו ישירות), ומעביר את מצב העולם החלקי למחלקה SokobanController שעליכם לממש. בפרט בה עליכם לממש את הפונקציה get_next_action שמקבלת את מצב העולם העדכני, ומחזירה אחת מ-4 הפעולות המותרות - "R", "L", "D", "U".

הידע החלקי מתבטא בכך שבכל קלט ש-SokobanController תקבל, לא תהיה לסוכן אפשרות להבדיל בין תא עם קרח ובלעדיו (שניהם יקבלו את אותו קוד מספרי - 10, 15, 17 כתלות באובייקטים הנמצאים בתא). הדרך היחידה לגילוי הקרח היא להפעיל פעולה (למשל ללכת או לדחוף ארגז) על התא הרצוי, ולהסיק את המצאות הקרח על סמך התוצאה של פעולה זאת (שאותה תוכלו לראות בקלט לפעולה הבאה).

לבסוף, המטרה היא להביא את המערכת למצב שבו כל היעדים מכוסים ע"י הארגזים מכמות צעדים קטנה ככל האפשר.

חובה להשתמש בקובץ ex2.py המצורף שמכיל את חתימות הפונקציות שעליכם לממש.

ייצוג הקלט

בדומה לתרגיל ראשון, הקלט גם ל-constructor וגם ל-get_next_action הוא מטריצה MxN, כשכל כניסה בה מייצגת אובייקט נמצא במשבצת הרלוונטית

משמעות של הערכים של תוכן התאים היא:

- 99 - קיר
- 10 - משבצת שנראית ריקה
- 15 - ארגז על משבצת שנראית ריקה
- 17 - שחקן על משבצת שנראית ריקה
- 20 - יעד
- 25 - יעד עם ארגז עליו
- 27 - יעד עם שחקן עליו

כשמשבצת שנראית ריקה יכולה להיות משבצת ריקה רגילה או משבצת עם קרח.

אופן וחוקיות התזוזה

זהים לגמרי לתרגיל 1, כשתזוזה על משבצת שנראית ריקה מתבצעת לפי הטיפוס האמיתי שלה.

חבילות מותרות לשימוש

מותר להשתמש **רק** בחבילות שנמצאות בספריה הסטנדרטית של פייתון.
אסור להשתמש ב-threading (למרות שהיא נמצאת בספריה הסטנדרטית)
כמו כן, **אסור** לייבא קובץ checker.py או inputs.py, או לנסות לגשת בכל דרך אחרת למפה האמיתית

הסבר קצר על הקוד המצורף

הקוד מורכב מ-4 קבצי פייתון:

1. ex2.py - קובץ שבו נעשית העבודה העיקרית שלכם, הקובץ היחיד שעליכם לשנות ולהגיש אמור להכיל את המחלקה SokobanController המכילה פונקציה get_next_action.
2. checker.py - קובץ המכיל את קוד הבדיקה שבקריאה לכל get_next_action מעביר לה את המפה עם קרח מוסתר
3. inputs.py - קובץ המכיל מפות שונות
4. utils.py - קובץ עזר המכיל הרבה פונקציות שאולי תראו כשימושיות

בדיקת התרגיל

התרגיל ייבדק באופן אוטומטי, ולכן חשוב להקפיד על שמות מדויקים של קבצים, מחלקות ופעולות.
הבדיקה תתבצע על ידי הרצת קובץ checker.py
זמנים שהקוד חייב לעמוד בהם - ה-constructor חייב להסתיים ב-60 שניות. get_next_action חייבת להסתיים ב-5 שניות.
כמו כן, כל ריצת הקוד מהתחלה עד להשגת מטרה צריכה להסתיים ב $(5+M+N)^2$ צעדים, כש-M ו-N הם המימדים של המטריצה שאותה מקבלים כקלט.
נא להוריד את כל ההדפסות לפני הגשת התרגיל!

הגשה, בדיקת תרגיל וציונים

- ההגשה ביחידים או בזוגות בלבד
- מספיק שאחד מבני הזוג יגיש את התרגיל, אין צורך ששני בני הזוג יגישו
- את ת.ז. של המגישים יש לרשום במשתנה ids בקובץ ex2.py בתור strings בתוך רשימה קיימת. אם המגיש יחיד, אז הרשימה צריכה לכלול string בודד
- יש להגיש רק את הקובץ ex2.py אין להגיש קבצי עזר. אין להגיש בפורמט אחר (zip, rar וכו')
- 80% מהציון יינתן על עמידה בדרישות בסיס, ו20% על בסיס תחרות בין המגישים
- דרישות בסיס - פתרון כל תרגיל כך ששום מימד של מטריצת הקלט לא עולה על 5.
- נוסחה לציון בתחרות זהה לנוסחה מתרגיל ראשון