**מבוא לבינה מלאכותית**

**סמסטר חורף תשפ"ד**

**מטלה 2**תאריך הגשה: 15.02.24 23:55

# הנחיות

* שאלות בנושא מטלה זו יש לשאול דרך המודל, בפורום "מטלה 2".
* הוראות להגשת המטלה מופיעים בסוף מסמך זה.
* הקבצים הנדרשים להרצת הקוד הינם:
  + minimax\_isolation.py
  + player\_agent.py
  + heuristics.py
  + alpha\_beta\_isolation.py
  + heuristic\_alpha\_beta\_isolation.py
  + game\_state.py
  + game\_engine.py
* העבודה להגשה בזוגות בלבד אלא אם ניתן אישור מיוחד להגיש אחרת.
* לפני שניגשים לממש את המטלה מומלץ לעיין רבות בהסברים וכן בקוד הקיים.
* פתרון המטלה שתגישו ייבדק מול שאר ההגשות על ידי תוכנת העתקות.
* **מי שימצא כי העתיק יכשל בקורס וכן יועבר לוועדת משמעת אוניברסיטאית**.
* הפרויקט נכתב וייבדק בשפת התכנות python.
* יש להתקין את החבילה numpy.
* מסמך זה בנוי באופן הבא: תיאור המטלה, בעיית משחק שני שחקנים, מרחב הבעיה, שאלות המטלה, הסבר על המימוש הקיים והסבר על הגשת המטלה.

# תיאור המטלה נתון משחק שני שחקנים בשם Isolation. בהינתן לוח משחק , לכל שחקן יש דמות המתחילה במיקום מסויים על הלוח. כל שחקן בתורו יכול להזיז את הדמות שלו לאורך 8 כיוונים (למעלה-למטה, שמאלה-ימינה ובאלכסונים). הדמות יכולה לזוז לכל משבצת בכיוון מסויים שלושת התנאים הבאים מתקיימים:

1. לא ניתן להגיע למשבצת שכבר ביקרו בה
2. לא ניתן לדלג מעל משבצות שביקרו בהן(לעבור באלכסון דרך מותר כל עוד משבצות האלכסון פנויות)
3. לא ניתן לדלג מעל לדמות השחקן השני

# בעיית משחק שני שחקנים בהינתן מיקום התחלתי של שני השחקנים, עליכם לממש אלגוריתמים שיחשבו את ערך ה-maximin של המשחק.

# מרחב הבעיה

בכל שלב במשחק, הדמות נמצאת באחד התאים במטריצה, כאשר המטרה שלה היא שלדמות השניה יגמרו המהלכים לפניה.

דוגמה ללוח משחק:

# שאלות המטלה

* עליכם לממש את אלגוריתם alphabeta. לשם כך האלגוריתמים minimax ו-maximin ממומשים בקובץ minimax\_isolation.py, ואילו את אלגוריתם alphabeta יש לשנות ולהתאים בקובץ alpha\_beta\_isolation.py. האלגוריתם הממומש כעת בקובץ alpha\_beta\_isolation.py הינו גם כן minimax ו-maximin, ויש לשנות את הפונקציות בקובץ כך שהאלגוריתם שיממומש יהיה alpha beta pruning).

שימו לב, כאשר מבצעים גיזום יש להחזיר את v וגם None, מכיוון שתוכנית זו שומרת את הצעד שיש לבצע עבור השחקן הפורס את עץ המשחק(כלומר, “return v, None”). ניתן לשנות את חתימות הפונקציה בשביל נכונות האלגוריתם alphabeta.

* עליכם לממש את אלגוריתם alphabeta גם בקובץ heuristic\_alpha\_beta\_isolation.py (אותם שינויים כמו בסעיף א'). בנוסף, כדי להתמודד עם עצי משחק גדולים יותר- עליכם לממש יוריסטיקה עבור קודקודים שאינם מציינים סיום משחק. השלימו את הפונקציה base\_heuristic בקובץ heuristics.py כך שתקבל מצב של משחק ותחזיר את ההפרש בין כמות **הצעדים החוקיים של השחקן הראשון** לבין **הצעדים החוקיים של השחקן השני.** כלומר, עליכם תמיד להחזיר p1\_moves - p2\_moves(לא משנה תור מי לשחק בצעד הנתון).
* הינכם יכולים לממש יוריסטיקה מתקדמת בפונקציה advanced\_heuristic. היוריסטיקה הטובה ביותר תזכה את המגישים ב 2 נקודות בונוס לציון הסופי. כדי להחליט על היוריסטיקה הטובה ביותר, נערוך טורניר בין קבוצות המגישים כך שהאסטרטגיה של השחקן תשתמש ב advanced\_heuristic על מנת להחליט על כל אחד מצעדי המשחק. הקבוצה שתנצח בטורנטיר תקבל את נקודות הבונוס.

**שימו לב:** כאשר אתם מריצים את אלגוריתם minimax אתם תוכלו לחשב משחקים רק עבור לוח בגודל 3X3. כדי לנסות לשחק על לוח בגודל 4X4 עליכם לממש את alphabeta pruning. בשביל לוחות גודולים יותר תצטרכו להשתמש ביוריסטיקה ולהגביל את עומק החיפוש.

# הגשת המטלה

* יש להגיש **שלושה קבצים בלבד** לאתר המודל בדומה להגשת המטלה הקודמת. הקבצים הנדרשים:
  + heuristics.py
  + alpha\_beta\_isolation.py
  + heuristic\_alpha\_beta\_isolation.py

הגשת המטלות תתבצע ישירות מול מערכת המודל בצורה אלקטרונית.

* **שימו לב**: ישנה אפשרות להגיש את המטלה התכנותית מספר פעמים ובכל הגשה לקבל חיווי, כלומר תקבלו באופן מיידי את הציון לביצוע התרגיל.
* ניתן לראות את הפידבק להרצה (קומפילציה, מספר טסטים שעברו, שגיאות זמן ריצה וכו'...).

לאחר סיום ההרצה יתקבלו התוצאות.