**2018년 2학기 Java 프로그래밍 로보코드 보고서**

|  |  |
| --- | --- |
| 학번: 5360262  이름: 정명관 | 학과: 컴퓨터공학과  총점: 26 |
| 보고서 점수 (20점 만점)  메소드 20 개 -> 20 점 | 최종 경기 점수 (10점 만점)  ROUND 3. -> 6 점 |

**[Main Loop]**

**탱크의 주요 움직임 특징 및 공격 특징 설명**

**설명시 메소드 번호를 대괄호[ ] 속에 넣어서 설명 할 것.**

1. 본 탱크는 자신의 X좌표와[2] 자신의 Y좌표[3]를 활용하여 현재 게임의 경기장 크기로 이동거리를 계산하여[4] 처음에는 벽을 찾아서 이동합니다.[1]
2. 그리고 그 다음부터는 경기장의 크기와 탱크의 현재 X좌표와 Y좌표에 따라서 각 조건마다 몇 개의 움직임의 메소드들을[13][14][15][16] 통해 벽에 부딪치지 않게 벽에 붙어서 벽 외곽을 돌면서 이동하며 적의 공격을 피하게 됩니다.[17]
3. 만약 로봇과 부딪치게 된다면 적의 좌우 상대각도를 계산하여 뒤로 이동하거나 앞으로 이동합니다.[5]
4. 공격방법은 크게 3가지로 적은 스캔 하면 먼저 나의 탱크의 에너지 값을 구한다.[6]
5. 적의 거리가 300이상이고 적의 체력이 나보다 많다면 2의 화력으로 공격하고 적의 거리가 300 이상이고 적의 체력이 나보다 적다면 3의 화력으로 공격하고 그 외에는 1의 화력으로 공격한다.[7]
6. 만약 진행 중 벽에 부딪치면 왼쪽으로 90도 회전하여서 앞으로 100만큼 이동한다.[8]

**[각 번호별 메소드 설명]**

|  |  |  |
| --- | --- | --- |
| **No. 1** | **public void NormalMove()** | |
| Class | | Class MyRobots |
| Parameter | | 없음 (void) |
| Return | | 없음 |
| Description | | 나의 로봇의 기본적인 움직임을 가진 메소드  다음의 과정을 통해 움직임을 가진다.   1. 일단 전체 경기장 크기에서 각각의 18만큼 빼고 현재 x,y좌표를 빼주어 벽보다 안쪽으로 돌 수 있는 변수를 만든다. 2. 그리고 현재 위치에서 몸체가 향하는 각도를 90도를 나눈 나머지 만큼 왼쪽으로 회전한다. 3. 그리고 앞으로 1번의 변수만큼 움직인다. 4. 다시 오른쪽으로 움직인다. 이로 인해서 벽 외곽을 계속 돌게 된다. |
| Lines of Code (LOC) | | 5 lines |

|  |  |  |
| --- | --- | --- |
| **No. 2** | **public double MyPositionX()** | |
| Class | | Class MyRobots |
| Parameter | | 없음 (void) |
| Return | | 현재 나의 위치의 X좌표 값(double) |
| Description | | 현재 나의 위치의 X좌표 값을 활용하기 위해 만든 메소드 |
| Lines of Code (LOC) | | 1 lines |

|  |  |  |
| --- | --- | --- |
| **No. 3** | **public double MyPositionY()** | |
| Class | | Class MyRobots |
| Parameter | | 없음 (void) |
| Return | | 현재 나의 위치의 Y좌표 값(double) |
| Description | | 현재 나의 위치의 Y좌표 값을 활용하기 위해 만든 메소드 |
| Lines of Code (LOC) | | 1 lines |

|  |  |  |
| --- | --- | --- |
| **No. 4** | **public double MaxArea()** | |
| Class | | Class MyRobots |
| Parameter | | 없음 (void) |
| Return | | 내가 경기할 경기장 외곽의 크기를 계산한 값 (double) |
| Description | | 나의 로봇이 외곽을 돌기 위해 크기를 계산하여 값을 반환하는 메소드 |
| Lines of Code (LOC) | | 3 lines |

|  |  |  |
| --- | --- | --- |
| **No. 5** | **public void onHitRobot(HitRobotEvent e)** | |
| Class | | Class MyRobots |
| Parameter | | 다른 로봇이랑 부딪혔을 때 발생하는 이벤트 값 |
| Return | | 없음 (void) |
| Description | | 다른 로봇이랑 부딪혔을 때 발생하는 메소드  조건문을 사용하여 적이 나의 기준으로 상대적으로 좌우 90상에 위치하고 있다면 뒤로 100만큼 이동하고 아니라면 앞으로 100만큼 이동해서 적 로봇과 떨어진다. |
| Lines of Code (LOC) | | 6 lines |

|  |  |  |
| --- | --- | --- |
| **No. 6** | **public double MyTankHp()** | |
| Class | | Class MyRobots |
| Parameter | | 없음 (void) |
| Return | | 나의 탱크의 에너지 값을 반환(double) |
| Description | | 나의 탱크의 에너지 값을 반환하는 메소드 |
| Lines of Code (LOC) | | 1 lines |

|  |  |  |
| --- | --- | --- |
| **No. 7** | **public void onScannedRobot(ScannedRobotEvent e)** | |
| Class | | Class MyRobots |
| Parameter | | 다른 로봇이 레이더에 스캔 되었을 때 발생하는 이벤트 값 |
| Return | | 없음 (void) |
| Description | | 다른 로봇이 레이더에 스캔 되었을 때 발생하는 메소드  조건문을 사용하여 다음의 과정을 통해 공격을 한다.   1. 일단 나의 탱크 에너지를 변수에 저장한다. 2. 적의 거리가 300이상이고 에너지가 나보다 많다면 2의 화력으로 공격한다. 3. 적의 거리가 300이상이고 에너지가 나보다 적다면 3의 화력으로 공격한다. 4. 그 외에는 1의 화력으로 공격한다. |
| Lines of Code (LOC) | | 11lines |

|  |  |  |
| --- | --- | --- |
| **No. 8** | **public void onHitWallEvent(HitWallEvent e)** | |
| Class | | Class MyRobots |
| Parameter | | 나의 로봇이 벽에 부딪혔을 때 발생하는 이벤트 값 |
| Return | | 없음 (void) |
| Description | | 나의 로봇이 벽에 부딪혔을 때 벽에서 떨어지는 메소드  왼쪽으로 90도만큼 회전하여 앞으로 100만큼 이동해서 벽에서 떨어진다. |
| Lines of Code (LOC) | | 2 lines |

|  |  |  |
| --- | --- | --- |
| **No. 9** | **public void onWin(WinEvent e)** | |
| Class | | Class MyRobots |
| Parameter | | 이겼을 때 발생하는 이벤트 값 |
| Return | | 없음 (void) |
| Description | | 이겼을 때 실행이 되는 메소드, wins라는 [9]번 메소드를 호출한다. |
| Lines of Code (LOC) | | 1 lines |

|  |  |  |
| --- | --- | --- |
| **No. 10** | **public void onDeath(WinEvent e)** | |
| Class | | Class MyRobots |
| Parameter | | 패배하였을 때 발생하는 이벤트 값 |
| Return | | 없음 (void) |
| Description | | 패배하였을 때 실행이 되는 메소드, deaths라는 [10]번 메소드를 호출한다. |
| Lines of Code (LOC) | | 1 lines |

|  |  |  |
| --- | --- | --- |
| **No. 11** | **public void wins(){** | |
| Class | | Class MyRobots |
| Parameter | | 없음 (void) |
| Return | | 없음 (void) |
| Description | | 이겼을 때 메시지를 출력하는 메소드 |
| Lines of Code (LOC) | | 1 lines |

|  |  |  |
| --- | --- | --- |
| **No. 12** | **public void deaths()** | |
| Class | | Class MyRobots |
| Parameter | | 없음 (void) |
| Return | | 없음 (void) |
| Description | | 패배하였을 때 메시지를 출력하는 메소드 |
| Lines of Code (LOC) | | 1 lines |

|  |  |  |
| --- | --- | --- |
| **No. 13** | **public void noHitWallMove1()** | |
| Class | | Class MyRobots |
| Parameter | | 없음 (void) |
| Return | | 없음 (void) |
| Description | | 나의 로봇이 경기장 외곽을 부딪히고 돌게 되는 첫 번째 움직임의 메소드  다음의 과정을 통해 움직임을 가진다.   1. 앞으로 경기장크기에서 나의 위치의 Y좌표값을 빼고 18만큼 빼준 거리만큼 움직인다. 2. 오른쪽으로 90도 회전한다. |
| Lines of Code (LOC) | | 2 lines |

|  |  |  |
| --- | --- | --- |
| **No. 14** | **public void noHitWallMove2()** | |
| Class | | Class MyRobots |
| Parameter | | 없음 (void) |
| Return | | 없음 (void) |
| Description | | 나의 로봇이 경기장 외곽을 부딪히고 돌게 되는 두 번째 움직임의 메소드  다음의 과정을 통해 움직임을 가진다.   1. 앞으로 경기장크기에서 나의 위치의 X좌표값을 빼고 18만큼 빼준 거리만큼 움직인다. 2. 오른쪽으로 90도 회전한다. |
| Lines of Code (LOC) | | 2 lines |

|  |  |  |
| --- | --- | --- |
| **No. 15** | **public void noHitWallMove3()** | |
| Class | | Class MyRobots |
| Parameter | | 없음 (void) |
| Return | | 없음 (void) |
| Description | | 나의 로봇이 경기장 외곽을 부딪히고 돌게 되는 세 번째 움직임의 메소드  다음의 과정을 통해 움직임을 가진다.   1. 앞으로 나의 위치의 Y좌표값에서 18만큼 빼준 거리만큼 움직인다. 2. 오른쪽으로 90도 회전한다. |
| Lines of Code (LOC) | | 2 lines |

|  |  |  |
| --- | --- | --- |
| **No. 16** | **public void noHitWallMove4()** | |
| Class | | Class MyRobots |
| Parameter | | 없음 (void) |
| Return | | 없음 (void) |
| Description | | 나의 로봇이 경기장 외곽을 부딪히고 돌게 되는 네 번째 움직임의 메소드  다음의 과정을 통해 움직임을 가진다.   1. 앞으로 나의 위치의 X좌표값에서 18만큼 빼준 거리만큼 움직인다. 2. 오른쪽으로 90도 회전한다. |
| Lines of Code (LOC) | | 2 lines |

|  |  |  |
| --- | --- | --- |
| **No. 17** | **public void noHitWall()** | |
| Class | | Class MyRobots |
| Parameter | | 없음 (void) |
| Return | | 현재 나의 위치의 Y좌표값(double) |
| Description | | 로봇이 경기장 외곽을 부딪히고 움직이게 되는 핵심 메소드  조건문을 사용하여 각각의 조건에 따라 다음의 과정을 통해 움직임을 가진다.   1. 현재 나의 X좌표가 30보다 작거나 같으면서 Y좌표도 30보다 작거나 같다면 **noHitWallMove1()**를 호출하여 움직인다. 2. 현재 나의 X좌표가 30보다 작거나 같고 Y좌표가 1170보다 크거나 같다면 **noHitWallMove2()**를 호출하여 움직인다. 3. 현재 나의 X좌표가 1170보다 크거나 같고 Y좌표가 1170보다 크거나 같으면 **noHitWallMove3()**를 호출하여 움직인다. 4. 현재 나의 X좌표가 1170보다 크거나 같고 Y좌표가 30보다 작거나 같으면 **noHitWallMove4()**를 호출하여 움직인다. 5. 현재 나의 X좌표가 30보다 작거나 같으면 **noHitWallMove1()**를 호출하여 움직인다. 6. 현재 나의 Y좌표가 1170보다 크거나 같다면 **noHitWallMove2()**를 호출하여 움직인다. 7. 현재 나의 X좌표가 1170보다 크거나 같다면 **noHitWallMove3()**들 호출하여 움직인다. 8. 마지막으로 위의 조건이 아닌 경우에는 **NormalMove()** 메소드를 호출하여 나의 탱크의 기본 움직임으로 움직인다. |
| Lines of Code (LOC) | | 27 lines |

|  |  |  |
| --- | --- | --- |
| **No. 18** | **public void onHitByBullet(HitByBulletEvent e)** | |
| Class | | Class MyRobots |
| Parameter | | 다른 로봇의 총알에 맞았을 때 발생하는 이벤트 값 |
| Return | | 없음 (void) |
| Description | | 다른 로봇의 총알에 맞았을 때 발생되는 메소드  어떤 탱크의 이름이 날 공격했는지 출력한다. |
| Lines of Code (LOC) | | 1 lines |

|  |  |  |
| --- | --- | --- |
| **No. 19** | **public void MyBodyColor()** | |
| Class | | Class MyRobots |
| Parameter | | 없음 (void) |
| Return | | 없음 (void) |
| Description | | 나의 탱크의 몸체의 색을 정하는 메소드 |
| Lines of Code (LOC) | | 1 lines |

|  |  |  |
| --- | --- | --- |
| **No. 20** | **public void MyGunColor()** | |
| Class | | Class MyRobots |
| Parameter | | 없음 (void) |
| Return | | 없음 (void) |
| Description | | 나의 탱크의 총과 총알의 색을 정하는 메소드 |
| Lines of Code (LOC) | | 2 lines |