|  |  |  |
| --- | --- | --- |
| **Function** | **Function Purpose** | **Trace** |
| **opponentScoresHigher** | Auxiliary function called as part of HSD Tactic. Returns a Boolean if the player’s play cannot be followed up by a higher scoring play by the opponent. | Using Hand [(6,6),(6,4),(6,0),(6,4),(4,3),(4,1),(4,0)] On Board (6,5)....(5,4)  Enemy Can Score: 4 After Player Scored: 2  Using Hand  [(3,2),(2,2),(2,1),(2,0),(6,4),(4,3),(4,1),(4,0)] On Board (2,5)....(5,4)  Enemy Can Score: 1 After Player Scored: 2  Using Hand  [(4,3),(3,2),(3,1),(3,0),(6,4),(4,3),(4,1),(4,0)] On Board (3,5)....(5,4)  Enemy Can Score: 3 After Player Scored: 0  Using Hand  [(5,5),(5,1),(5,0),(6,4),(4,3),(4,1),(4,0)] On Board (5,4)....(4,4)  Enemy Can Score: 6 After Player Scored: 0  Using Hand  [(5,5),(5,1),(5,0),(3,2),(2,2),(2,1),(2,0)] On Board (5,4)....(4,2)  Enemy Can Score: 4 After Player Scored: 0 |
| **highScoreDom :: Tactic** | A tactic which returns a single domino which scores highly to the smartPlayer.  It takes a hand and reduces it down to a set of playable moves. Then filters this list of moves down to those where it doesn’t enable the opponent to score higher on their next turn. | Calling With Hand :  [(5,1),(1,0),(5,4),(6,6)]  Opponent Could Play: [(6,5),(6,3),(6,2),(6,1),(6,0),(5,5),(5,3),(5,2),(5,0),(4,3),(4,2),(3,3),(3,1),(3,0),(2,2),(2,1),(2,0),(1,1),(0,0)]  Added To List Of Safe HSD: (3,PossMove {dom = (5,1), end = R, joinPip = 5})  Added To List Of Safe HSD: (4,PossMove {dom = (5,4), end = R, joinPip = 5})  Highest Scoring Dom: [PossMove {dom = (5,4), end = R, joinPip = 5}] |
| **dropFiveFour :: Tactic** | In the case the player has the domino (5,4) it will be returned as it is considered a good starting play. | Hand is [(5,1),(3,0),(5,0),(2,1),(6,2),(6,5),(4,1),(1,0),(4,3)] returning []  Hand is [(6,3),(5,3),(6,2),(5,4),(1,1),(4,4),(3,1),(6,6),(5,5)] returning (5,4) |
| **domWins :: Tactic** | Returns dominoes which win the game for the player on this current turn.  Attempts to play each domino on the board and returns those which give a total score of 61. | Score: 45 Possible Winning Doms: []  Player Plays (2,1) At end R  Score: 45 Possible Winning Doms: []  Player Plays (1,0) At end R  Score: 47 Possible Winning Doms: []  Player Plays (0,0) At end R  Score: 49 Possible Winning Doms: []  Player Plays (4,4) At end R  Score: 55 Possible Winning Doms: []  Player Plays (6,3) At end R  Score: 55 Possible Winning Doms: [PossMove {dom = (6,0), end = R, joinPip = 0}]  Player Plays (6,0) At end R |
| **makeFiftyNine :: Tactic** | If a domino from the hand is available to make the player’s current score up to fifty-nine then it should be played, as there are more ways to score 59 + 2 than 60 + 1 58+ 3 etc…. | From Hand: [(4,1),(6,2),(4,3),(6,6),(2,2)] On Board: (0,1) ... (0,6) With Score: 58  Player Plays (6,2) At end R  From Hand: [(4,1),(6,6),(2,2)] On Board: (2,0) ... (3,4) With Score: 58  Can Make FiftyNine With: PossMove {dom = (4,1), end = R, joinPip = 4} |
| **bestPipVal** | Returns the pips which occur more than three times in a hand.  This is an auxiliary function for the playMajority tactic, which is used to decide which pips the player has the majority of. | Player Plays (4,2) At end L  Player has hand: [(6,5),(4,4),(5,2),(3,3),(5,3),(6,2),(6,3),(6,1)]  Sorted Pips: [1,2,2,3,3,3,4,5]  Pips With Majority: [(3,3)]  Player Plays (6,1) At end R  Player has hand: [(6,5),(4,4),(5,2),(3,3),(5,3),(6,2),(6,3)]  Sorted Pips: [2,2,3,3,3,4,5]  Pips With Majority: [(3,3)]  Player Plays (6,5) At end R  Player has hand: [(4,4),(5,2),(3,3),(5,3),(6,2),(6,3)]  Sorted Pips: [2,2,3,3,3,4]  Pips With Majority: [(3,3)]  Player Plays (6,3) At end L  Player has hand: [(4,4),(5,2),(3,3),(5,3),(6,2)]  Sorted Pips: [2,2,3,3,4]  Pips With Majority: []  Player Plays (5,3) At end R  Player has hand: [(4,4),(5,2),(3,3),(6,2)]  Sorted Pips: [2,2,3,4]  Pips With Majority: []  Player Plays (3,3) At end L  Player has hand: [(4,4),(5,2),(6,2)]  Sorted Pips: [2,2,4]  Pips With Majority: []  Player Plays (5,2) At end L  Player has hand: [(4,4),(6,2)]  Sorted Pips: [2,4]  Pips With Majority: [] |
| **bestWithMajority** | This function returns plays from the list of possible plays which have one of the majority. | Pips With Majority: [(2,3)]  Player has hand: [(3,2),(0,0),(5,0),(5,2),(6,2),(6,6),(2,1),(6,5)]  Majority Plays: [PossMove {dom = (2,1), end = L, joinPip = 1}]  Pips With Majority: []  Player has hand: [(0,0),(5,0),(6,2),(6,6),(6,5)]  Majority Plays: []  Pips With Majority: [(3,3)]  Player has hand: [(6,4),(5,3),(3,1),(6,3),(2,2),(5,4),(1,0),(6,1),(3,3)]  Majority Plays: [PossMove {dom = (5,3), end = L, joinPip = 5},PossMove {dom = (5,3), end = R, joinPip = 5}] |
| **getPlaysWithPip** | This function returns PossMoves derived from the current hand with a given integer as the open pip value. | Player has hand: [(6,5),(4,4),(5,2),(3,3),(5,3),(6,2),(6,3),(6,1)]  Sorted Pips: [1,2,2,3,3,3,4,5] Pips With Majority: [(3,3)]  Player has hand: [(6,5),(4,4),(5,2),(3,3),(5,3),(6,2),(6,3)]  Sorted Pips: [2,2,3,3,3,4,5] Pips With Majority: [(3,3)]  Plays With Given Pip Value [PossMove {dom = (6,3), end = L, joinPip = 6},PossMove {dom = (6,3), end = R, joinPip = 6}]  Player has hand: [(4,4),(5,2),(3,3),(5,3),(6,2),(6,3)]  Sorted Pips: [2,2,3,3,3,4] Pips With Majority: [(3,3)]  Plays With Given Pip Value [PossMove {dom = (6,3), end = L, joinPip = 6}] |
| **playMajority :: Tactic** | Tactic which returns dominoes which when played, have a complementing domino which can be also played. The player of this domino also holds most of the value of the open pip on the played domino. | Player Plays (4,2) At end L  Player has hand: [(3,2),(0,0),(5,0),(5,2),(6,2),(6,6),(2,1),(6,5)] Majority Plays: [PossMove {dom = (2,1), end = L, joinPip = 1}]  Player has hand: [(0,0),(5,0),(6,2),(6,6),(6,5)] Majority Plays: []  Player has hand: [(6,4),(5,3),(3,1),(6,3),(2,2),(5,4),(1,0),(6,1),(3,3)] Majority Plays: [PossMove {dom = (5,3), end = L, joinPip = 5},PossMove {dom = (5,3), end = R, joinPip = 5}]  Player has hand: [(6,4),(5,3),(3,1),(6,3),(2,2),(1,0),(6,1),(3,3)] Majority Plays: [PossMove {dom = (5,3), end = L, joinPip = 5}]  Player has hand: [(6,4),(3,1),(2,2),(1,0),(6,1),(3,3)] Majority Plays: []  Player has hand: [(6,4),(3,1),(2,2),(1,0),(6,1)] Majority Plays: []  Player has hand: [(6,4),(2,2),(1,0),(6,1)] Majority Plays: []  Player has hand: [(6,4),(2,2),(6,1)] Majority Plays: [] |
| **stopOpponentWin :: Tactic** | This tactic returns dominoes which stop the opponent from winning the game if they can.  This only returns dominoes that block the opponent from winning the game. | The opponent's score is less than 53. The opponent can't win  The player can play [PossMove {dom = (6,2), end = L, joinPip = 6},PossMove {dom = (1,0), end = R, joinPip = 0},PossMove {dom = (2,0), end = R, joinPip = 0}]  By Playing: PossMove {dom = (6,2), end = L, joinPip = 6} the opponent can no longer win  The player can play [PossMove {dom = (2,1), end = L, joinPip = 2},PossMove {dom = (2,2), end = L, joinPip = 2},PossMove {dom = (2,0), end = L, joinPip = 2},PossMove {dom = (4,1), end = R, joinPip = 4}]  By Playing: PossMove {dom = (2,1), end = L, joinPip = 2} the opponent can no longer win  The player can play [PossMove {dom = (4,1), end = L, joinPip = 1},PossMove {dom = (1,0), end = L, joinPip = 1},PossMove {dom = (4,1), end = R, joinPip = 4}] |
| **opponentStitch :: Tactic** | Tactic returns dominoes which the player can play to try to stitch their opponent on an end. | Player can play [PossMove {dom = (3,2), end = L, joinPip = 2},PossMove {dom = (2,0), end = L, joinPip = 2},PossMove {dom = (4,4), end = R, joinPip = 4},PossMove {dom = (4,1), end = R, joinPip = 4}] Opponent can play: [(6,5),(6,4),(6,3),(6,2),(6,1),(6,0),(5,5),(5,2),(5,1),(5,0),(4,3),(4,2),(4,0),(3,3),(2,2),(2,1),(1,1),(1,0),(0,0)]  Player can play [PossMove {dom = (3,2), end = L, joinPip = 2},PossMove {dom = (2,0), end = L, joinPip = 2},PossMove {dom = (4,1), end = R, joinPip = 4}] Opponent can play: [(6,5),(6,4),(6,3),(6,2),(6,1),(6,0),(5,5),(5,2),(5,1),(5,0),(4,3),(4,2),(4,0),(3,3),(2,1),(1,1),(1,0),(0,0)]  Player can play [PossMove {dom = (3,2), end = L, joinPip = 2},PossMove {dom = (2,0), end = L, joinPip = 2},PossMove {dom = (5,3), end = R, joinPip = 5}] Opponent can play: [(6,5),(6,4),(6,3),(6,2),(6,1),(6,0),(5,5),(5,2),(5,1),(5,0),(4,3),(4,2),(4,0),(3,3),(2,1),(1,1),(1,0),(0,0)]  Player can play [PossMove {dom = (6,3), end = L, joinPip = 6},PossMove {dom = (0,0), end = R, joinPip = 0},PossMove {dom = (5,0), end = R, joinPip = 0}] Opponent can play: [(6,6),(6,5),(6,4),(6,2),(6,1),(5,3),(5,1),(4,3),(4,1),(4,0),(3,2),(3,1),(3,0),(2,2),(2,1),(2,0),(1,1),(1,0)]  Player can play [PossMove {dom = (3,3), end = L, joinPip = 3},PossMove {dom = (3,3), end = R, joinPip = 3}] Opponent can play: [(6,6),(6,5),(6,4),(6,3),(6,2),(6,1),(5,3),(5,1),(4,3),(4,1),(4,0),(3,2),(3,1),(3,0),(2,2),(2,1),(2,0),(1,1),(1,0)]  Player can play [PossMove {dom = (5,2), end = L, joinPip = 2},PossMove {dom = (4,2), end = L, joinPip = 2}] Opponent can play: [(6,6),(6,5),(6,4),(6,3),(6,2),(6,1),(5,3),(5,1),(4,3),(4,1),(4,0),(3,2),(3,1),(3,0),(2,2),(2,1),(2,0),(1,1),(1,0)] |
| **knockOff :: Tactic** | Returns dominoes where the player has a subsequent domino which they can play following an initial play to “knock off” their first play. | Hand: [(5,1),(5,5),(4,4),(1,0),(3,0),(3,2)]  Has Complement: PossMove {dom = (5,1), end = R, joinPip = 5}  Returns: [PossMove {dom = (5,1), end = R, joinPip = 5},PossMove {dom = (5,5), end = R, joinPip = 5}]  Hand: [(3,0),(1,0),(6,4),(4,4),(5,1),(3,3),(5,3),(6,6),(4,3)]  Has Complement: PossMove {dom = (5,1), end = L, joinPip = 5}  Has Complement: PossMove {dom = (5,3), end = L, joinPip = 5}  Has Complement: PossMove {dom = (6,4), end = R, joinPip = 4}  Has Complement: PossMove {dom = (4,4), end = R, joinPip = 4}  Has Complement: PossMove {dom = (4,3), end = R, joinPip = 4}  Returns: [PossMove {dom = (5,1), end = L, joinPip = 5},PossMove {dom = (5,3), end = L, joinPip = 5},PossMove {dom = (6,4), end = R, joinPip = 4},PossMove {dom = (4,4), end = R, joinPip = 4},PossMove {dom = (4,3), end = R, joinPip = 4}]  Hand: [(3,0),(1,0),(4,4),(5,1),(3,3),(4,3)]  Has Complement: PossMove {dom = (5,1), end = R, joinPip = 5}  Returns: [PossMove {dom = (5,1), end = R, joinPip = 5}] |
| **getOpponentDoms** | An auxiliary function which acquires a list of dominoes which a player's opponent may have. This is based on history of dominoes played, and those that are already in the current player's possession. | Initial Board Player's hand: [(1,1),(4,2),(4,0),(6,1),(6,3),(3,2),(5,2),(2,2),(4,1)] Doms The Opponent Could Have: [(6,6),(6,5),(6,4),(6,2),(6,0),(5,5),(5,4),(5,3),(5,1),(5,0),(4,4),(4,3),(3,3),(3,1),(3,0),(2,1),(2,0),(1,0),(0,0)]  Player's hand: [(1,1),(4,0),(6,1),(6,3),(3,2),(5,2),(2,2),(4,1)] Doms The Opponent Could Have: [(6,6),(6,5),(6,4),(6,2),(6,0),(5,5),(5,4),(5,3),(5,1),(5,0),(4,3),(3,3),(3,1),(3,0),(2,1),(2,0),(1,0),(0,0)]  Player's hand: [(1,1),(4,0),(6,1),(6,3),(3,2),(5,2),(4,1)] Doms The Opponent Could Have: [(6,6),(6,5),(6,4),(6,2),(6,0),(5,5),(5,4),(5,3),(5,1),(5,0),(4,3),(3,3),(3,1),(3,0),(2,1),(2,0),(1,0),(0,0)]  Player's hand: [(1,1),(4,0),(6,1),(6,3),(5,2),(4,1)] Doms The Opponent Could Have: [(6,6),(6,5),(6,4),(6,2),(6,0),(5,5),(5,4),(5,3),(5,1),(5,0),(4,3),(3,3),(3,2),(3,1),(3,0),(2,1),(2,0),(1,0),(0,0)]  Player's hand: [(1,1),(4,0),(6,1),(6,3),(4,1)] Doms The Opponent Could Have: [(6,6),(6,5),(6,4),(6,2),(6,0),(5,5),(5,4),(5,3),(5,1),(5,0),(4,3),(3,3),(3,2),(3,1),(3,0),(2,1),(2,0),(1,0),(0,0)]  Player's hand: [(1,1),(4,0),(6,1),(6,3),(4,1)]  Doms The Opponent Could Have: [(6,6),(6,5),(6,4),(6,2),(6,0),(5,5),(5,4),(5,3),(5,1),(5,0),(4,3),(3,3),(3,2),(3,1),(3,0),(2,1),(2,0),(1,0),(0,0)] |
| **getDomsFromHist** | Gets dominoes which have been played on the board already. Useful when deciding what dominoes, that the opponent might have. | Player Plays (6,3) At end R  Doms In History: [(3,3)]  Player Plays (4,3) At end L  Doms In History: [(0,4),(4,3),(3,3)]  Player Plays (2,0) At end L  Doms In History: [(2,0),(0,4),(4,3),(3,3),(3,1)]  Player Plays (2,2) At end L  Doms In History: [(2,2),(2,0),(0,4),(4,3),(3,3),(3,1),(1,5)]  Player Plays (4,2) At end L  Doms In History: [(4,2),(2,2),(2,0),(0,4),(4,3),(3,3),(3,1),(1,5),(5,6)]  Player Plays (4,4) At end L  Doms In History: [(4,4),(4,2),(2,2),(2,0),(0,4),(4,3),(3,3),(3,1),(1,5),(5,6),(6,0)]  Player Plays (6,4) At end L  Doms In History: [(6,4),(4,4),(4,2),(2,2),(2,0),(0,4),(4,3),(3,3),(3,1),(1,5),(5,6),(6,0),(0,5)] |