Object-Oriented Programming
Term Project Proposal: Object-Oriented Design

TA : 최현식 (hyunsikchoi@korea.ac.kr)
조용현 (yonghyun@korea.ac.kr)

Topic
Object-Oriented Design의 기본이 되는 클래스 구조를 직접 작성함으로써, 수업의 핵심 개념을 이해한다.

Background
OOP의 3대 특성

- Inheritance (chapter 8)
- Encapsulation (chapter 5)
- Polymorphism (chapter 9)

Objective
보드 게임용 (체스 또는 장기) 오브젝트들을 디자인 하고 JAVA 언어로 작성하라.

Requirement
1. **한 개 이상의 최상위 클래스를 작성한다.**

(1) 필수적인 변수를 작성한다. (Essential Variable)

- private int x ;
- private int y ;
- private identifier;

(2) 필수적인 메소드를 작성한다. (Essential Method)

- private Move()
- private Attack()

최상위 클래스를 작성하여 일반적인 자료(오브젝트의 위치)와 메소드(이동, 공격)등을 선언한다.

2. 5~7개의 상속 클래스를 작성한다.

(1) 원하는 변수를 작성한다. (예: 에너지, 성격, 공격력, 돈 등등)

(2) 원하는 메소드를 작성한다. (도주, 무례한 행동하기, 먼산 등등)

최상위 클래스의 상속 클래스들을 작성하여, 스스로 고안한 자료와 메소드등을 작성한다.

3. Board class를 작성 한다.

(1) 1, 2에서 작성한 클래스들의 인스턴스를 선언할 board class를 작성한다.

(2) 2차원 배열을 사용해서, 보드를 작성한다.

- private int map[10][10];

Board class에는 main 함수가 위치하며, 작성한 오브젝트들의 메소드들을 이곳에서 수행한다.

Example

(1) 최상위 클래스

```
import java.util.List;

public abstract class AbstractHorse {
    private int x;
    private int y;
    private int id;
    private Rank rank;
    private Board board;

    public AbstractHorse(Board board, Rank rank, int id, int x, int y) {
        this.board = board;
        this.rank = rank;
        this.id = id;
        this.x = x;
        this.y = y;
    }

    abstract public void move(int x,int y);
    abstract public List<Integer> getMoveablePosition();
    abstract public void attack(int x, int y);

    public int [] getPosition() {
        return new int [] {x,y};
    }

    public void setPosition(int x,int y) {
        this.x = x;
        this.y = y;
    }

    public static enum Rank {
        SWORDMAN,
        QUEEN
    };
}
```

- 최상위 클래스인 AbstractHorse를 정의한다.
- 자신의 위치를 정의하는 x , y를 선언한다.
- 인스턴스를 구분하는 id를 선언한다.
- 상속받는 클래스의 종류를 구분하는 Rank를 선언한다.
- main 메소드가 정의되어있는 board 클래스와의 연동을 위해서 board를 선언한다.

- AbstractHorse의 생성자를 정의한다.
- 자신의 위치를 이동시키는 추상 메소드 move를 선언한다.
- 다른 인스턴스를 공격하는 추상 메소드 attack을 선언한다.

- 자신의 위치를 반환하는 getPosition함수를 정의한다.
- 자신의 위치를 설정하는 setPosition함수를 정의한다.

- 상속받는 클래스의 종류에 대한 열거형(enum)을 정의한다.

(2) 상속 클래스

```java
import java.util.List;
/**
 *
 */

/**
 * @author c0d3h4ck
 *
 */
public class SwordMan extends AbstractHorse {

    /**
     * @param identity
     * @param position
     */
    public SwordMan(Board board, int id, int x, int y) {
        super(board, Rank.SWORDMAN, id, x, y);
    }

    @Override
    public List<Integer> getMoveablePosition() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public void move(int x, int y) {
        // TODO Auto-generated method stub

    }

    public void attack(int x, int y) {

    }

}
```

- 최상위 클래스인 AbstractHorse의 상속 클래스인 SwordMan을 정의한다.
- SwordMan의 생성자를 정의한다.

- AbstractHorse로부터 상속받은 getMoveablePosition 메소드의 기능을 정의한다.
- AbstractHorse로부터 상속받은 move 메소드의 기능을 정의한다.
- AbstractHorse로부터 상속받은 attack 메소드의 기능을 정의한다.

◈ (예제에서는 모두 미 구현 되어있음)

(3) Board 클래스

```
public class Board {
    private AbstractHorse [][] plane;
    public Board() {
        plane = new AbstractHorse[10][10];

        plane[0][0] = new SwordMan(this,0,0,0);
        plane[9][0] = new SwordMan(this,1,9,0);
    }

    public boolean menu() {
        System.out.println("1. Move");
        System.out.println("2. Attack");
        System.out.println("3. Quit");

        return false;
    }

    public static void main(String [] args) {
        Board board = new Board();
        boolean isContinue;

        do {
            isContinue = board.menu();

        } while(isContinue == true);
    }
}
```

- main 메소드가 있는 Board 클래스를 정의한다.
- AbstractHorse 클래스의 10 X 10 2차원 배열을 선언하여 보드의 격자를 정의한다.
- SwordMan을 (0,0) 과 (0,9) 에 배치 한다.

- 사용자 인터페이스로 이용할 수 있는 menu 메소드를 정의한다. (미 구현)
- 메인메소드를 정의한다.