COM2001: Advanced Programming Topics

Assignment 1

Test Results

Below are the test results and the implementation of each algorithms. The test cases are logically different for each algorithm in order to test all the possible outcome of the algorithms.

Parameters: **dom** : A domino
brd : A board
hand : A hand
end : An end
n : Points to score

Algorithms

1. goesP

- **Test Case**

1. A **random domino** is given to an **empty board** on **L** (left) and **R** (right)
2. A **matching domino** is given to a **random board** on **L** (left)
3. A **matching domino** is given to a **random board** on **R** (right)
4. A **nonmatching domino** is given to a **random board** on **L** (left) and **R** (right)

Test Case	dom	brd	end	Expected Result	Actual Result	Status
1	(1,4)	[]	L	True	True	Pass
	(3,2)	[]	R	True	True	Pass
2	(5,4)	[(4,2), (2,6), (6,1)]	L	True	True	Pass
	(1,0)	[(1,2), (2,6), (6,1)]	L	True	True	Pass
	(3,3)	[(3,2), (2,6), (6,1)]	L	True	True	Pass
3	(5,3)	[(0,4), (4,1), (1,5)]	R	True	True	Pass
	(6,2)	[(0,4), (4,3), (3,2)]	R	True	True	Pass
	(0,0)	[(0,4), (4,6), (6,0)]	R	True	True	Pass
4	(3,1)	[(0,4), (4,6), (6,0)]	L	False	False	Pass
	(2,5)	[(0,4), (4,6), (6,0)]	R	False	False	Pass
	(5,5)	[(3,5), (5,6), (6,0)]	R	False	False	Pass

Table 1 Test results of goesP

2. knockingP

- **Test Case**

1. An **empty hand** is given to an **empty board**
2. A **random hand** is given to an **empty board**
3. An **empty hand** is given to a **random board**
4. A **random hand** is given to a **random board**

Test Case	hand	brd	Expected Result	Actual Result	Status
1	`[]`	`[]`	True	True	Pass
2	`[(6,1), (6,5), (3,3), (0,0)]`	`[]`	False	False	Pass
	`[(6,4), (3,2), (1,2), (2,6)]`	`[]`	False	False	Pass
3	`[]`	`[(5,2), (2,1), (1,5), (4,0)]`	True	True	Pass
	`[]`	`[(5,2), (2,4), (4,4), (4,0)]`	True	True	Pass
4	`[(6,5), (3,3), (0,0)]`	`[(5,2), (2,4), (4,4), (4,0)]`	False	False	Pass
	`[(6,4), (3,2), (1,2), (2,6)]`	`[(5,2), (2,4), (4,0)]`	True	True	Pass

Table 2 Test results of knockingP

3. playedP

- **Test Case**

1. A **random domino** is given to an **empty board**
2. An **unplayed domino** is given to a **board**
3. A **played domino** is given to a **board**

Test Case	dom	brd	Expected Result	Actual Result	Status
1	`(3,5)`	`[]`	False	False	Pass
	`(0,0)`	`[]`	False	False	Pass
2	`(4,6)`	`[(5,2), (2,4)]`	False	False	Pass
	`(5,1)`	`[(5,2), (2,4)]`	False	False	Pass
3	`(2,5)`	`[(5,2), (2,4)]`	True	True	Pass
	`(5,2)`	`[(5,2), (2,4)]`	True	True	Pass
	`(2,4)`	`[(5,2), (2,4)]`	True	True	Pass

Table 3 Test results of playedP

4. possPlays

- **Test Case**
  1. An **empty hand** is given to an **empty board**
  2. An **empty hand** is given to a **random board**
  3. A **random hand** is given to an **empty board**
  4. A **random hand** is given to a **random board**

Test Case	hand	brd	Expected Result	Actual Result	Status
1	`[]`	`[]`	`([], [])`	`([], [])`	Pass
2	`[]`	`[(5,2), (2,4), (4,4), (4,0)]`	`([], [])`	`([], [])`	Pass
	`[]`	`[(5,5)]`	`([], [])`	`([], [])`	Pass
3	`[(6,1), (6,5), (3,3), (0,0)]`	`[]`	`([(6,1),(6,5),(3,3),(0,0)], [(6,1),(6,5),(3,3),(0,0)])`	`([(6,1),(6,5),(3,3),(0,0)], [(6,1),(6,5),(3,3),(0,0)])`	Pass
	`[(5,5), (5,0), (0,4), (4,1), (1,5)]`	`[]`	`([(5,5),(5,0),(0,4),(4,1),(1,5)], [(5,5),(5,0),(0,4),(4,1),(1,5)])`	`([(5,5),(5,0),(0,4),(4,1),(1,5)], [(5,5),(5,0),(0,4),(4,1),(1,5)])`	Pass
4	`[(1,3), (2,4), (5,3), (6,0)]`	`[(5,2), (2,1), (1,6)]`	`([(3,5)], [(6,0)])`	`([(3,5)], [(6,0)])`	Pass
	`[(3,3),(2,6)]`	`[(5,2),(2,1),(1,4)]`	`([], [])`	`([], [])`	Pass

Table 4 Test results of possPlays

5. playDom

- **Test Case**

1. A **random domino** is given to an **empty board** on **L** (left) and **R** (right)
2. A **matching domino** is given to a **board** on **L** (left)
3. A **matching domino** is given to a **board** on **R** (right)
4. A **nonmatching domino** is given to a board on **L** (left) and **R** (right)

Test Case	dom	brd	end	Expected Result	Actual Result	Status
1	(3,5)	[]	L	Just [(3,5)]	Just [(3,5)]	Pass
	(3,5)	[]	R	Just [(3,5)]	Just [(3,5)]	Pass
	(6,6)	[]	R	Just [(6,6)]	Just [(6,6)]	Pass
2	(2,4)	[(4,3),(3,5),(6,2)]	L	Just [(2,4),(4,3),(3,5),(6,2)]	Just [(2,4),(4,3),(3,5),(6,2)]	Pass
	(1,4)	[(4,5),(5,3),(3,3)]	L	Just [(1,4),(4,5),(5,3),(3,3)]	Just [(1,4),(4,5),(5,3),(3,3)]	Pass
3	(2,4)	[(4,3),(3,5),(6,2)]	R	Just [(4,3),(3,5),(6,2),(2,4)]	Just [(4,3),(3,5),(6,2),(2,4)]	Pass
	(1,4)	[(2,5),(5,3),(3,4)]	R	Just [(2,5),(5,3),(3,4),(4,1)]	Just [(2,5),(5,3),(3,4),(4,1)]	Pass
4	(1,1)	[(4,3),(3,5),(6,2)]	L	Nothing	Nothing	Pass
	(6,0)	[(2,5),(5,3),(3,4)]	R	Nothing	Nothing	Pass

Table 5 Test results of playDom

6. scoreBoard

- **Test Case**

1. An **empty board** is given
2. A **random board** is given

Test Case	brd	Expected Result	Actual Result	Status
1	`[]`	0	0	Pass
2	`[(1,4),(4,5)]`	2	2	Pass
	`[(2,6),(6,5)]`	0	0	Pass
	`[(6,6),(6,3)]`	8	8	Pass
	`[(5,2), (2,4), (4,4), (4,0)]`	1	1	Pass

Table 6 Test results of scoreBoard

7. scoreN

- **Test Case**

1. An **empty board** is given
2. A **random board** is given

Test Case	brd	n	Expected Result	Actual Result	Status
1	`[]`	1	`([(0,3),(0,5),(1,2),(1,4),(2,3)], [(0,3),(0,5),(1,2),(1,4),(2,3)])`	`([(0,3),(0,5),(1,2),(1,4),(2,3)], [(0,3),(0,5),(1,2),(1,4),(2,3)])`	Pass
	`[]`	3	`([(3,6),(4,5)], [(3,6),(4,5)])`	`([(3,6),(4,5)], [(3,6),(4,5)])`	Pass
	`[]`	4	`([(6,6)], [(6,6)])`	`([(6,6)], [(6,6)])`	Pass
	`[]`	6	`([], [])`	`([], [])`	Pass
2	`[(2,3),(3,5)]`	0	`([(6,2)], [(5,0),(5,2),(5,6)])`	`([(6,2)], [(5,0),(5,2),(5,6)])`	Pass
	`[(2,3),(3,5)]`	1	`([(0,2)], [(5,1)])`	`([(0,2)], [(5,1)])`	Pass
	`[(2,3),(3,5)]`	2	`([(1,2),(5,2)], [(5,4)])`	`([(1,2),(5,2)], [(5,4)])`	Pass
	`[(2,3),(3,5)]`	3	`([(2,2),(4,2)], [])`	`([(2,2),(4,2)], [])`	Pass
	`[(2,6),(6,6)]`	8	`([(3,2)], [])`	`([(3,2)], [])`	Pass

Table 7 Test results of scoreN

Test Execution Logs

```
*Dominoes> goesP (1,4) [] L
True

*Dominoes> goesP (3,2) [] R
True

*Dominoes> goesP (5,4) [(4,2), (2,6), (6,1)] L
True

*Dominoes> goesP (1,0) [(1,2), (2,6), (6,1)] L
True

*Dominoes> goesP (3,3) [(3,2), (2,6), (6,1)] L
True

*Dominoes> goesP (5,3) [(0,4), (4,1), (1,5)] R
True

*Dominoes> goesP (6,2) [(0,4), (4,3), (3,2)] R
True

*Dominoes> goesP (0,0) [(0,4), (4,6), (6,0)] R
True

*Dominoes> goesP (3,1) [(0,4), (4,6), (6,0)] L
False

*Dominoes> goesP (2,5) [(0,4), (4,6), (6,0)] R
False

*Dominoes> goesP (5,5) [(3,5), (5,6), (6,0)] R
False
```

```
*Dominoes> knockingP [] []
True

*Dominoes> knockingP [(6,1), (6,5), (3,3), (0,0)] []
False

*Dominoes> knockingP [(6,4), (3,2), (1,2), (2,6)] []
False

*Dominoes> knockingP [] [(5,2), (2,1), (1,5), (4,0)]
True

*Dominoes> knockingP [] [(5,2), (2,4), (4,4), (4,0)]
True

*Dominoes> knockingP [(6,5), (3,3), (0,0)] [(5,2), (2,4), (4,4), (4,0)]
False

*Dominoes> knockingP [(6,4), (3,2), (1,2), (2,6)] [(5,2), (2,4), (4,0)]
True
```

```
*Dominoes> playedP (3,5) []
False

*Dominoes> playedP (0,0) []
False

*Dominoes> playedP (4,6) [(5,2), (2,4)]
False

*Dominoes> playedP (5,1) [(5,2), (2,4)]
False

*Dominoes> playedP (2,5) [(5,2), (2,4)]
True

*Dominoes> playedP (5,2) [(5,2), (2,4)]
True

*Dominoes> playedP (2,4) [(5,2), (2,4)]
True
```

```
*Dominoes> possPlays [] []
([],[])

*Dominoes> possPlays [] [(5,2), (2,4), (4,4), (4,0)]
([],[])

*Dominoes> possPlays [] [(5,5)]
([],[])

*Dominoes> possPlays [(6,1), (6,5), (3,3), (0,0)] []
([(6,1),(6,5),(3,3),(0,0)], [(6,1),(6,5),(3,3),(0,0)])

*Dominoes> possPlays [(5,5), (5,0), (0,4), (4,1), (1,5)] []
([(5,5),(5,0),(0,4),(4,1),(1,5)], [(5,5),(5,0),(0,4),(4,1),(1,5)])

*Dominoes> possPlays [(1,3), (2,4), (5,3), (6,0)] [(5,2), (2,1), (1,6)]
([(3,5)], [(6,0)])

*Dominoes> possPlays [(3,3),(2,6)] [(5,2),(2,1),(1,4)]
([], [])
```

```
*Dominoes> playDom (3,5) [] L
Just [(3,5)]

*Dominoes> playDom (3,5) [] R
Just [(3,5)]

*Dominoes> playDom (6,6) [] R
Just [(6,6)]

*Dominoes> playDom (2,4) [(4,3),(3,5),(6,2)] L
Just [(2,4),(4,3),(3,5),(6,2)]

*Dominoes> playDom (1,4) [(4,5),(5,3),(3,3)] L
Just [(1,4),(4,5),(5,3),(3,3)]
```

```
*Dominoes> playDom (2,4) [(4,3),(3,5),(6,2)] R
Just [(4,3),(3,5),(6,2),(2,4)]

*Dominoes> playDom (1,4) [(2,5),(5,3),(3,4)] R
Just [(2,5),(5,3),(3,4),(4,1)]

*Dominoes> playDom (1,1) [(4,3),(3,5),(6,2)] L
Nothing

*Dominoes> playDom (6,0) [(2,5),(5,3),(3,4)] R
Nothing
```

```
*Dominoes> scoreBoard []
0

*Dominoes> scoreBoard [(6,6)]
4

*Dominoes> scoreBoard [(2,6),(6,5)]
0

*Dominoes> scoreBoard [(6,6),(6,3)]
8

*Dominoes> scoreBoard [(5,2),(2,4),(4,4),(4,0)]
1
```

```
*Dominoes> scoreN [] 1
([(0,3),(0,5),(1,2),(1,4),(2,3)], [(0,3),(0,5),(1,2),(1,4),(2,3)])

*Dominoes> scoreN [] 3
([(3,6),(4,5)], [(3,6),(4,5)])

*Dominoes> scoreN [] 4
([(6,6)], [(6,6)])

*Dominoes> scoreN [] 6
([], [])

*Dominoes> scoreN [(2,3),(3,5)] 0
([(6,2)], [(5,0),(5,2),(5,6)])

*Dominoes> scoreN [(2,3),(3,5)] 1
([(0,2)], [(5,1)])

*Dominoes> scoreN [(2,3),(3,5)] 2
([(1,2),(5,2)], [(5,4)])

*Dominoes> scoreN [(2,3),(3,5)] 3
([(2,2),(4,2)], [])

*Dominoes> scoreN [(2,6),(6,6)] 8
([(3,2)], [])
```