인공지능

과제 1

과제 개요

- 주제: “지뢰 찾기” 게임 해결을 위한 전문가 시스템 구현
  - 시스템의 구성:
    1. 오라클 시스템
       - 과제에서 주어지는 시스템으로, 지뢰 찾기 맵을 생성.
       - 만들어진 맵을 토대로 전문가 시스템과 통신하며 맵을 업데이트, 게임 오버 및 클리어 상황 반환.
    2. 전문가 시스템
       - 과제로 구현해야 하는 시스템
       - 오라클과 통신하며 오라클에서 건내준 정보를 토대로 자신의 맵을 업데이트, 그 정보들에 기반하여 지뢰를 판단해야 함

과제 평가

- 주제: “지뢰 찾기” 게임 해결을 위한 전문가 시스템 구현
  - 과제 평가: 오라클이 보유한 여러 장의 지도에 대하여 구현 프로그램이 성공적으로 추정한 지뢰의 개수를 점수로 사용.
    - 점수 척도: (시스템이 발견한 지뢰 수 / 총 존재하는 지뢰 수)
      - Ex) 시스템 발견 지뢰수: 15개, 총 지뢰수 40개 == 15/40 → 0.375
    - 점수 부여
      - 성능이 20%넘는 사람들 중에서 상위 50%: 만점
      - 성능이 20%넘는 사람들 중에서 상위 50 ~ 60%: 90점
      - 성능이 20%넘는 사람들 중에서 상위 60 ~ 70%: 80점
      - 그 외: 70점 (단, 20% 이상의 성능이 나왔을 시, 20% 이하는 50점)
      - 동작하지 않는 코드 제출 및 성능이 2% 이하 (8x8에서 Random으로 찍을 시 이정도 나옴)일 시 0점

오라클의 동작 및 설명

- 입력
  - 전문가 시스템이 게임 보드를 보고 예상하는 (혹은 맨 처음 상태에서 고른) 지뢰 보드 좌표 (x, y)와 해당 좌표의 지뢰 존재 여부를 받음.

  - 지뢰 존재 여부로 0을 부여해주면 지뢰가 해당 좌표에 없다는 정보를 주는 것이고, 1을 부여해주면 지뢰가 해당 좌표에 존재 한다는 것을 나타냄

오라클의 동작 및 설명

- 출력
  - 입력으로 들어온 좌표에 지뢰가 있는지 없는지 판단.
  - 만약 제대로 판별했을 시 Coordinate Type의 ArrayList를 반환.
  - Coordinate Instance는 해당 x, y축과 그 좌표의 값을 저장하고 있음 (Coordinate 설명 참고)
    - 만약 그 좌표 주위에 아무것도 없다면 0, 주변에 지뢰가 있을 시에는 지뢰 수, 지뢰일 경우에는 99를 반환.

- 게임 종료
  1. 게임 보드에 존재하는 모든 지뢰를 찾음 (게임 클리어)
  2. 잘못된 판단을 함 (지뢰가 없는 곳에 지뢰 표시, 또는 그 반대 -> 게임 오버)
  3. 이미 확인된 부분을 다시 접근함 (게임 오버)

오라클의 동작 및 설명

- 동작
  1. Oracle 초기화 (Map 초기화)
  2. 입력 받음 (x, y, 지뢰 여부)
  3. 입력 좌표에 따른 출력 반환 (또는 Game Over 출력)
     1. Game Over 되지 않았을 시, ArrayList를 반환하고 2-3 반복
     2. Game Over 시 점수 저장
  4. 새 게임 입력 기다림

과제 설명

- 평가용 지뢰 지도의 구성
  - 주어진 Oracle의 경우 Random으로 지뢰 찾기 게임 보드를 생성함.
  - 평가 시에는 따로 정해진 지도를 사용
  - 총 지도의 개수는 5개로 8x8 2개, 16x16 2개로 예정
  - 32x32의 경우, 시간상 불가능 하다고 판단하여 제외.
    - 단, 16x16에 대한 점수가 다른 사람의 점수와 같을 시에만 사용 예정

- 평가
  - 제출한 전문가 시스템을 실행하여 적절한 수준의 성능이 3번 나올 때까지 실행
  - 그 점수들의 평균을 기반으로 점수를 채점
    - 단, 모든 학생들의 평균 성적이 낮을 경우, 가장 잘 나온 점수만을 과제 점수에 반영

과제 설명

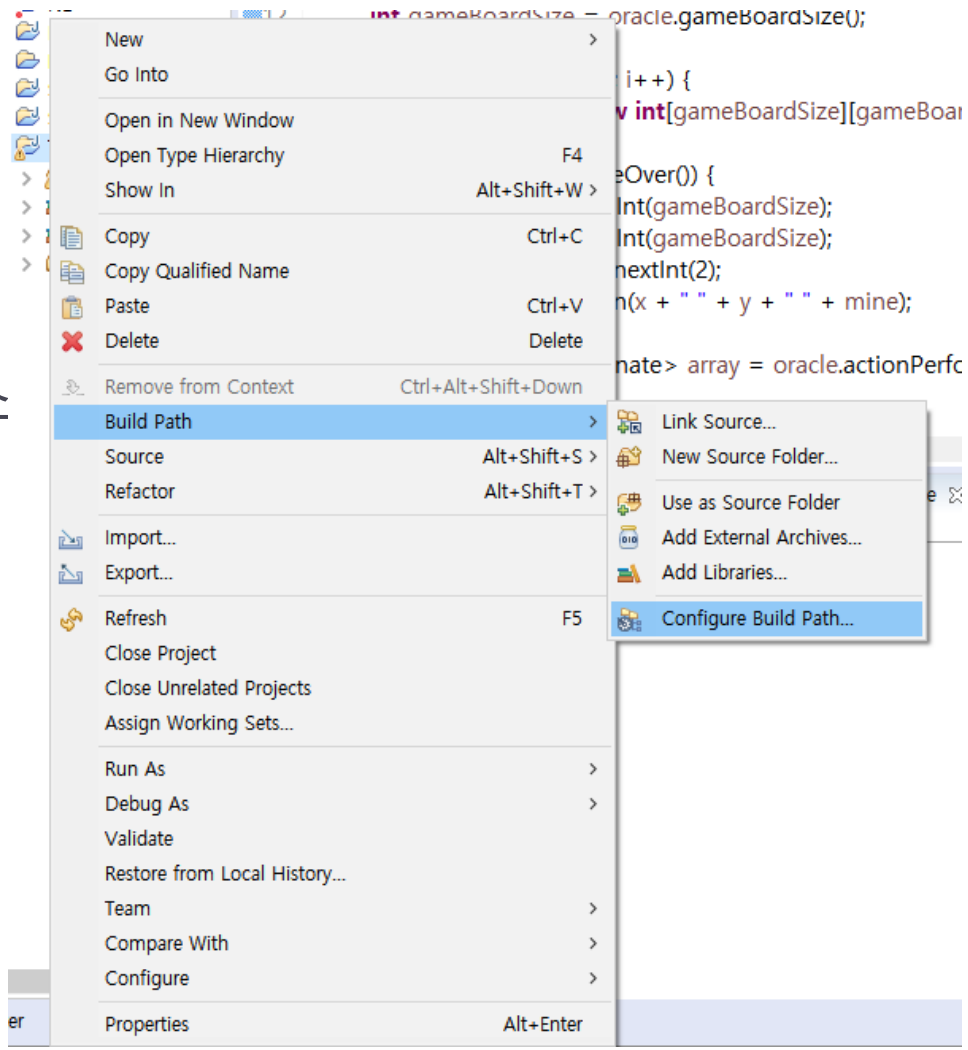
- 제출마감일
  - 과제 설명일 4주 후 (9월 21일 – 10월 19일)
- 제출 방법
  - 구현 프로그램과 소스코드(소스 코드 필수!)를 담당 조교에게 이메일 (alstn9173@kangwon.ac.kr)로 제출.
- 제출 형식
  - [인공지능 과제1]학번_이름
- 주의: 성능이 동일한 프로그램이 발생할 경우 동일한 성능을 보고한 프로그램의 소스 코드를 점검할 것임.

과제 설명

- 제출 파일
  1. 구현한 전문가 시스템의 소스 코드
  2. 보고서 파일
     - 겉 표지 필요 없음
     - 출력하여 제출하는 것이 아닌 이메일을 통한 제출
     - 들어가야 할 내용들
       1. 학번, 이름
       2. 전문가 시스템 구현 논리
       3. 주어진 Board 1 ~ 5 file들에 대한 결과
- 위의 파일들을 압축하여 제출 요망
- 미리 제출 시에 다른 메일에 밀려 빼먹을 가능성도 있으므로, 제출은 되도록 기한에 맞추어 (10/19 -1주일) 제출하는 것을 권장

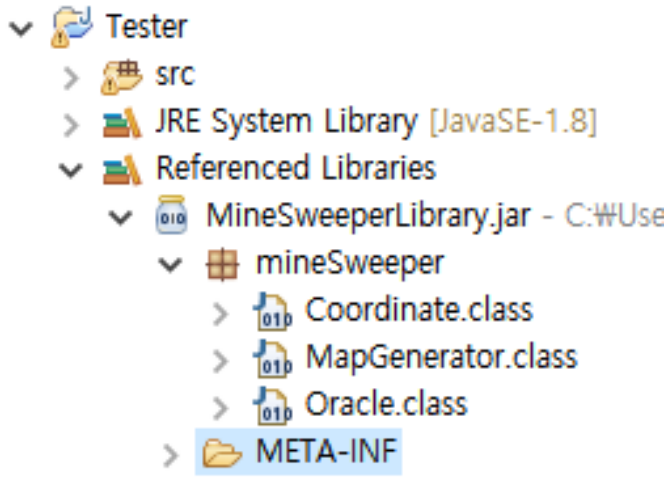
Oracle Library (1/11)

- 라이브러리 사용법
  - 주어진 library file 준비 (MineSweeperLibrary.jar)
  - 프로젝트 – 오른쪽 마우스 클릭 – Build Path – Configure Build Path



Oracle Library (2/11)

- 라이브러리 사용법
  - Libraries 창에서 Add External JARs… 선택
  - MineSweeperLibrary.jar 선택
  - Apply
  - 우측 그림처럼 Referenced Libraries가 뜨면 완료



Oracle Library (3/11)

- 본 실습을 위해서 구현된 Library로, 좌표와 지뢰 여부를 받아 게임 오버 또는 ArrayList를 반환

- 사용 가능 Method
  - restart() – 게임 재 시작
  - restartWithSameMap() – 현재 플레이 한 맵 그대로 재시작
  - isGameOver() – 게임 오버 여부 반환 (Boolean Type)
    game over 시 true, 아니면 false
  - getB() – 게임 보드 크기 반환 (8, 16, 32, or custom)
  - printScore() – 현재까지 플레이한 게임들의 점수 반환
  - actionPerform(x, y, mine) – 입력 Method로 출력으로 ArrayList 반환
  - currentState() – 현재 게임 보드 상태 출력 (자신의 프로그램디버그 용, 배열 반환 안 함!)
  - setMap(filePath) – 해당 filePath에 있는 파일을 읽어 map 초기화

Oracle Library (4/11)

- 사용법
  - import mineSweeper.*;
  - Oracle oracle = new oracle();

- Constructor
  - 총 4개의 Consturctor 존재
  1. Oracle() – 기본 Consturctor, log 출력, board size: 8x8
  2. Oracle(false) – log 출력 disable
  3. Oracle(0) – Stage 지정
     (**0**: 8x8, **1**: 16x16, **2**: 32x32)
  4. Oracle(0, false) – 둘 다 지정

Oracle Library (5/11)

- restart()
  - 게임 재 시작 Method
  - 게임 오버 및 클리어 후에는 자동으로 재 시작되지 않으므로 본 Method를 호출하여 재시작 시켜주어야 함

- restartWithSameMap()
  - 현재 플레이 했던 맵 그대로 실행

Oracle Library (6/11)

- isGameOver()
  - 현재 게임 상태를 반환 (Boolean type)
  - 게임 오버 시에는 ture, 그렇지 않을 시 false 반환

- getBoardSize()
  - 현재 게임 보드의 크기를 가져옴 (integer type)

Oracle Library (7/11)

- printScore()
  - 지금까지 진행했던 모든 게임들의 Score를 출력
  - 출력 결과가 0.09999999999999998 같은 형태로 나올 수 있으나, 이는 컴퓨터가 실수 처리를 제대로 하지 못하기에 발생하는 오류로써 근사값인 0.1 이라고 생각할 것!
  - 출력 결과는 다음과 같음

```
Attempt 0: 0.0
Attempt 1: 0.0
Attempt 2: 0.0
Attempt 3: 0.0
Attempt 4: 0.09999999999999998
Attempt 5: 0.0
```

Oracle Library (8/11)

- currentState()
  - 현재 보드 상태를 콘솔에 출력
  - 본 메소드는 자신의 프로그램이 정확히 작동하고 있는지 여부를 알아보기 위한 메소드로써, 디버그용임
  - 오른쪽과 같은 형태로 출력
  - 현재 보드 상태 배열은 반환하지 않음

```
   0  1  2  3  4  5  6  7
 -------------------------
0 |??|??|??|??|??|??| 1|  |
 -------------------------
1 |??|??|??|??|??|99| 1|  |
 -------------------------
2 |??|??|??|??|??| 2| 1|  |
 -------------------------
3 |??|??|??|??|??| 1|  |  |
 -------------------------
4 |??|??|??|??|??| 1| 1|  |
 -------------------------
5 |??|??|??|??|??|99| 2| 1|
 -------------------------
6 |??|??|??|??|??|??|??|??|
 -------------------------
7 |??|??|??|??|??|??|??|??|
 -------------------------
```

Oracle Library (9/11)

- setMap(String filePath)
  - 파일 경로를 받아 해당 경로의 파일을 읽어 맵을 생성
  - 되도록 스테이지에 맞는 데이터를 주는걸 권장
    (stage – stage 0: 8x8 10개, stage 1: 16x16 40개, stage 2: 32x32 150개)
  - 사용 예: oracle.setMap(“data/Board 1.txt”);
    - 프로젝트 내의 data folder에 있는 Board 1.txt를 토대로 맵을 생성

Oracle Library (10/11)

- actionPerform(int x, int y, int mine)
  - 입력을 받아 그에 해당하는 출력을 반환하는 메소드
  - 출력 Type은 ArrayList<Coordinate> 임 (Coordinate 설명 참고)

  - 반환 결과
    1. 제대로 된 입력을 집어넣음
       - 입력이 빈칸임 -> 그 빈칸 주변에 있는 모든 정보들
       - 입력 좌표가 숫자 또는 지뢰를 정확히 판단 -> Size가 1인 ArrayList
    2. 이번 입력으로 인해 Game Over가 되었음
       - 빈 ArrayList 반환
    3. 잘못된 입력 (입력 좌표가 배열 크기 초과 또는 mine에 0 또는 1 이외의 값을 집어 넣음)
       - Null값 반환 <- 따라서 이럴 경우 조심할 것!!
    4. 게임 오버 시에 또 다시 행동을 하려고 함
       - Null값 반환

Oracle Library (11/11)

- Coordinate Class
  - Field: int x, int y, int value
  - Constructor: new Coordinate(int x, int y, int value);
  - Methods: getX() - return x
    getY() - return y
    getValue() - return value
    equals(Object c) - c와 비교하여 같은지 판단
    toString()

  - Value값 설명
    - 해당 좌표 주위에 지뢰 없음: 0
    - 해당 좌표 주위에 지뢰 있음: [주위에 있는 지뢰 수]
    - 해당 좌표가 지뢰임: 99