***חלק ב'***

1. השחקן GreedtAgent בודק עבור כל מהלך אפשרי שלו את כל המהלכים האפשריים של יריביו, ומבין כל השילובים האלו השחקן מכניס לרשימה את כל המהלכים בעלי הערך היוריסטי הגבוה ביותר. לאחר מכן בוחר רנדומלית מהלך מתוך רשימה זו.

הפונקציה היוריסטית בה השחקן משתמש מניחה כי בכל מהלך עתידי של השחקן יאכל פרי עד שנגמר מספר התורות במשחק או עד שייגמרו כל הפירות במשחק (סוכמת גם את אורכי הנחשים מכיוון שייתכן כי יהפכו לפירות אם ימותו). בנוסף, עבור כל צעד עתידי שיותר מתקדם מהצעד הנוכחי נקטין את המשקל של צעד זה בחישוב הפונקציה, מכיוון שהוא פחות וודאי. הפונקציה מחושבת באופן הבא – עבור מקדם הנחה נתון (a < 1) ועבור מספר צעדים אפשרי (k) הפונקציה מחשבת :

הפונקציה מחזירה את תוצאת החישוב הנ"ל + אורך הנחש לאחר הצעד.

הפונקציה היוריסטית מאפשרת להימנע ממוות מכיוון שעבור צעד בו הנחש מת, הפונקציה מחזירה רק את אורך הנחש ללא התוספת העתידית. מכיוון שתוספת זו תמיד גדולה מ-0, במקרה של מוות תוצאת היוריסטיקה תהיה פחות עדיפה על מהלכים אחרים בהם הנחש לא מת, לכן לא תיבחר.

***חלק ג'***

1. נגדיר יוריסטיקה המתחשבת בערכים הבאים:

* snake\_length = the length of the snake in the next state (considering eating fruits or dying)
* distance\_from\_closest\_walls = calculates the distance from the head position of the snake to the closest horizonal wall and vertical wall. This value holds the sum distance from both walls.
* distance\_from\_closest\_fruit = calculates the "Manhattan" distance from the head position of the snake to the closest fruit.
* closest\_rival\_snake = calculates the "Manhattan" distance from the head position of the snake to the position of each rival snake and chooses the minimal distance. Meaning, the distance from the closest point of the rival snake body to the head of the player snake. Then, if this distance is greater than the length from this closest point of the rival snake body to its tail, then by the time the player will potentially reach the rival snake, it wont be at this point anymore, so we don’t take this rival snake into consideration.

הפונקציה היוריסטית מחזירה את המשוואה הבאה:

(w\_of\_len \* snake\_length) + (w\_of\_walls\*distance\_from\_closest\_walls) +  
 (w\_of\_rival\*closest\_rival\_snake) – (w\_of\_fruits\*distance\_from\_closest\_fruit)

כאשר הפרמטרים מהצורה W\_\* מציינים משקל מסוים אותו אנחנו מכפילים בערך המתאים עבור כל פרמטר על מנת לקבל את התוצאה הרצויה.

1. נתייחס עבור הערכים שתוארו לעיל ונסביר את המוטיבציה מאחוריהם:

* אורך הנחש (snake\_length) – כאשר אנחנו מחשבים במהלך הבא של הנחש את אורכו אנחנו בפועל מייחסים חשיבות האם הנחש אכל פרי או לא. לכן אם במהלך הבא יש אפשרות לאכול פרי, הפונקציה היוריסטית נותנת למהלך זה משקל גדול יותר.
* מרחק מהקירות (distance\_from\_closest\_walls) – המוטיבציה מאחורי ערך זה היא שככל שהנחש קרוב יותר לקיר יש פוטנציאל גדול יותר להיתקע ופחות מהלכים טובים אפשריים. לכן הפונקציה היוריסטית תתעדף מהלכים שמרחיקים את הנחש מהקיר.
* מרחק מהפרי הקרוב ביותר (distance\_from\_closest\_fruit) - המוטיבציה מאחורי ערך זה היא שככל שהנחש קרוב יותר לפרי יש פוטנציאל גדול יותר לאכול אותו. לכן הפונקציה היוריסטית תתעדף מהלכים שמקרבים את הנחש לפירות.
* מרחק מהיריב הקרוב ביותר (closest\_rival\_snake) – המוטיבציה מאחורי ערך זה היא שכאשר הנחש נתקל ביריב הנחש מת, וככל שהוא קרוב יותר ליריב סיכויו להיתקל בו גבוהים יותר. הפונקציה היוריסטית לוקחת גם בחשבון את המרחק מהנקודה הקרובה ביותר של היריב אל ראש הנחש ביחס למרחק של אותה נקודה מהזנב של אותו נחש, מכיוון שכאשר הנחש יגיע לנקודה זו היריב כבר לא יהיה שם. לכן הפונקציה היוריסטית תתעדף מהלכים שמרחיקים את הנחש מהיריב.

כל הערכים (פרט לאורך הנחש) שציינו אינם נלקחים בחשבון ביורסטיקה של הנחש הפשוט, לכן לפי המוטיבציה שהסברנו אנחנו צופים כי הביצועים של השחקן החדש ישתפרו.