Assignment 2 Domino Games Design

Data Types

|  |  |
| --- | --- |
| **Data type** | **Description** |
| type Dom = (Int,Int) | Data type for a domino represented as a tuple |
| type Board = [Dom] | Data type representing a list of dominoes |
| type Hand = [Dom] | Data type representing a list of dominoes |
| data End = L|R | Data type used to specify an end of the board |
| type Move = (End,Dom) | Data type representing a move which returns a domino to play and the end to play it at |
| type DomsPlayer = Board -> Hand -> Move | A datatype which takes a function, allows for different domino players to be created. A domino player will take a hand and a board and return a move to make. |

shuffleDoms Design

Seed

defineHand design

Seed

Seed

[Dom]

shuffleDoms

Randomise the set of dominos

Hand

[Dom]

Seed

defineHand

Select N dominoes from the randomised dom set

Randomised Zipped list of (dom,int) pairs

Sorted list of (dom,int)

pairs

mergeSort

Sort the zipped list according to int value

shuffleDoms

Randomise the set of dominos

simplePlayer Design

Move

Board, Hand

(Dom, Dom)

(Dom, Dom)

getEnds

returns the two end values of the baord

Board

Board

getEnds

returns the two end values of the baord

Boolean

Boolean

Board, Dom

goesRP

Check a domino can be played right

goesLP

Check a domino can be played left

simplePlayer

get move of first playable dominoe

hsdPlayer Design

End, Board, Hand, Int, Dom

Dom

Dom

Move

Dom, Dom, Board

getHighestatEnd

gets highest scoring dom from leftdrops

[Dom]

Board, Hand

rightdrops

get list of doms that go right

[Dom]

Board, Hand

leftdrops

get list of doms that go left

compareHighest

compares two highest scoring end doms and gets the higher if there are doms that go left and right

getHighestatEnd

gets highest scoring dom from rightdrops

Move

Board, Hand

hsdPlayer

get move of highest scoring dom

hsdPlayer Design (continued)

Move, Board

compareHighest

compares two highest scoring end doms and gets the higher if there are doms that go left and right

updateBoard

update the board state after playing a dom that will go

Board

Board

Int

scoreBoard

gets 5’s and 3’s score of board

Score53

5’s and 3’s score of a number

domScore

allow for doubles at ends

Int

Int

Dom, End

hsdPlayer Design (continued)

getHighestatEnd

gets highest scoring dom from an end

Move, Board

updateBoard

update the board state after playing a dom that will go

Board

Board

Int

scoreBoard

gets 5’s and 3’s score of board

Score53

5’s and 3’s score of a number

domScore

allow for doubles at ends

Int

Int

Dom, End

playDomsRound

(Int,Int) (the accumulated scores of the players)

*Function uses let.. in.. as hands, scores and board will be initialised*

Hand, Hand, Board, DomsPlayer, DomsPlayer, Int, Int

(Int,Int) (the accumulated scores of the players)

play

Simulates the game between two players, updating their hands, scores and board after each turn played

Seed

Seed

Hand

Hand

Seed

Seed

[Dom]

[Dom]

shuffleDoms

Randomise the set of dominos

shuffleDoms

Randomise the set of dominos

defineHand2

Select N dominoes from the randomised dom set

defineHand1

Select N dominoes from the randomised dom set

DomsPlayer, DomsPlayer, Seed

playDomsRound

Play a game of five and threes between two dom players

playDomsRound (continued)

knockingP

Check if both player is knocking if so end the game

Hand, Board

possPlays

get all doms that play left and right

Hand, Board

[Dom]

[Dom]

rightdrops

get list of doms that go right

leftdrops

get list of doms that go left

Boolean

play

Simulates the game between two players, updating their hands, scores and board after each turn played

(Hand,Hand)

Board, Hand

playDomsRound (continued)

getMove

gets a players move, checks if knocking

Hand, Board, Domsplayer

play

Simulates the game between two players, updating their hands, scores and board after each turn played

Move

Boolean

Board, Hand

knockingP

Check if player is knocking if so he does not play

[Dom]

rightdrops

get list of doms that go right

Board, Hand

(Hand,Hand)

Board, Hand

possPlays

get all doms that play left and right

leftdrops

get list of doms that go left

[Dom]

playDomsRound (continued)

play

Simulates the game between two players, updating their hands, scores and board after each turn played

updateBoard

update the board state after a player move

Hand

Board

Move, Board

Move, Hand

updateHand

Update the players hand after a move

playDomsRound (continued)

Dom, End

Int

domScore

allow for doubles at ends

Int

Int

Score53

5’s and 3’s score of a number

Board

Board, Hand

Int

scoreBoard

gets 5’s and 3’s score of board

Boolean

updatePlayerScore

Update the players score after a move, verifies they did move by knocking check

Int

**See above examples for functions knocking uses**

knockingP

Check if player is knocking if so the score does not update

Board, Hand, Int

play

Simulates the game between two players, updating their hands, scores and board after each turn played