❑ 응용 프로그래밍

(1) **[생성형 AI 활용]** 인터페이스 멤버로 가능한 모든 형태와 구체화하는 방법을 예와 함께 설명하세요. 인터페이스를 구현할 때 주의할 점은 무엇인가요?

생성형 AI 플랫폼	
질문	
답변	
실행결과	

❑ 응용 프로그래밍

(2) 제시된 실행 결과를 보고 DataAccessObject 인터페이스와 OracleDB와 MySqlDB 구현 클래스를 작성하시오.

```
Oracle DB에서 검색
Oracle DB에 삽입
Oracle DB를 수정
Oracle DB에서 삭제
MySql DB에서 검색
MySql DB에 삽입
MySql DB를 수정
MySql DB에서 삭제
```

```
public class InterfaceTest {
	public static void dbWork(DataAccessObject dao) {
		dao.select();
		dao.insert();
		dao.update();
		dao.delete();
	}

	public static void main(String[] args) {
		dbWork(new OracleDB("Oracle DB"));
		dbWork(new MySqlDB("MySql DB"));
	}
}
```

[프로그램 소스]

```
interface DataAccessObject{
	void select();
	void insert();
	void update();
	void delete();
}

class OracleDB implements DataAccessObject{
	private String name;
```

```
	public OracleDB(String name){this.name = name;}
	@Override
	public void select() {
		System.out.println(name +" 에서 검색");
	}
	@Override
	public void insert() {
		System.out.println(name +" 에 삽입");
	}
	@Override
	public void update() {
		System.out.println(name +"를 수정");
	}
	@Override
	public void delete() {
		System.out.println(name +"에서 삭제");
	}
}

class MySqlDB implements DataAccessObject{
	private String name;
	public MySqlDB(String name){this.name = name;}

	@Override
	public void select() {
		System.out.println(name + "에서 검색");
	}
	@Override
	public void insert() {
		System.out.println(name +"에 삽입");
	}
	@Override
	public void update() {
		System.out.println(name + "를 수정");
	}
	@Override
	public void delete() {
		System.out.println(name +"에서 삭제");
	}
}
public class Answer1 {
	public static void dbWork(DataAccessObject dao) {
		dao.select();
		dao.insert();
```

```
                dao.update();
                dao.delete();
        }
        public static void main(String[] args) {
                dbWork(new OracleDB("Oracle DB"));
                dbWork(new MySqlDB("MySql DB"));
        }
}
```

[실행 결과]

(3) 조건대로 프로그램을 작성하고 테스트 하시오

>>SmartDevice 인터페이스를 정의하여 다음과 같은 메소드를 선언한다

void turnOn() -> 장치 켜기, void turnOff()->장치 끄기, Boolean isToggle() -> 상태 변경

>>SmartLight, SmartThermostat, SmartTV 클래스들을 구현하여 각 장치의 turnOn()과 turnOff(), isToggle 메서드를 정의한다. 각 클래스는 상태를 나타내는 필드를 갖는다

>>SmartHomeController 클래스를 작성하여 여러 스마트 장치들을 한 번에 제어할 수 있게 한다. 프로그램을 작성하고 테스트 하시오.

```
        public static void main(String[] args) {
                SmartHomeController controller = new SmartHomeController();

                SmartDevice light = new SmartLight();
                SmartDevice thermostat = new SmartThermostat();
                SmartDevice tv = new SmartTV();

                //장치 제어
                controller.controlDevice(light);
                controller.controlDevice(light);
                controller.controlDevice(thermostat);
                controller.controlDevice(thermostat);
                controller.controlDevice(tv);
                controller.controlDevice(tv);
        }
```

```
조명이 켜졌습니다.
조명이 꺼졌습니다.
온도 조절기가 켜졌습니다.
온도 조절기가 꺼졌습니다.
TV가 켜졌습니다.
TV가 꺼졌습니다.
```

[프로그램 소스]

```
//SmartDevice 인터페이스 정의
interface SmartDevice {
        void turnOn(); // 장치 켜기
        void turnOff(); // 장치 끄기
        boolean isToggle();  //상태 변경
}
```

```
//스마트 조명 클래스
class SmartLight implements SmartDevice {
	private boolean state;

	public boolean isToggle() {
		state = !state;
		return state;
	}

	@Override
	public void turnOn() {
		System.out.println("조명이 켜졌습니다.");
	}

	@Override
	public void turnOff() {
		System.out.println("조명이 꺼졌습니다.");
	}
}

//스마트 온도 조절기 클래스
class SmartThermostat implements SmartDevice {
	private boolean state;

	public boolean isToggle() {
		state = !state;
		return state;
	}

	@Override
	public void turnOn() {
		System.out.println("온도 조절기가 켜졌습니다.");
	}

	@Override
	public void turnOff() {
		System.out.println("온도 조절기가 꺼졌습니다.");
	}
}

//스마트 TV 클래스
class SmartTV implements SmartDevice {
	private boolean state;
```

```
	public boolean isToggle() {
		state = !state;
		return state;
	}

	@Override
	public void turnOn() {
		System.out.println("TV 가 켜졌습니다.");
	}

	@Override
	public void turnOff() {
		System.out.println("TV 가 꺼졌습니다.");
	}
}

//스마트 홈 시스템 관리 클래스
class SmartHomeController {
	public void controlDevice(SmartDevice device) {
		if (device.isToggle()) {
			device.turnOn();
		} else {
			device.turnOff();
		}
	}
}

public class Main {
	public static void main(String[] args) {
		SmartHomeController controller = new SmartHomeController();

		SmartDevice light = new SmartLight();
		SmartDevice thermostat = new SmartThermostat();
		SmartDevice tv = new SmartTV();

		//장치 제어
		controller.controlDevice(light);
		controller.controlDevice(light);
		controller.controlDevice(thermostat);
		controller.controlDevice(thermostat);
		controller.controlDevice(tv);
		controller.controlDevice(tv);
	}
}
```

[실행 결과]

(4) 다음과 같은 인터페이스를 구현하는 클래스 TV와 SmartPhone를 작성하고 main()에서 테스트 하시오. 제시된 결과를 참조하시오

```
TV를 켭니다.
 ----  TV 볼륨을 10으로 합니다 ----
현재 TV 볼륨: 10
 ----  TV 볼륨을 무음으로 합니다 ----
무음 처리합니다.
SmartPhone을 켭니다.
 ----  SmartPhone 볼륨을 50으로 합니다 ----
현재 SmartPhone 볼륨: 50
건전지를 교환합니다.
```

```
public interface RemoteControl {
	//상수 필드 선언
	public final static int MAX_VOLUME = 100;
	public int MIN_VOLUME=0;

	//추상 메소드 선언
	public void turnOn();
	public void turnOff();
	public void setVolume(int volume);

	default void setMute(boolean mute) { //디폴트 메소드
		if(mute) {
			System.out.println("무음 처리합니다.");
		} else {
			System.out.println("무음 해제합니다.");
		}
	}
	static void changeBattery() {
		System.out.println("건전지를 교환합니다.");
	}
}
```

[프로그램 소스]

```
class TV implements RemoteControl {
	private int volume;
	public void turnOn() {	//turnOn() 추상 메소드의 실체 메소드
		System.out.println("TV를 켭니다.");
	}
	public void turnOff() {	//turnOff() 추상 메소드의 실체 메소드
		System.out.println("TV를 끕니다.");
	}
	public void setVolume(int volume) {	//setVolume() 추상 메소드의 실체 메소드
```

```
                if(volume>RemoteControl.MAX_VOLUME) {
                        this.volume = RemoteControl.MAX_VOLUME;
                } else if(volume<RemoteControl.MIN_VOLUME) {
                        this.volume = RemoteControl.MIN_VOLUME;
                } else {
                        this.volume = volume;
                }
                System.out.println("현재 TV 볼륨: " + volume);
        }
}

class SmartPhone implements RemoteControl {
        private int volume;
        public void turnOn() {      //turnOn() 추상 메소드의 실체 메소드
                System.out.println("SmartPhone을 켭니다.");
        }
        public void turnOff() {     //turnOff() 추상 메소드의 실체 메소드
                System.out.println("SmartPhone을 끕니다.");
        }
        public void setVolume(int volume) {   //setVolume() 추상 메소드의 실체 메소드
                if(volume>RemoteControl.MAX_VOLUME) {
                        this.volume = RemoteControl.MAX_VOLUME;
                } else if(volume<RemoteControl.MIN_VOLUME) {
                        this.volume = RemoteControl.MIN_VOLUME;
                } else {
                        this.volume = volume;
                }
                System.out.println("현재 SmartPhone 볼륨: " + volume);
        }
}

public class Answer2{
        public static void main(String[] args) {
                RemoteControl rc;
                rc = new TV();
                rc.turnOn();
                System.out.println(" ----   TV 볼륨을 10으로 합니다 ----");
                rc.setVolume(10);
                System.out.println(" ----   TV 볼륨을 무음으로 합니다 ----");
                rc.setMute(true);                       //default 메소드 호출

                rc = new SmartPhone();
                rc.turnOn();
                System.out.println(" ----   SmartPhone 볼륨을 50으로 합니다 ----");
```

```
                rc.setVolume(50);
                RemoteControl.changeBattery();  //정적 메소드 호출
        }
}
```

[실행 결과]

(5) 다음과 같은 조건을 만족하는 프로그램을 작성하시오

(a) 인터페이스 IGraphics를 작성한다.
- double perimeter()과 void draw()가 선언, 매개변수는 없음

(b) IGraphics을 구현하는 Rectangle 클래스를 작성한다.
- Rectangle 클래스는 2개의 private 실수 필드인 length와 width을 가진다.
- 두개의 필드를 매개값으로 초기화 하는 생성자
- draw()에서는 "도형 Rectangle을 그립니다."를 출력한다
- perimeter()에서는 사각형 둘레를 리턴 한다.

(c) IGraphics을 구현하는 Circle클래스를 작성한다
- Rectangle 클래스는 1개의 private 실수 필드인 radius을 가진다.
- 한 개의 필드를 매개값으로 초기화 하는 생성자
- draw()에서는 "도형 Circle을 그립니다."를 출력한다.
- perimeter()에서는 원 둘레를 리턴 한다.

ShapeTest 클래스를 작성한다.
- 메인 메소드에서 IGraphics 타입의 배열 arrayOfShapes를 3개생성하고, Rectangle, Circle 객체로 초기화
- 작성된 메소드를 호출하여 제시된 결과처럼 출력한다.

```
---------- Shape Draw---------
Rectangle Draw
Rectangle [length=12.3, width=3.4]
둘레 : 31.40

Rectangle Draw
Rectangle [length=20.3, width=5.6]
둘레 : 51.80

Circle Draw
Circle [radius=12.3]
둘레 : 77.24
```

[프로그램 소스]

```
interface IGraphics {
        void draw();
        double perimeter();
}

class Rectangle implements IGraphics {
        private double length, width;
```

```
	public Rectangle(double l, double w) {
		length = l;
		width = w;
	}

	public void draw() {
		System.out.println("Rectangle Draw");
	}

	public double perimeter() {
		return (length+width)*2;

	}

	@Override
	public String toString() {
		return "Rectangle [length=" + length + ", width=" + width + "]";
	}
}

class Circle implements IGraphics {
	private double radius;

	public Circle(double radius) {
		this.radius = radius;
	}
	private static final double PI=3.14;
	public void draw() {
		System.out.println("Circle Draw");
	}
	public double perimeter() {
		return 2*PI*radius;
	}
	@Override
	public String toString() {
		return "Circle [radius=" + radius + "]";
	}
}

public class ShapeTest {
	public static void main(String arg[]) {
		IGraphics [] arrayOfShapes = new IGraphics [3];

		arrayOfShapes[0] = new Rectangle(12.3, 3.4);
```

```
                arrayOfShapes[1] = new Rectangle(20.3, 5.6);
                arrayOfShapes[2] = new Circle(12.3);

                System.out.println("---------- Shape Draw---------");

                for (IDrawable id : arrayOfShapes) {
                        id.draw();
                    System.out.printf("%s₩n둘레 : %.2f₩n₩n", id, id.perimeter());
                }
                /*
                System.out.println();
                for (int i = 0; i < arrayOfShapes.length; i++)
                        arrayOfShapes[i].draw();
                        */
        }
}
```

[실행 결과]

(6) 아래의 설명에 따라 인터페이스와 클래스들을 정의하고 프로그램을 테스트하시오.

- IComparable 인터페이스는 int compareTo(Object other) 형태의 추상 메소드를 가지며 현재 객체가 other 객체보다 키가 크면 1, 같으면 0, 작으면 –1을 반환한다.

- Person 클래스는 이름(name), 키(height) 필드와 객체의 정보를 출력하는 toString()을 가진다. Person 클래스의 생성자에서는 전달된 값을 이름과 키 필드에 저장한다. Person 클래스는 IComparable 인터페이스를 구현한다.

- class PersonTest는 main()과 getMaximum()으로 구성 된다.

- main()에서는 Person 타입의 배열을 선언하여 세 사람의 이름과 키를 저장하고 getMaximum()을 호출하여 가장 키가 큰 사람의 정보를 출력한다.

- getMaximum()에서는 이 IComparable 인터페이스를 이용하여서 가장 키가 큰 사람의 객체를 반환한다.

- 프로그램을 수행하면 아래의 그림과 같은 결과를 보인다. 단, 제시된 main()메소드는 그대로 사용합니다

```
---------- Person List ----------
Person [name=Benny, height=180.0]
Person [name=Daniel, height=178.0]
Person [name=joon, height=188.0]
가장 키 큰 : Person [name=joon, height=188.0]
```

```
  public static void main(String[] args) {
            Person[] per = { new Person("Benny", 180), new Person("Daniel", 178), new Person("joon", 188)};

            System.out.println("---------- Person List ----------");
```

```
        for (Person p : per)
            System.out.println(p);

        System.out.println("가장 키 큰 : " + getMaximum(per));
    }
```

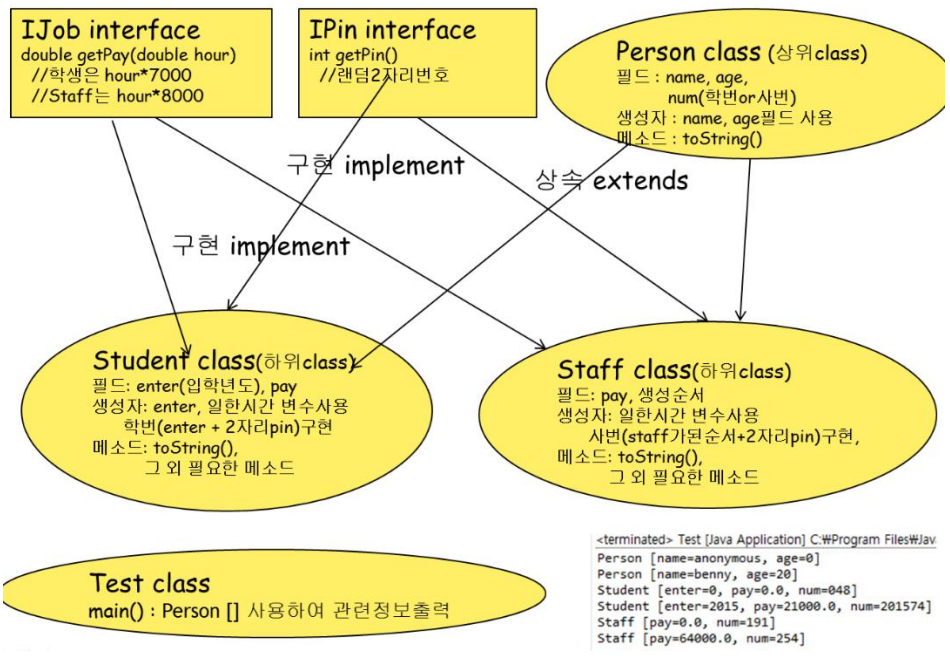
[프로그램 소스]

```
interface IComparable {
    // 이 객체가 다른 객체보다 크면 1, 같으면 0, 작으면 -1을 반환한다.
    public int compareTo(Object other);
}
class Person implements IComparable {
    private String name;
    private double height;

    public Person() {
        this("anonymous", 0);
    };

    public Person(String n, double h) {
        this.name = n;
        this.height = h;
    }

    public int compareTo(Object other) {
        Person p=null;
        if(other instanceof Person)
            p=(Person)other;

        if (this.height > ((Person) other).height)
            return 1;
        else if (this.height == ((Person) other).height)
            return 0;
        else
            return -1;
    }

    public String toString() {
        return "Person [name=" + name + ", height=" + height + "]";
    }
}

public class PersonTest {
```

```
	public static Person getMaximum(Person[] arr) {
		Person max = arr[0];

		for (int i = 1; i < arr.length; i++) {
			if (max.compareTo(arr[i]) == -1)
				max = arr[i];
		}
		return max; // 가장 키가 큰 사람의 정보를 출력한다
	}

	public static void main(String[] args) {
	    Person[] per = { new Person("Benny", 180), new Person("Daniel", 178), new Person("joon", 188)};

	    System.out.println("------------ Person List ------------");

		for (Person p : per)
			System.out.println(p);

		System.out.println("가장 키 큰 : " + getMaximum(per));
	}
}
```

[실행 결과]

(7) 다음 그림과 같은 프로그램을 완성하시오. 단, 제시된 main()메소드는 그대로 사용합니다.




```
	public static void main(String[] args) {
		Person [] per = new Person[6];
```

```
		per[0] = new Person();
		per[1] = new Person("benny", 20);
		per[2] = new Student();
		per[3] = new Student(2015, 3);
		per[4] = new Staff();
		per[5] = new Staff(8);

		for(Person obj : per) {
			System.out.println(obj);
		}
	}
```

[프로그램 소스]

```
interface IPin {
	int getPin();
}

interface IJob {
	double getPay(double hour);
}

//Person.java
class Person {
	private String name;
	private int age;
	protected String num;

	Person() {
		this("anonymous", 0);
	}

	Person(String n, int a) {
		this.name = n;
		this.age = a;
	}

	@Override
	public String toString() {
		return "Person [name=" + name + ", age=" + age + "]";
	}
}

class Staff extends Person implements IJob, IPin {
```

```
	private static int order;
	private double pay;

	Staff() {
		this(0);
	}

	Staff(double h) {
		order++;
		this.pay = getPay(h);
		super.num = order+"" + getPin(); //super.num =
Integer.toString(order)+Integer.toString(getPin());
	}

	public double getPay(double h) {
		return h*8000;
	}
	public int getPin() {
		return (int)(Math.random()*90) + 10;
	}
	@Override
	public String toString() {
		return "Staff [pay=" + pay + ", num=" + super.num + "]";
	}
}

//Student.java
class Student extends Person implements IJob, IPin {
	private int enter;
	private double pay;

	Student() {
		this(0, 0);
	};
	Student(int e, double h) {
		this.enter = e;
		this.pay = getPay(h);
		super.num = this.enter +"" +getPin(); //Integer.toString(this.enter) + Integer.toString(getPin())
	};
	public double getPay(double h) {
		return h*7000;
	}

	public int getPin() {
```

```
                return (int)(Math.random()*90) + 10;
        }

        @Override
        public String toString() {
         return "Student [enter=" + enter + ", pay=" + pay + ", num=" + super.num + "]";
        }
}

//Test.java
public class Test {
        public static void main(String[] args) {
                Person [] per = new Person[6];

                per[0] = new Person();
                per[1] = new Person("benny", 20);
                per[2] = new Student();
                per[3] = new Student(2015, 3);
                per[4] = new Staff();
                per[5] = new Staff(8);

                for(Person obj : per) {
                        System.out.println(obj);
                }
        }
}
```

[실행 결과]