תרגיל בית 1

# מבוא

מטרת התרגיל היא להתנסות בפתרון בעיות תכנון מורכבות על ידי שיטות חיפוש שונות. הבעיה שתידרשו לפתור בתרגיל זה היא גרסה של משחק וומפוס (wumpus). מהות המשחק היא לנווט במבוך דו-מימדי ולהגיע לזהב תוך התגברות על מכשולים מסוגים שונים.

כקלט תקבלו את המצב ההתחלתי של המשחק: מטריצה בגודל M על N שמייצגת את מפת המשחק. ערכים במטריצה מסמנים את האובייקטים הנמצאים בתא הרלוונטי של המפה.

סוגי האובייקטים השונים שתפגשו הם: קירות, בורות, דלתות, מפתחות, זהב, מפלצות וגיבורים. תיאור מפורט של כל האובייקטים נמצא בסעיף "אובייקטים".

השליטה שלכם היא על פעולות הגיבורים בלבד. המטרה שלכם היא לנווט את אחד הגיבורים לאחד התאים המכילים זהב. תיאור מפורט של כל הפעולות ותוצאות הפעלתן נמצא בסעיף "פעולות".

כפלט עליכם להחזיר פתרון לבעיה - סדרת פעולות ממצב התחלתי למצב מטרה (אחד הגיבורים נמצא בתא בו נמצא הזהב). אם הבעיה אינה פתירה - עליכם להחזיר פלט מיוחד. תיאור מפורט של סוגי פלט שונים שתוכלו לקבל ניתן למצוא בסעיף "פלט".

# תיאור המשימה

לתרגיל זה מצורף קוד המממש אלגוריתמי חיפוש שנלמדו במהלך הקורס. עליכם לממש את המחלקה המייצגת משחק וומפוס כך שניתן יהיה לפתור משחק נתון על ידי שימוש בקוד החיפוש המצורף.

עליכם לממש את הפונקציות הבאות (ניתן כמובן גם לממש פונקציות נוספות):

1. פונקציית actions המקבלת מצב ומחזירה tuple הכולל את כל הפעולות המותרות עבור אותו מצב.
2. פונקציית result המקבלת מצב ופעולה ומחזירה מצב חדש, הוא הנוצר כתוצאה מהפעלת אותה פעולה על המצב הנתון.
3. פונקציית goal\_test המקבלת מצב ומחזירה True אם המצב הנתון נחשב כמצב מטרה עפ"י הקריטריונים לעיל ו-False אחרת.
4. פונקציית היוריסטיקה (h) המקבלת מצב ומחזירה את העלות המשוערת להגעה מהמצב הנתון למצב מטרה. שימו לב כי הביצועים של אלגוריתמי החיפוש תלויים בטיב הפונקציה ההיוריסטית.
5. כמו כן, עליכם לכתוב את מספרי תעודת זהות שלכם במשתנה ids מחוץ למחלקה.

חובה להשתמש בקובץ ex1.py המצורף המכיל את חתימות הפונקציות שיש לממש.

**שימו לב!**  אלגוריתמי חיפוש כבר נתונים לכם, כל מה שנשאר כדי לפתור את הבעיה זה לממש את הפונקציות האלה. כדי להפעיל את הקוד, צריך להריץ את ה-main בקובץ check.py

# אובייקטים

להלן רשימה של אובייקטים אפשריים וקידוד שלהם במטריצת הקלט:

* מעברים - מקומות בהם הגיבורים וחצים יכולים לעבור ללא מגבלות - מקודד ע"י מספר 10
* קירות - מקומות בהם אין מעבר לגיבורים ואי אפשר לירות חצים דרכם - מקודד ע"י 20
* בורות - מקומות בהם אין מעבר לגיבורים אבל ניתן לירות חצים דרכם - מקודד ע"י 30
* גיבורים - סוכנים שלכם לצורך השגת מטרה - מקודדים במספרים 11 עד 14 (כולל). לכל גיבור מספר ייחודי משלו. לא יהיו לעולם יותר מ-4 גיבורים בקלט (אבל יכולים להיות פחות).
* דלתות - דלת זהה לקיר כל עוד היא לא נפתחה. כדי לפתוח את הדלת, יש צורך במפתח, **כשלכל דלת מפתח ייחודי משלה**. לאחד איסוף המפתח, **כל** גיבור (ללא התחשבות במי אסף את המפתח) יכול לפתוח את הדלת ע"י כך שהוא עובר לתא בו נמצאת הדלת. הדלת הפתוחה זהה למעבר רגיל. אי אפשר לסגור את הדלת שנפתחה. הדלתות מקודדות במספרים 45 עד 49 כולל. לעולם לא יהיו יותר מ-5 דלתות בקלט, אבל יכולות להיות פחות. **אין הבטחה שלכל דלת בקלט יש מפתח!**
* מפתחות - מקודדים ע"י מספרים 55 עד 59. הספרה האחרונה היא מקודדת את הדלת שהמפתח פותח. למשל, מפתח 57 מאפשר מעבר דרך דלת 47. כדי לקחת את המפתח, הגיבור אמור לעבור לתא בו המפתח נמצא. כתוצאה מכך המפתח נעלם (ובמקומו נשאר מעבר רגיל), וכל סוכן יכול לפתוח את הדלת מתאימה. לעולם לא יהיו יותר מ-5 דלתות בקלט, **ויכולים להיות מפתחות שלא פותחים שום דלת.** לכל צורך אחר, התא עם המפתח ייחשב כמעבר רגיל.
* מפלצות - מקודדות ע"י מספר 60. לא ניתן לעבור בתא בו יש מפלצת חיה. ניתן להרוג את המפלצת ב-2 דרכים - עם חרב או עם חץ. כדי להרוג את המפלצת עם חץ, צריך לבצע ירי (ראה סעיף "פעולות"). כדי להרוג מפלצת עם החרב, צריך להגיע לתא סמוך למפלצת (מלמעלה, מלמטה, מימין או משמאל למפלצת. תא הנמצא באלכסון **לא נחשב סמוך!**). כתוצאה מכך, גם הגיבור וגם המפלצת נהרגים.
* זהב - מקודד ע"י מספר 70. ברגע שאחד הגיבורים דורך על התא עם הזהב ושורד את המהלך (זאת אומרת, אין מפלצת בתאים הסמוכים), המשחק נגמר. החצים יכולים לעבור דרך הזהב

# פעולות

כל פעולה מיוצגת ע"י tuple, שבמקום ה-0 שלו תמיד נמצא קוד הפעולה עצמה, ובשאר המקומות פרמטרים ספציפיים לפעולה. רשימת הפעולות:

* תזוזה – (move”, no\_of\_hero, direction") - מזיזה את אחד הגיבורים לתא הסמוך בכיוון מסויים.
  + הפרמטר no\_of\_hero זה מספר הגיבור שברצונכם להזיז (מספר בין 11 ל-14 כולל)
  + הפרמטר direction מסמן את הכיוון שבו ברצונכם לבצע תזוזה. יכול לקבל את אחד מהערכים הבאים:
  + "U" "D", "R", "L" - שמאלה, ימינה, למטה ולמעלה בהתאמה.
  + תוצאות הפעולה תלויות באובייקט שנמצא בתא היעד - התא שאילו מתבצעת התזוזה
    - אם היעד הוא קיר, גיבור אחר, או מחוץ לגבולות המפה – אין שינוי במצב המשחק
    - אם היעד הוא בור – הגיבור נעלם
    - אם היעד הוא מעבר הסמוך למפלצת – הגיבור והמפלצת הורגים אחד את השניה ונעלמים (נשאר מעבר בשתי התאים)
    - אם היעד הוא מעבר רגיל – הגיבור זז לתא היעד
    - אם היעד הוא דלת שלא אספתם מפתח מתאים לה – תוצאה זהה לניסיון תזוזה לתוך קיר
    - אם היעד הוא דלת והשגתם את המפתח המתאים – כתוצאה מתזוזה הדלת נעלמת (במקומה נשאר מעבר) והגיבור עובר לתא שהתפנה
* פעולת ירי חץ - (shoot", no\_of\_hero, direction") - יורה חץ לכיוון מסוים. החץ עובר כל התאים בכיוון הזה עד שלא פוגש אובייקט שעוצר אותו, או שהוא יוצא מגבולות המפה. אובייקטים שעוצרים את החץ - קיר, דלת, גיבור ומפלצת. אם החץ פוגע בגיבור או מפלצת - הם נהרגים ונעלמים.

# פלט

יש שלושה סוגי קלט שאתם יכולים לקבל כתוצאה מהרצת התוכנה:

* שגיאה
* אין פתרון (None,א 2-, 2-). הפלט הזה ייתקבל אם נגמר הזמן להרצת בעיה, והקוד עוד לא מצא פתרון
* פתרון בפורמט (רצף פעולות, זמן ריצה, כמות צעדים)

# דגשים

* שימו לב שכדי לקבל את המצב מתוך קודקוד החיפוש (node), ניתן לגשת ל-node.state
* אתם יכולים לייצג מצב של הבעיה איך שאתם רוצים, אבל שימו לב שהקוד של החיפוש דורש שהמצב יהיה hashable , ולכן לא ניתן להשתמש ברשימה או במילון (או ב – tuple המכיל רשימה או מילון) על מנת ליצג מצב. ניתן (אך לא מומלץ) לממש מחלקה המתארת מצב. במקרה כזה עליכם לממש למחלקה הזאת פונקציית hash
* אמנם ייצוג המצב הוא בחירה שלכם אולם **יש להקפיד על ייצוג הפעולות בדיוק כפי שכתוב** – אותיות גדולות/קטנות, קווים תחתונים, ייצוג הפעולות נכון וכו'. הבדיקה אוטומטית, ולכן אם תטעו כאן תקבלו ציון נמוך מאוד, וחבל.
* **תקראו את הערות בקוד, יש שם הרבה מידע על השימוש בפונקציות**

# בדיקת התרגיל

התרגיל המוגש ייבדק באופן אוטומטי, ולכן חשוב להקפיד על שמות מדויקים של קבצים, מחלקות ופעולות. הבדיקה האוטומטית תשתמש באלגוריתמי חיפוש שונים (A\* וGBFS) על מנת לפתור את אוסף הקלטים בגדלים שונים. לאחר מכן הפתרון ייבדק צעד אחר צעד על מנת לוודא שהפתרון אכן תקין.

הקובץ check.py המצורף מריץ כל את אלגוריתמי החיפוש שנבדוק על מספר קלטים. קובץ זה לא מבצע בדיקת נכונות של הפתרון המוחזר, ולכן זוהי אחריות שלכם לוודא שהפתרונות שלכם נכונים.

**שימו לב!** הביצועים, וכמובן גם הציון, יינתן רק על סמך הריצות של GBFS בלבד. כל שאר האלגוריתמים נמצאים בקוד בשביל התנסות בלבד.

# הסבר קצר על הקוד המצורף

הקוד מורכב מחמישה קבצי פייתון:

1. ex1.py – קובץ שבו נעשית העבודה העיקרית שלכם, והקובץ היחיד שעליכם לשנות. אמור להכיל את ה-class של WumpusProblem המכיל את כל הפונקציות כפי שמתואר בסעיף "תיאור משימה". הקובץ כולל את חתימות של הפונקציות שעליכם לממש. **התרגיל יבדק עם קובץ ex1.py שלכם, ושאר הקבצים כפי שהם מופיעים במצבם המקורי ולכן אין טעם לשנות קבצים אחרים** (למעט הבעיות השונות שנבדוק עליהם את הקוד).
2. check.py – קובץ המכיל פונקציות מעטפת המנסות לפתור את הבעיה, ומכיל בעיה קטנה לדוגמא שאפשר לפתור. זה הקובץ שעליכם להריץ לבדיקת הפתרון שלכם.
3. search.py – מכיל את אלגוריתמי החיפוש, ומכיל את מחלקת node.
4. שאר הקבצים – קבצי עזר

# הגשה, בדיקת התרגיל וציונים

* הגשה בזוגות או יחידים בלבד
* **מספיק שאחד מבני הזוג יגיש** את התרגיל, אין צורך ששני בני הזוג יגישו.
* את הת.ז. של המגישים יש לרשום במשתנה ids בקובץ ex1.py (בתור strings בתוך רשימה קיימת. אם המגיש יחיד, אז רשימה צריכה להכיל string בודד)
* יש להגיש רק את הקובץ ex1.py. אין להגיש קבצי עזר המצורפים לתרגיל. אין להגיש קובץ בפורמט אחר (לדוגמא zip או rar)
* 80% מהציון יינתן על עמידה בדרישות הבסיס, ו20% יינתן על בסיס תחרות בין המגישים
* התחרות תתבצע על ביצועים של אלגוריתם GBFS על מגוון קלטים. על האלגוריתם להוציא **התכנית הקצרה ככל ניתן** תוך 60 שניות