

RE-ason Elevator Control Simulator

<Unit Test>

Team 01

박현규 200911393

김도현 201211328

송해찬 201211358

1. Unit Test Order

1) 검사 제외 항목

- Listener들, GUI를 위한 메소드들, Class 변수들...

2) 검사 항목

① Queue

② Movement Algorithm

③ State

④ Direction

2. Unit Test

1) Queue & Direction

①size()

Procedure	Expected Result	Test Result
a. 초기 queue의 size는 1이다. b. Pressed Button과 queue.insert는 같아야 한다. c. queue의 size가 증가한다.	queue의 사이즈 값이 나온다.	PASS

②removeMin()

Procedure	Expected Result	Test Result
a. 엘리베이터가 목표 층에 도착한다. b. queue에서 빠져나간다. c. 도착한 층의 index를 반환한다.	도착한 층의 index값이 Null값이 아니어야한다.	PASS

③getDirection()

Procedure	Expected Result	Test Result
a. 엘리베이터의 방향은 queue가 가지고 있다. b. queue에 있는 방향을 검사하여 위/아래 방향을 반환한다.	반환된 방향의 값이 Null값이 아니어야한다.	PASS

```
Finished after 0.007 seconds
Runs: 3/3      Errors: 0      Failures: 0

testQueue [Runner: JUnit 4] (0.000 s)
├─ testRemoveMin (0.000 s)
├─ testSize (0.000 s)
├─ testGetDirection (0.000 s)
└─ ...
}
queue left = new queue();

@Test
public void testSize() {
    assertEquals(left.size(), 1);
}

@Test
public void testRemoveMin() {
    left.insert(5);
    assertNotNull(left.removeMin());
}

@Test
public void testGetDirection() {
    assertNotNull(left.getDirection());
}
}
```

④enqueue(int floor)

Procedure	Expected Result	Test Result
a. 큐에 Node가 삽입되어 floor를 저장한다.	a. size 값이 증가해야한다. b. node가 삽입되어있어야 한다.	PASS

⑤min()

Procedure	Expected Result	Test Result
a. 가장 작은 값을 return	a. return값이 null이면 안된다.	PASS

⑥max()

Procedure	Expected Result	Test Result
a. 가장 큰 값을 return	a. return값이 null이면 안된다.	PASS

```

queue testqueue = new queue();
@Test
public void enqueueTest() {
    assertEquals(1, testqueue.size());
    assertNotNull(testqueue.trail);
}

@Test
public void minTest(){
    assertNotNull(testqueue.min());
}

@Test
public void maxTest(){
    assertNotNull(this.testqueue.max());
}
    
```

Runs: 3/3 Errors: 0 Failures: 0

testQueue [Runner: JUnit 4] (0.000 s)
 enqueueTest (0.000 s)
 minTest (0.000 s)
 maxTest (0.000 s)

2) Movement Algorithm

① getLeftQueue

Procedure	Expected Result	Test Result
a. MovementAlgorithm 안에서 queue를 생성 b. queue를 반환한다.	전달 받은 queue의 값이 Null값이 아니어야한다.	PASS

② getRightQueue

Procedure	Expected Result	Test Result
a. MovementAlgorithm 안에서 queue를 생성 b. queue를 반환한다.	전달 받은 queue의 값이 Null값이 아니어야한다.	PASS

② init()

Procedure	Expected Result	Test Result
a. MovementAlgorithm 안에서 queue를 생성	생성한 큐가 left,right에 들어가야 한다.	PASS

```
movementAlgorithm mA = new movementAlgorithm();
@Test
public void testInit() {
    mA.init();
    assertNotNull(mA.left);
    assertNotNull(mA.right);
}
@Test
public void testGetLeftQueue(){
    mA.init();
    assertNotNull(mA.getLeftQueue());
}
@Test
public void testGetRightQueue(){
    mA.init();
    assertNotNull(mA.getRightQueue());
}
```

Runs: 3/3 Errors: 0 Failures: 0

testMa [Runner: JUnit 4] (0.000 s)
testInit (0.000 s)
testGetRightQueue (0.000 s)
testGetLeftQueue (0.000 s)

3) State

① getState

Procedure	Expected Result	Test Result
a. 엘리베이터 object의 현재 위치를 계산한다. b. 현재 위치를 반환한다.	층의 현재 위치가 Null 값이 아니어야한다.	PASS

```
Finished after 0.042 seconds
Runs: 1/1      Errors: 0      Failures: 0
testLeftObject [Runner: JUnit 4] (0.000 s)
  testGetState (0.000 s)
leftObjectPanel l = new leftObjectPanel();
@Test
public void testGetState() {
    assertNotNull(l.getState());
}
```