객체지향 프로그래밍 실습 과제

5주차. 복잡한 Class and Objects

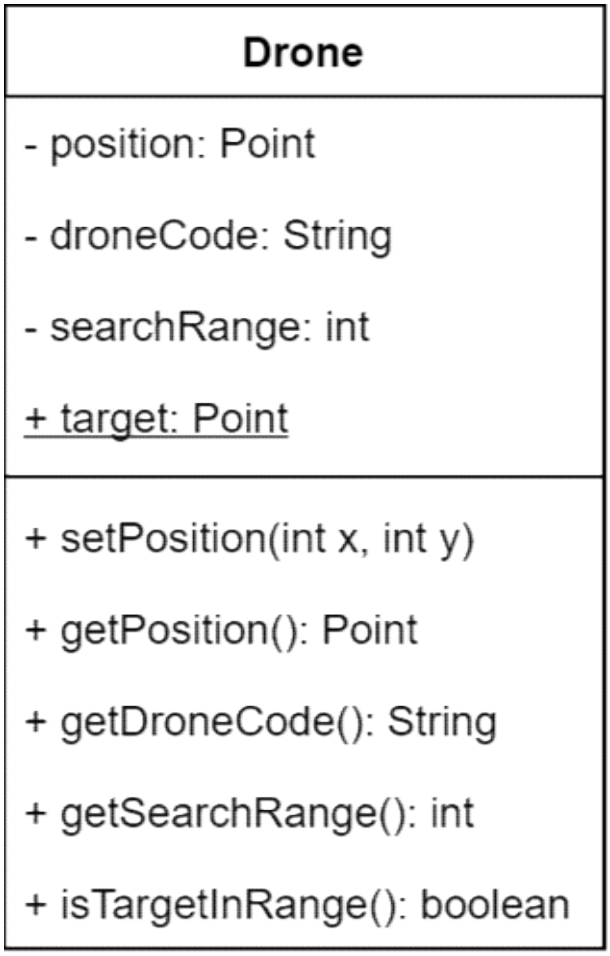
과제 개요

4주차 실습과제를 업그레이드하여 Composition, static member 등 여러 구조를 적용해보자.

클래스 설명에 따로 제한사항이 주어지지 않는 한, 메서드 또는 필드변수를 자유롭게 선언 및 활용할 수 있다.

과제 세부사항

1. Drone class (40%)



1. position 멤버는 Drone의 현재 위치를 담고 있다.
2. droneCode는 Drone의 관제코드를 의미한다.
3. searchRange는 Drone의 적군 탐색 가능 범위를 의미한다.
4. target는 Drone이 추적해야 할 목표의 위치로써, 모든 객체가 같은 target을 가진다.
5. 필드변수를 초기화 할 수 있도록 생성자를 구현한다.
6. Drone 클래스의 getters, setters는 diagram에 주어진 메서드 외에 구현하지 않는다.
7. setPosition(int x, int y)는 드론의 현재 위치를 변경하기 위한 기능이다.
8. isTargetInRange()는 Drone의 searchRange 범위 내에 target이 위치해 있는지 여부를 반환한다. DroneUtil.calculateDistance() 메서드를 활용한다.

2. DroneManager class (20%)

DroneManager
- droneList: List<Drone>
+ addDrone(Drone drone) + removeDrone(String droneCode) + removeDrone(int index) + getDrone(String droneCode): Drone + getDrone(int index): Drone + getDroneList(): List<Drone> + setTarget(int positionX, int positionY)

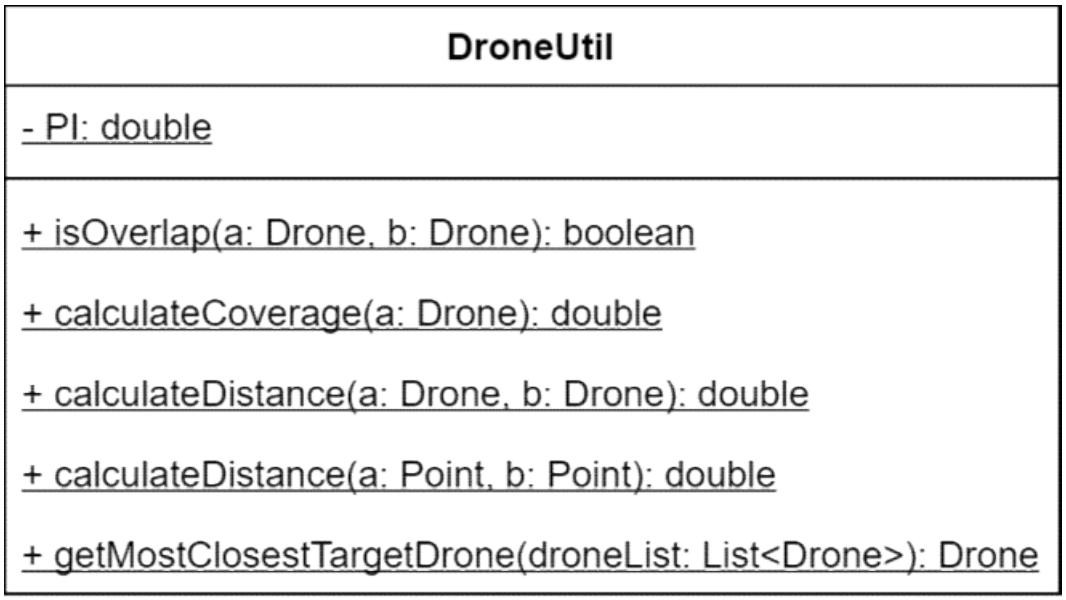
1. Drone을 여러 개 담을 수 있도록 리스트를 필드로 갖는다.
2. addDrone(Drone drone) 메서드는 drone 객체를 droneList에 추가하는 역할을 수행한다.
3. removeDrone(...) 메서드는 droneList에 추가된 drone 객체를 삭제하는 역할을 수행한다.
    1) removeDrone(String droneCode)은 droneList에서 droneCode가 동일한 객체를 리스트에서 삭제한다.
    2) removeDrone(int index)는 droneList에서 index 번째에 위치한 객체를 리스트에서 삭제한다.
4. getDrone(...)는 droneList에 있는 drone객체 중 하나를 반환하는 역할을 수행한다.
    1) getDrone(String droneCode)는 droneCode가 동일한 객체를 droneList에서 찾아, 이를 반환한다.
    2) getDrone(int index)는 droneList에서 index 번째에 위치한 객체를 반환한다.
5. setTarget() 메서드는 Drone.target 클래스변수의 값을 재설정한다.

3. Point class (10%)

Point
- positionX: int - positionY: int
+ setPosition(x: int, y: int) + getPositionX(): int + getPositionY(): int

1. Point는 2차원 좌표계를 표현하기 위한 클래스로써, positionX와 positionY 값을 가진다.
2. 필드변수를 초기화 할 수 있도록 생성자를 구현한다.

4. DroneUtil class (20%)



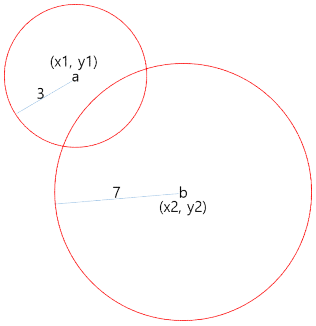
1. DroneUtil은 Drone과 Point와 관련하여 여러 가지 연산을 지원하기 위한 클래스이다.
2. isOverlap() 메서드는 Drone a와 b의 searchRange가 겹치는지 아닌지를 반환한다.
3. calculateCoverage() 메서드는 searchRange를 반지름으로 가지는 원의 넓이를 반환한다.
4. calculateDistance()는 Drone a와 b 또는 Point a와 b 사이의 거리를 반환한다.
5. getMostClosestTargetDrone() 메서드는 droneList에 담겨있는 Drone 중에서 target과 가장 가까운 거리에 위치한 Drone 객체를 반환한다.
6. PI 변수는 3.14159265359로 초기화 한다.
7. DroneUtil 클래스의 모든 필드와 메서드는 static 타입이다.
8. DroneUtil 클래스 외부에서 DroneUtil 클래스의 메서드를 호출할 때에는 static import를 활용한다.

5. DroneTest class (10%)

1. main 메소드를 생성하여 아래 사항이 작동할 수 있도록 한다.
2. 아래 출력 예시와 같은 기능을 수행할 수 있도록 구현한다.

주의 사항

1. droneCode가 중복되지 않는다고 가정한다.
2. 소스코드가 들어있는 Eclipse 프로젝트 폴더와 실행결과 캡처 사진을 압축하여 제출한다.
3. 압축파일 명은 "학번_이름_HW05"으로 한다.
4. Java code convention(Camel case 등)을 준수하고 간단한 주석을 작성한다.
5. 프로그램 종료(8번 선택지)를 입력하기 전까지 프로그램은 계속 실행되고 있어야 한다.
6. Eclipse Preference에서 텍스트파일 인코딩을 UTF-8로 설정한다. (미흡 시 감점)
7. isOverlap()메서드는 아래처럼 searchRange가 각각 3, 7인 Drone의 원이 겹치는 경우를 true로 판정한다.



출력 예시

```
1. 드론 추가
2. 드론 삭제
3. 드론 위치 변경
4. 드론 조회
5. 타겟 지정
6. 타겟과 가장 가까운 드론 검색
7. 드론 상호 조회
8. 프로그램 종료
선택: 1

drone code: Lightning
position x: 10
position y: 3
search range: 5

1. 드론 추가
2. 드론 삭제
3. 드론 위치 변경
4. 드론 조회
5. 타겟 지정
6. 타겟과 가장 가까운 드론 검색
7. 드론 상호 조회
8. 프로그램 종료
선택: 1

drone code: Falcon
position x: 7
position y: 6
search range: 4
```

드론을 추가할 때 search range를 받아야함.
드론 삭제, 드론 위치 변경은 지난 주차 과제와 동일

```
1. 드론 추가
2. 드론 삭제
3. 드론 위치 변경
4. 드론 조회
5. 타겟 지정
6. 타겟과 가장 가까운 드론 검색
7. 드론 상호 조회
8. 프로그램 종료
선택: 4

drone code:  Lightning  x: 10  y:  3  search range: 5

drone code:     Falcon  x:  7  y:  6  search range: 4
```

드론 조회

```
1. 드론 추가
2. 드론 삭제
3. 드론 위치 변경
4. 드론 조회
5. 타겟 지정
6. 타겟과 가장 가까운 드론 검색
7. 드론 상호 조회
8. 프로그램 종료
선택: 5
position x: 6
position y: 8
```

타겟 지정

```
1. 드론 추가
2. 드론 삭제
3. 드론 위치 변경
4. 드론 조회
5. 타겟 지정
6. 타겟과 가장 가까운 드론 검색
7. 드론 상호 조회
8. 프로그램 종료
선택: 6

drone code:     Falcon  x:  7  y:  6  search range: 4

distance: 2.236
```

6번 선택 시, 타겟과 가장 가까운 드론에 대한 정보와 둘 간의 거리를 출력한다.

```
1. 드론 추가
2. 드론 삭제
3. 드론 위치 변경
4. 드론 조회
5. 타겟 지정
6. 타겟과 가장 가까운 드론 검색
7. 드론 상호 조회
8. 프로그램 종료
선택: 7

First drone code: Falcon

Second drone code: Lightning
distance: 4.243    Both are overlap
```

```
1. 드론 추가
2. 드론 삭제
3. 드론 위치 변경
4. 드론 조회
5. 타겟 지정
6. 타겟과 가장 가까운 드론 검색
7. 드론 상호 조회
8. 프로그램 종료
선택: 3

drone code: Lightning
position x: 100
position y: 100

1. 드론 추가
2. 드론 삭제
3. 드론 위치 변경
4. 드론 조회
5. 타겟 지정
6. 타겟과 가장 가까운 드론 검색
7. 드론 상호 조회
8. 프로그램 종료
선택: 7

First drone code: Falcon

Second drone code: Lightning
distance: 132.231    Both are not overlap
```

7번 선택 시, 상호 비교할 드론의 drone code를 입력받고, 둘 간의 거리와 overlap 여부를 출력한다.

제출 기한

1. 일요일 23:59 까지: 100%
2. 월요일 23:59 까지: 70%
3. 이외: 0%