소프트웨어분석설계

6주차 과제

팀 H

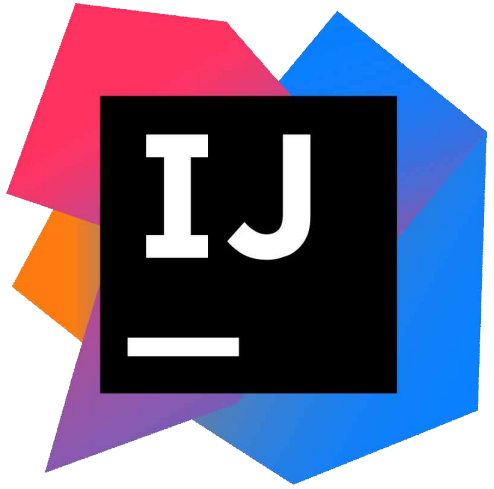
제출일 : 2023.04.13

팀원 : 20180662 장준영

20170677 오융택

1. 실습 진행 환경

다이어그램 실습 환경 : StarUML



자바 실습 환경 : IntelliJ

2. P180 연습문제 3번

2_(1) 문제에서 제시한 요구사항들을 만족하는 클래스 다이어그램을 그리기.

2_(2) p.414~417 코드를 읽고 작성함

book.java

```java
package week6.ex1;

6 usages  new *
public class Book {
    2 usages
    private String name;
    2 usages
    private int publicshYear;
    2 usages
    private int price;

    2 usages  new *
    public Book(String name, int publicshYear, int price) {
        this.name = name;
        this.publicshYear = publicshYear;
        this.price = price;
    }

    1 usage  new *
    public String getName() { return this.name; }

    1 usage  new *
    public int getPublicshYear() { return this.publicshYear; }

    2 usages  new *
    public int getPrice() { return this.price; }
}
```

BookDiscountPricePolicy.java

```java
package week6.ex1;

1 usage  new *
public class BookDiscountPricePolicy implements PricePolicy{
    2 usages  new *
    @Override
    public int calcPrice(int price, int n) { return (int)(price * n * 0.95); }
}
```

2_(2) p.414~417 코드를 읽고 작성함

Client.java

```java
package week6.ex1;

no usages  new *
public class Client {
    no usages  new *
    public static void main(String[] args){
        Member member1 = new Member( name: "member1");
        Member member2 = new Member( name: "member2");

        Book book1 = new Book( name: "Book1",  publicshYear: 2005,  price: 1000);
        Book book2 = new Book( name: "Book2",  publicshYear: 2004,  price: 1200);
        Book book3 = new Book( name: "Book3",  publicshYear: 2001,  price: 4300);

        Rental rental1 = new Rental(member1, book2, new OrdinaryPricePolicy(),  n: 2);
        Rental rental2 = new Rental(member2, book1, new OrdinaryPricePolicy(),  n: 3);
        Rental rental3 = new Rental(member2, book3, new BookDiscountPricePolicy(),  n: 4);
        Rental rental4 = new Rental(member1, book2, new MemberDiscountPricePolicy(),  n: 3);

        System.out.println(rental1.getPrice());
        System.out.println(rental2.getPrice());
        System.out.println(rental3.getPrice());
        System.out.println(rental4.getPrice());
    }
}
```

Member.java

```java
package week6.ex1;

6 usages  new *
public class Member {
    2 usages
    private String Name;
    2 usages
    private int accPrice;

    2 usages  new *
    public Member(String name) {
        this.Name = name;
        accPrice = 0;
    }

    1 usage  new *
    public String getName() { return Name; }

    1 usage  new *
    public void addAccPrice(int accPrice) { accPrice += accPrice; }

    2 usages  new *
    public int getAccPrice() { return accPrice; }
}
```

2_(2) p.414~417 코드를 읽고 작성함

MemberPricePolicy.java

```java
package week6.ex1;

1 usage  new *
public class MemberDiscountPricePolicy implements PricePolicy{
    2 usages  new *
    @Override
    public int calcPrice(int price, int n) { return (int)(price * n * 0.90); }
}
```

OrdinaryPricePolicy.java

```java
package week6.ex1;

1 usage  new *
public class OrdinaryPricePolicy implements PricePolicy{

    2 usages  new *
    @Override
    public int calcPrice(int price, int n) { return price * n; }
}
```

2_(2) p.414~417 코드를 읽고 작성함

PricePolicy.java

```java
package week6.ex1;

6 usages  3 implementations  new *
interface PricePolicy {
    2 usages  3 implementations  new *
    public int calcPrice(int price, int n);
}
```

Rental.java

```java
package week6.ex1;

8 usages  new *
public class Rental {
    2 usages
    private Member member;
    2 usages
    private Book book;
    2 usages
    private PricePolicy pricePolicy;
    2 usages
    int n;

    4 usages  new *
    public Rental(Member member, Book book, PricePolicy pricePolicy, int n) {
        this.member = member;
        this.book = book;
        this.n = n;
        this.pricePolicy = pricePolicy;
        this.member.addAccPrice(pricePolicy.calcPrice(book.getPrice(), n));
    }

    4 usages  new *
    public int getPrice() { return pricePolicy.calcPrice(book.getPrice(), n); }
}
```

2_(3) 작성한 코드는 문제에서 제시한 요구사항을 제대로 반영하지 않고 있다. 어느 부분이 어떻게 변경되어야 하는지 찾아보고 가격정책 3가지를 정확하게 반영하도록 코드를 수정하시오

2_(3)에서 작성된 코드는 client 코드에서 직접 가격정책을 설정하여 사용하고 있다.
이것은 DIP 위배라고 볼 수 있다. 이를 해결하기 위해선 PricePolicy를 직접 입력받는 것이 아니라 Rental class에서 추론하여 선택할 수 있도록 해야 한다.

수정된 코드

```java
public class Rental {
    5 usages
    private Member member;
    4 usages
    private Book book;
    4 usages
    private PricePolicy pricePolicy;
    3 usages
    int n;

    2 usages  new *
    public Rental(Member member, Book book, int n) {
        this.member = member;
        this.book = book;
        this.n = n;
        if(this.member.getAccPrice() >= 10000)
            setStrategy(new MemberDiscountPricePolicy());
        else if ((2023 - this.book.getPublicshYear()) >= 10)
            setStrategy(new BookDiscountPricePolicy());
        else
            setStrategy(new OrdinaryPricePolicy());
        this.member.addAccPrice(pricePolicy.calcPrice(book.getPrice(), n));
    }

    1 usage  new *
    public int getPrice() { return pricePolicy.calcPrice(book.getPrice(), n); }

    3 usages  new *
    public void setStrategy(PricePolicy pricePolicy) { this.pricePolicy = pricePolicy; }

    1 usage  new *
    public PricePolicy getPricePolicy() { return this.pricePolicy; }
```

2_(4) Main이 아래와 같을 때 실행하고 결과를 제출하시오. 실행결과 샘플을 보고 rental클래스의 summary함수도 구현할 것

Rental.summary() :

```
    public void summary() {
        System.out.println(this.member.getName() + " rented " + this.n + this.book.getName() + " :: " + this.getPricePolicy() + " :: " + getPrice() + " ACC :: " + member.getAccPrice());
    }
}
```

실행 결과 :

```
"C:\Program Files\Java\jdk-18.0.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.3.3\lib\idea_rt.jar=51309:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.3.3\bin" -
Tom rented 2Operating System Conceptw :: week6.ex1.BookDiscountPricePolicy@377dca04 :: 47500 ACC :: 0
Jane rented 4Introduction to SoftWare Testing :: week6.ex1.OrdinaryPricePolicy@728938a9 :: 136000 ACC :: 0

Process finished with exit code 0
```

2_(5) SpecialPricePolicy 정책(3권이상 구매하는 경우, 50%할인을 해주는 정책)이 추가되도록 요구사항이 변경되었다. 반영하여 코드를 작성하고, (4)의 main을 실행시키고 결과를 보이시오.

SpecialPricePolicy :

```
package week6.ex1;

no usages  new *
public class SpecialPricePolicy implements PricePolicy{

    2 usages  new *
    @Override
    public int calcPrice(int price, int n) {
        return price * ((n+2)/2);
    }
}
```

Rental.constructor: :

```java
public Rental(Member member, Book book, int n) {
    this.member = member;
    this.book = book;
    this.n = n;
    if (n > 3){
        setStrategy(new SpecialPricePolicy());
    }
    else if(this.member.getAccPrice() >= 10000)
        setStrategy(new MemberDiscountPricePolicy());
    else if ((2023 - this.book.getPublicshYear()) >= 10)
        setStrategy(new BookDiscountPricePolicy());
    else
        setStrategy(new OrdinaryPricePolicy());
    this.member.addAccPrice(pricePolicy.calcPrice(book.getPrice(), n));
}
```

실행 결과 :

```
"C:\Program Files\Java\jdk-18.0.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.3.3\lib\idea_rt.jar=49718:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.3.3\bin" -Df
Tom rented 2Operating System Conceptw :: week6.ex1.BookDiscountPricePolicy@377dca04 :: 47500 ACC :: 0
Jane rented 4Introduction to SoftWare Testing :: week6.ex1.SpecialPricePolicy@728938a9 :: 102000 ACC :: 0

Process finished with exit code 0
```

3. 연습문제 4번 풀기 (p.419 표)

3_(1) 책 p.181~의 코드로 코딩하기 : Ball.java 잘 살피기. setter들과 그로 인해 결정된 것에 따른 draw와 move.

BallFrame.java

```
package week6.ex2;

import ...

9 usages   new *
public class BallFrame extends JFrame {
    no usages
    private static final long serialVersionUID = 1L;
    5 usages
    public static final int WIDTH = 400;
    5 usages
    public static final int HEIGHT = 400;
    3 usages
    private Field field;
    1 usage   new *
    public BallFrame(Ball[] balls) {
        super( title: "Balls");
        setDefaultCloseOperation(EXIT_ON_CLOSE);
        setLayout(new BorderLayout());

        field = new Field(balls);
        Thread th = new Thread(field);
        th.start();

        add(field, BorderLayout.CENTER);

        setSize(WIDTH, HEIGHT);
        setVisible(true);
    }
}
```

Field.java

```
package week6.ex2;

import ...

2 usages
public class Field extends JPanel implements Runnable {
    no usages
    private static final long serialVersionUID = 1L;
    5 usages
    private Ball[] balls;
    1 usage
    public Field(Ball[] balls) {
        this.balls = balls;
        setLayout(new FlowLayout());
    }

    public void paint(Graphics g) {
        g.clearRect( x: 0,  y: 0, BallFrame.WIDTH, BallFrame.HEIGHT);
        for (int i = 0; i < balls.length; i++) {
            g.setColor(balls[i].getColor());
            g.fillOval(balls[i].getX(), balls[i].getY(), Ball.SIZE, Ball.SIZE);
        }
    }

    public void run() {
        while (true) {
            repaint();
            try {
                Thread.sleep( millis: 5);
            } catch (InterruptedException e) {}
        }
    }
}
```

3_(1)

ball.java(1)

```java
package week6.ex2;

import java.awt.Color;

public class Ball extends Thread {
    public static final int SIZE = 20;
    public static final int INTERVAL = 10;
    private int x, y;
    private int xInterval, yInterval;
    private DirectionStrategy directionStrategy;
    private DrawStrategy drawStrategy;
    private Color color;

    public Ball(int x, int y) {
        this.x = x;
        this.y = y;
        this.xInterval = this.yInterval = 0;
    }

    public int getX() { return x; }

    public int getY() { return y; }

    public int getxInterval() { return xInterval; }

    public int getyInterval() { return yInterval; }

    public void setX(int x) { this.x = x; }

    public void setY(int y) { this.y = y; }

    public void setIntervals(int xInterval, int yInterval) {    //움직임 간격 설정
        this.xInterval = xInterval;
        this.yInterval = yInterval;
    }
```

ball.java(2)

```java
    public void setColor(Color color) { this.color = color; }

    public Color getColor() { return this.color; }

    public void setDirectionStrategy(DirectionStrategy directionStrategy) {
        this.directionStrategy = directionStrategy;
    }

    public void setDrawStrategy(DrawStrategy drawStrategy) {
        this.drawStrategy = drawStrategy;
    }

    public void draw() { drawStrategy.draw( ball: this); }

    public void move() { directionStrategy.move( ball: this); }

    public void run() {
        draw();
        move();
    }
```

3_(1)

client.java

```
package week6.ex2;

import java.util.Random;

public class Client {
    private static int INIT_LOCATION[] = {50, 100, 150};

    public static void main(String[] args) {
        Ball balls[] = new Ball[3];
        for (int i = 0; i < balls.length; i++) {
            balls[i] = new Ball(INIT_LOCATION[i], INIT_LOCATION[i]);
            balls[i].setDirectionStrategy(new HorizonalMoveStrategy());
            balls[i].setDrawStrategy(new RedDrawStrategy());

            balls[i].start();
        }

        new BallFrame(balls);
    }
}
```

HorizonalMovingStrategy.java

```
package week6.ex2;
public class HorizonalMoveStrategy extends DirectionStrategy{
    public void move(Ball ball){
        ball.setIntervals(Ball.INTERVAL, yInterval: 0);
        while (true) {
            ball.setX(ball.getX()+ball.getxInterval());
            if ((ball.getX() < 0 && ball.getxInterval() < 0)
                    || ball.getX() + Ball.SIZE > BallFrame.WIDTH - 15 && ball.getxInterval() > 0){

                ball.setIntervals(-ball.getxInterval(), yInterval: 0);
            }
            try {
                Thread.sleep( millis: 30);
            }catch (InterruptedException e){

            }
        }
    }
}
```

3_(1)

DirectionStrategy.java

```java
package week6.ex2;
abstract public class DirectionStrategy {
    abstract public void move(Ball ball);
}
```

DrawStrategy.java

```java
package week6.ex2;
abstract public class DrawStrategy {
    abstract public void draw(Ball ball);
}
```

RedDrawStrategy.java

```java
package week6.ex2;

import java.awt.Color;
public class RedDrawStrategy extends DrawStrategy {
    @Override
    public void draw(Ball ball) { ball.setColor(Color.red); }

}
```

BlueDrawStrategy.java

```java
package week6.ex2;

import java.awt.Color;

public class BlueDrawStrategy extends DrawStrategy {
    @Override
    public void draw(Ball ball) { ball.setColor(Color.blue); }

}
```

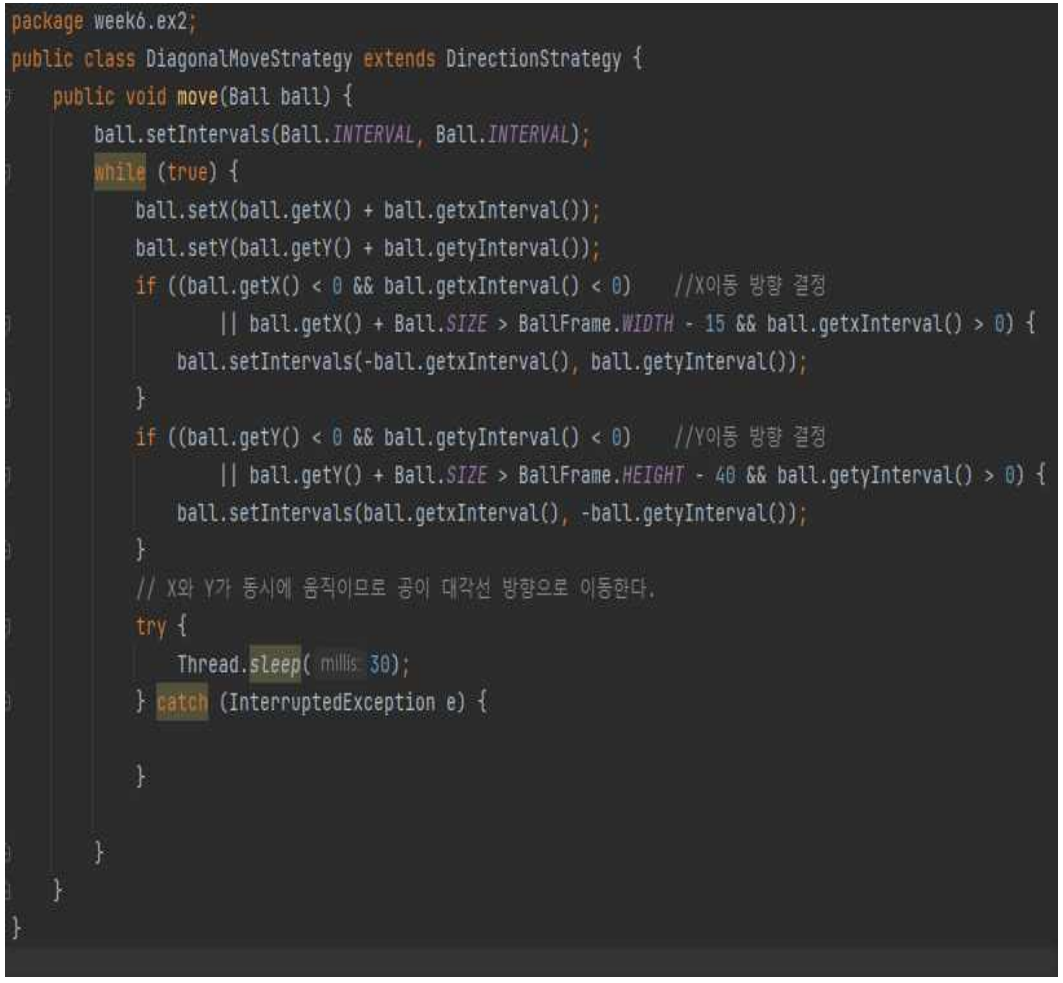
3_(2) 4-3, 4-4 : 빈 코드 채워서 완성시키기

VerticalMoveStrategy.java

```java
package week6.ex2;
public class VerticalMoveStrategy extends DirectionStrategy {
    public void move(Ball ball) {
        ball.setIntervals( xinterval: 0, Ball.INTERVAL);
        while (true) {
            ball.setY(ball.getY()+ball.getyInterval());
            if ((ball.getY() < 0 && ball.getyInterval() < 0)
                    || ball.getY() + Ball.SIZE > BallFrame.HEIGHT - 10 && ball.getyInterval() > 0){
                //만약 공의 Y좌표가 0보다 작고 interval이 0보다 작은 경우 또는
                // 공의 크기 + 좌표의 값이 화면의 높이 - 10보다 크고 공의 interval이 0보다 큰 경우
                ball.setIntervals( xinterval: 0, -ball.getyInterval()); // 공의 이동 방향을 180도 변경한다 (-interval)
            }
            try {
                Thread.sleep( millis: 30);
            }catch (InterruptedException e){

            }
        }
    }
}
```

DiagonalMoveStrategy.java



```java
package week6.ex2;
public class DiagonalMoveStrategy extends DirectionStrategy {
    public void move(Ball ball) {
        ball.setIntervals(Ball.INTERVAL, Ball.INTERVAL);
        while (true) {
            ball.setX(ball.getX() + ball.getxInterval());
            ball.setY(ball.getY() + ball.getyInterval());
            if ((ball.getX() < 0 && ball.getxInterval() < 0)    //X이동 방향 결정
                    || ball.getX() + Ball.SIZE > BallFrame.WIDTH - 15 && ball.getxInterval() > 0) {
                ball.setIntervals(-ball.getxInterval(), ball.getyInterval());
            }
            if ((ball.getY() < 0 && ball.getyInterval() < 0)    //Y이동 방향 결정
                    || ball.getY() + Ball.SIZE > BallFrame.HEIGHT - 40 && ball.getyInterval() > 0) {
                ball.setIntervals(ball.getxInterval(), -ball.getyInterval());
            }
            // X와 Y가 동시에 움직이므로 공이 대각선 방향으로 이동한다.
            try {
                Thread.sleep( millis: 30);
            } catch (InterruptedException e) {

            }
        }
    }
}
```

3_(3) Client.java를 수정하여 10개의 ball을 운영하기. Move와 draw 전략은 팀별로 정의.

```java
package week6.ex2;

import java.util.Random;

public class Client {
    private static int INIT_LOCATION[] = new int[10];

    public static void main(String[] args) {
        Random rd = new Random();
        for(int i=0;i<10;i++) {
            INIT_LOCATION[i] = rd.nextInt( bound: 300); //  공 10개의 시작 좌표는 랜덤 정수의 형태로 초기화 한다.
        }
        Ball balls[] = new Ball[10];
        for (int i = 0; i < balls.length; i++) {
            balls[i] = new Ball(INIT_LOCATION[i], INIT_LOCATION[i]);
            balls[i].setDirectionStrategy(new VerticalMoveStrategy());  // 모든 공이 수직 방향으로 이동
            balls[i].setDrawStrategy(new BlueDrawStrategy());           // 모든 공이 파란색
            balls[i].start();
        }

        new BallFrame(balls);
    }
}
```

3_(4) 각 팀에서 새로운 yellow로 칠하는 draw strategy를- 구현하여 추가하기.

YellowDrawStrategy.java

```java
package week6.ex2;

import java.awt.*;
public class YellowDrawStrategy extends DrawStrategy {
    @Override
    public void draw(Ball ball) { ball.setColor(Color.yellow); }

}
```

3_(5) Client.java를 수정하여 다른 시나리오 진행시키기 : 3개의 ball, vertical-blue, horizontal-red, diagonal-yellow,

10개의 ball의 생성 순서에 따라서 전략을 순회하면서 설정한다. 생성 순서의 식별은 client에서의 ball객체 생성 반복문의 Index를 사용한다.

ball.setDirectionAndDrawStrategy();

```
public void setDirectionAndDrawStrategy(int i) {

    if(i%3 == 1){
        this.drawStrategy = new RedDrawStrategy();
        this.directionStrategy = new DiagonalMoveStrategy();
    } else if (i%3 == 2) {
        this.drawStrategy = new BlueDrawStrategy();
        this.directionStrategy = new VerticalMoveStrategy();
    } else{
        this.drawStrategy = new YellowDrawStrategy();
        this.directionStrategy = new HorizonalMoveStrategy();
    }
}
```

3_(6) 각 팀에서 새로운 moving strategy를 제안하고 구현하여 추가하기.

점차 느려지다가 일정 속도에 도달하면 다시 초기 속도로 움직이는 Move Strategy

3_(5)와는 다르게 랜덤생성된 시작 좌표를 인수로 받아 params/4를 하여 MoveStrategy와 DrawStrategy를 선택한다.

ball.setDirectionAndDrawStrategy();(2)

실행결과화면

```
public void setDirectionAndDrawStrategy(int i){
    if(i%3 == 1){
        setDrawStrategy(new YellowDrawStrategy());
    } else if (i%3 == 2) {
        setDrawStrategy(new BlueDrawStrategy());
        setDirectionStrategy(new VerticalMoveStrategy());
    } else{
        setDrawStrategy(new RedDrawStrategy());
        setDirectionStrategy(new HorizonalMoveStrategy());
    }
    if(i%4 == 1){
        setDirectionStrategy(new DiagonalMoveStrategy());
    } else if (i%4 == 2) {
        setDirectionStrategy(new VerticalMoveStrategy());
    } else if (i%4 == 3){
        setDirectionStrategy(new HorizonalMoveStrategy());
    } else {
        setDirectionStrategy(new NewMoveStrategy());
    }
}
```

녹음 2023-04-14 141931.mp4