**תרגיל בית 2 – מבוא לבינה מלאכותית**

מגישים:  
 אופק מלכה 311381206  
 עדן בר 205907256

**משימה 1: משחק איקס עיגול**

|  |  |
| --- | --- |
| ''' BOARD INDEX LAYOUT  00 | 01 | 02  --------------------  10 | 11 | 12  --------------------  20 | 21 | 22 '''  *from* math *import* inf   *def* main():  board = [['', '', 'X'],  ['', 'X', ''],  ['O', '', '']]  player = 'O'  *while True*:  empty\_spots\_arr = empty\_spots(board)  *if* is\_winner(board, 'O'):  print("O WON!")  *break  if* is\_winner(board, 'X'):  print("X WON!")  *break  if not* empty\_spots\_arr:  print("TIE!")  *break* best\_score = -inf  *for* i, j *in* empty\_spots\_arr:  board[i][j] = player  score = minimax(board, 'X' *if* player != 'X' *else* 'O', *False*)  *if* score > best\_score:  best\_score = score  best\_move = (i, j)  board[i][j] = ''  i, j = best\_move  board[i][j] = player  player = 'X' *if* player != 'X' *else* 'O'  print\_board(board)   # The minimax algorithm *def* minimax(*board*: list[list[str]], *player*: str, *is\_maximising*: bool) -> int:  *if* is\_winner(*board*, *player*):  *return* 1 *if is\_maximising else* -1  *if* is\_winner(*board*, 'X' *if player* != 'X' *else* 'O'):  *return* -1 *if is\_maximising else* 1  empty\_spots\_arr = empty\_spots(*board*)  *if not* empty\_spots\_arr:  *return* 0  *if is\_maximising*:  best\_score = -inf  *for* i, j *in* empty\_spots\_arr:  *board*[i][j] = *player* score = minimax(*board*, 'X' *if player* != 'X' *else* 'O', *not is\_maximising*)  *if* score > best\_score:  best\_score = score  best\_move = (i, j)  *board*[i][j] = ''  *return* best\_score  *else*:  best\_score = inf  *for* i, j *in* empty\_spots\_arr:  *board*[i][j] = *player* score = minimax(*board*, 'X' *if player* != 'X' *else* 'O', *not is\_maximising*)  *if* score < best\_score:  best\_score = score  best\_move = (i, j)  *board*[i][j] = ''  *return* best\_score          # Human friendly print of a board *def* print\_board(*board*: list[list[str]]):  *for* i *in* range(3):  print(" {:1} | {:1} | {:1} ".format(*board*[i][0], *board*[i][1], *board*[i][2]))  *if* i < 2:  print("----------------")  # Returns a list of (y,x) tuples representing empty spots for a given board. *def* empty\_spots(*board*: list[list[str]]) -> list[tuple[int, int]]:  empty\_spots\_arr = []  *for* i *in* range(3):  *for* j *in* range(3):  *if not board*[i][j]:  empty\_spots\_arr.append((i, j))  *return* empty\_spots\_arr  # Returns rather player won the game or not for a given board. *def* is\_winner(*board*: list[list[str]], *player*: str) -> bool:  win\_patterns = [[*board*[0][0], *board*[0][1], *board*[0][2]], # Upper row  [*board*[1][0], *board*[1][1], *board*[1][2]], # Middle row  [*board*[2][0], *board*[2][1], *board*[2][2]], # Bottom row  [*board*[0][0], *board*[1][0], *board*[2][0]], # Right column  [*board*[0][1], *board*[1][1], *board*[2][1]], # Middle column  [*board*[0][2], *board*[1][2], *board*[2][2]], # Left column  [*board*[0][0], *board*[1][1], *board*[2][2]], # Main diagonal  [*board*[0][2], *board*[1][1], *board*[2][0]]] # Secondary diagonal  *return* ([*player*] \* 3) *in* win\_patterns   *if* \_\_name\_\_ == '\_\_main\_\_':  main() | לולאה ראשית, שחקן מחפש את המהלך הטוב ביותר עבורו ולאחריו התור עובר לשחקן היריב וחזור חלילה.  אלגוריתם המינימקס כפי שלמדנו בשיעור.  פונקציה להדפסת הלוח בצורה נוחה לבני אדם.  מציאת מקומות פנויים בלוח המשחק.  בדיקה האם שחקן מסויים ניצח את המשחק. |

**הרצות:**

|  |  |  |
| --- | --- | --- |
| **מקרים** |  |  |
| **תוצאות** |  |  |