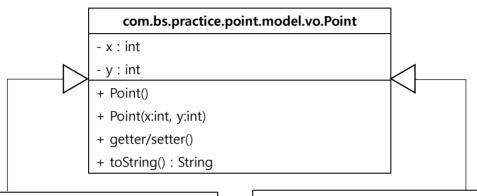
[문제1] 다음과 같은 조건을 만족하는 프로그램을 작성 하시오

1. 구현 클래스 다이어그램 (Class Diagram)

com.bs.practice.point.run.Run

+ main(args:String[]): void



com.bs.practice.point.model.vo.Circle

- radius : int
- + Circle()
- + Circle(x:int, y:int, radius:int)
- + getter/setter()
- + toString(): String

com.bs.practice.point.model.vo.Rectangle

- width: int
- height: int
- + Rectangle()
- + Rectangle(x:int, y:int, width:int, height:int)
- + getter/setter()
- + toString(): String

com.bs.practice.point.view.PointMenu

- -sc:Scanner = new Scanner(System.in)
- -cc:CircleController = new CircleController
- -rc:RectangleController = new

RectangleController

- + mainMenu(): void
- + circleMenu(): void
- + rectangleMenu():void
- + calcCircum():void
- + calcCircleArea():void
- + calcPerimeter():void
- + calcRectArea():void

com.bs.practice.point.controller. RectangleController

- r:Rectangle = new Rectangle()
- + calcArea(x:int, y:int, height:int, width:int):String
- +calcPerimeter(x:int, y:int, height:int,
- width:int):String

com.bs.practice.point.controller. CircleController

- c:Circle = new Circle()
- + calcArea(x:int, y:int, radius:int):String
- +calcCircum(x:int, y:int, radius:int) :String

* 각 클래스의 setter/getter 메소드는 직접 구현한다.

2. 구현 클래스 설명

Package명	Class명	Method	설명
com.bs.practice.	Point	+Point()	기본 생성자
point.model.vo		+Point(x:int, y:int)	매개변수 생성자
		+toString() : String	필드에 담긴 데이터를 반환하는 메소드
	Circle	+Circle()	기본 생성자
		+Circle(x:int, y:int, radius:int)	매개변수 생성자
		+toString() : String	필드에 담긴 데이터를 반환하는 메소드
	Rectangle	+Rectangle()	기본 생성자
		+Rectangle(x:int, y:int,	매개변수 생성자
		width:int, height:int)	
		+toString() : String	필드에 담긴 데이터를 반환하는 메소드
com.bs.practice.	CircleCon	+calcArea(x:int, y:int,	받은 매개변수를 이용하여 필드들을 초기
point.controller	troller	radius:int):String	화하고 초기화한 정보와 면적 반환
			// 면적 : PI * 반지름 * 반지름
		+calcCircum(x:int, y:int,	받은 매개변수를 이용하여 필드들을 초기
		radius:int):String	화하고 초기화한 정보와 둘레 반환
			// 둘레 : PI * 반지름 * 2
	Rectangle	+calcArea(x:int, y:int,	받은 매개변수를 이용하여 필드들을 초기
	Controller	height:int, width:int):String	화하고 초기화한 정보와 면적 반환
			// 면적 : 너비 * 높이
		+calcPerimeter(x:int, y:int,	받은 매개변수를 이용하여 필드들을 초기
		height:int, width:int):String	화하고 초기화한 정보와 둘레 반환
			// 둘레 : 2 * (너비 + 높이)
Com.bs.practice.	PointMen	+mainMenu():void	메인메뉴 출력, 잘못 입력했을 시 다시 받
point.view	u		도록 반복
		+circleMenu():void	원 메뉴 출력, 잘못 입력했을 시 메인메뉴
			로 돌아감
		+rectangleMenu():void	사각형 메뉴 출력, 잘못 입력했을 시 메인
			메뉴로 돌아감
		+calcCircum():void	X, y 좌표와 반지름을 받아 원의 정보와
			원 둘레 출력
		+calcCircleArea():void	X, y 좌표와 반지름을 받아 원의 정보와
			원 넓이 출력
		+calcPerimeter():void	X, y 좌표와 높이, 너비를 받아 사각형의
			정보와 사각형 둘레 출력
		+calcRectArea():void	X, y 좌표와 높이, 너비를 받아 사각형의
			정보와 사각형 넓이 출력

com.bs.practice.	Run	+main(args:String[]): void	mainMenu() 실행
point.run			

3. class 구조

```
public class Run{
    public static void main(String[] args) {
        // PointMenu의 mainMenu() 실행
    }
}
```

```
public class PointMenu{
      // 필드
      public void mainMenu() {
             ===== 메뉴 =====
             1. 원
                          → circleMenu()
             2. 사각형
                          → rectangleMenu()
             9. 끝내기
              메뉴 번호 :
      public void circleMenu(){
             ===== 원 메뉴 =====
             1. 원 둘레
                          → calcCircum()
             2. 원 넓이
                          → calcCircleArea()
             9. 메인으로
             메뉴 번호 :
      }
      public void retangleMenu() {
             ===== 메뉴 =====
             1. 사각형 둘레
                                  → calcPerimeter()
             2. 사각형 넓이
                                  → calcRectArea()
             3. 메인으로
             메뉴 번호 :
      }
```

```
public void calcCircum(){
              x 좌표 :
              y 좌표 :
              반지름 :
              // 데이터를 CircleController(cc)의 calcCircum()의 매개변수로 보내 반환 값 출력
       }
       public void calcCircleArea(){
              x 좌표 :
              y 좌표 :
              반지름 :
              // 데이터를 CircleController(cc)의 calcArea()의 매개변수로 보내 반환 값 출력
       }
       public void calcPerimeter(){
              x 좌표 :
              y 좌표 :
              높이 :
              너비:
              // 데이터를 RectangleController(rc) calcPerimeter()의 매개변수로 보내 반환값 출력
       public void calcPerimeter(){
              x 좌표 :
              y 좌표 :
              높이 :
              너비:
              // 데이터를 RectangleController(rc) calcArea()의 매개변수로 보내 반환값 출력
       }
}
```

5. 실행 결과 화면

```
===== 메뉴 =====
1. 원
2. 사각형
9. 끝내기
메뉴 번호 : 1
===== 원 메뉴 =====
1. 원 둘레
2. 원 넓이
9. 메인으로
메뉴 번호 : 1
x 좌표 : 2
y 좌표 : 4
반지름 : 1
둘레 : 2, 4, 1 / 6.283185307179586
===== 메뉴 =====
1. 원
2. 사각형
9. 끝내기
메뉴 번호 : 1
===== 원 메뉴 =====
1. 원 둘레
2. 원 넓이
9. 메인으로
메뉴 번호 : 2
x 좌표 : 2
y 좌표 : 4
반지름 : 1
면적: 2, 4, 1 / 3.141592653589793
===== 메뉴 =====
1. 원
2. 사각형
9. 끝내기
메뉴 번호 : 2
===== 사각형 메뉴 =====
1. 사각형 둘레
2. 사각형 넓이
9. 메인으로
메뉴 번호 : 1
x 좌표 : 2
y 좌표 : 4
높이 : 1
너비 : 3
둘레 : 2, 4 1 3 / 8
===== 메뉴 =====
1. 원
2. 사각형
9. 끝내기
메뉴 번호 : 2
```

```
===== 사각형 메뉴 =====
1. 사각형 둘레
2. 사각형 넓이
9. 메인으로
메뉴 번호 : 2
x 좌표 : 2
y 좌표 : 4
높이 : 1
너비 : 3
면적 : 2, 4 1 3 / 3
===== 메뉴 =====
1. 원
2. 사각형
9. 끝내기
메뉴 번호 : 3
종료합니다.
```