[lab04 보고서]

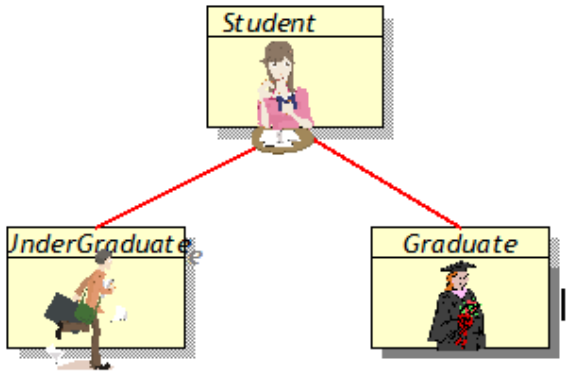
디지털미디어학과

2019111677

김지연

※ 아래 각 문제는 실습 과제입니다. 따라서 [분석-소스-결과]를 제출하세요.

1. 다음 그림에 해당하는 클래스를 정의하세요.



- 학생 클래스는 이름, 학번, 학과, 학년, 이수 학점 수라는 5개의 필드를 가진다.

- 학부생 클래스는 동아리명 필드를 추가적으로 갖고,

- 대학원생 클래스는 조교 유형과 장학금 비율이라는 2개의 필드를 추가로 갖는다.(조교 유형에는 "교육 조교"와 "연구 조교"가 있으며 장학금 비율은 0과 1사이의 값(예를 들어, 장학금 비율이 50%인 경우 이 값은 0.5임)이다.)

- 각 클래스는 적절한 생성자, 접근자, 설정자를 갖는다. (각자의 판단으로 적절하게 구성.)

위와 같은 클래스들의 객체를 각각 만들고 각 객체의 모든 정보(속성)를 출력하는 테스트 클래스도 정의하세요.

1-1. 분석

상속을 이용하여 부모 클래스인 학생클래스와 자식 클래스인 학부생과 졸업생 클래스를 만들어 속성을 출력하는 프로그램

우선 학생클래스에 이름, 학번, 학과, 학년, 이수학점이라는 5개의 필드를 만들고 각각의 접근자와 설정자를 구성한다. 5개의 속성을 가지는 명시적 생성자를 만든다. toString()메소드를 이용하여 공통적으로 반환하는 문자열(이름, 학번, 학과, 학년, 이수학점)을 만들어준다.

다음으로 학생 클래스를 상속받는 학부생 클래스를 만들어 준다. 학부생 클래스는 동아리명 필드를 추가적으로 갖고 이에 대한 설정자와 접근자를 설정해준다. 동아리명을 포함한 생성자를 만들어주고, 동아리명이라는 문자열을 포함한 toString()메소드를 만들어준다. 공통적인 속성은 부모 클래스에서 상속받아 사용 가능하다. 이때 toString()메소드는 오버라이딩 하여 사용한다. 오버라이딩 하지 않으면 원하지 않는 값을 반환하기 때문이다.

졸업생 클래스를 만들어준다. 졸업생 클래스는 조교유형과 장학금비율 필드를 추가적으로 갖는다. 조교유형은 논리형 연산자인 boolean연산자로 만들어준다. 참이라면 교육조교로 반환하고

false라면 연구조교를 반환한다. 조교유형과 장학금비율을 포함한 생성자와 toString()메소드를 만들어준다.

실행파일(StudentTest)에는 학부생과 졸업생 클래스의 객체를 생성하고 각 속성에 맞는 값을 입력한 후 출력해준다.

1-2. 소스코드

[클래스파일]

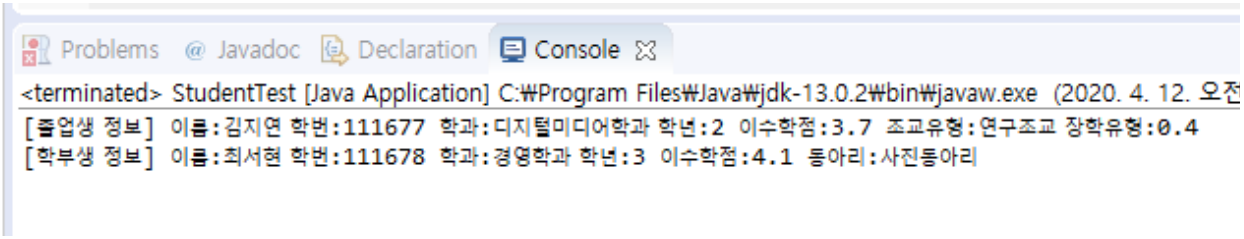
```
class Student {
	private String name;  //이름
	private int student_num;  //학번
	private String major;  //학과
	private int grade;  //학년
	private double score;  //학점
	//접근자, 설정자
	public String getName() {return name;}
	public void setName(String name) {this.name = name;}
	public int getStudent_num() {return student_num;}
	public void setStudent_num(int student_num) {this.student_num =
student_num;}
	public String getMajor() {return major;}
	public void setMajor(String major) {this.major = major;}
	public int getGrade() {return grade;}
	public void setGrade(int grade){this.grade = grade;}
	public double getScore() {return score;}
	public void setScore(double score) {this.score = score;}
	//명시적생성자
	public Student(String name, int student_num, String major, int grade,
double score) {
		setName(name);
		setStudent_num(student_num);
		setMajor(major);
		setGrade(grade);
		setScore(score);
	}
	@Override
	public String toString() {
		return "이름:"+getName()+" 학번:"+getStudent_num()+"
학과:"+getMajor()+" 학년:"+getGrade()+" 이수학점:"+getScore();
	}
}

//학생부 클래스
class UnderGraduate extends Student {

	private String group;
```

```
	public String getGroup() {return group;}
	public void  setGroup(String group) {this.group = group;}

	public UnderGraduate(String name, int student_num, String major, int
grade, double score, String group) {
		super(name, student_num, major, grade, score);
		setGroup(group);
	}
	@Override
	public String toString() {
		return "[학부생 정보] "+super.toString()+" 동아리:"+getGroup();
	}
}

//졸업생 클래스
class Graduate extends Student{

	private boolean assistant;
	private double scholarship;
	public String getAssistant() {
		if (assistant==true)
			return "교육조교";
		else
			return "연구조교";
	}

	public void setAssistant(boolean assistant) {this.assistant = assistant;}
	public double getScholarship() {return scholarship;}
	public void setScholarship(double scholarship) {this.scholarship =
scholarship;}

	public Graduate(String name, int student_num,String major, int grade,
double score, boolean assistant, double scholarship) {
		super(name, student_num, major, grade, score);
		setAssistant(assistant);
		setScholarship(scholarship);
	}
	@Override
	public String toString() {
		return "[졸업생 정보] "+super.toString()+"
조교유형:"+getAssistant()+" 장학유형:"+getScholarship();
	}
}
```

[실행파일]

```
public class StudentTest {
	public static void main(String[] args) {
			Graduate g = new Graduate("김지연", 111677,
"디지털미디어학과", 2,3.7, false, 0.4);
			UnderGraduate u = new UnderGraduate("최서현", 111678,
"경영학과", 3, 4.1, "사진동아리");
			System.out.println(g);
			System.out.println(u);
		}
	}
```

1-3. 결과화면

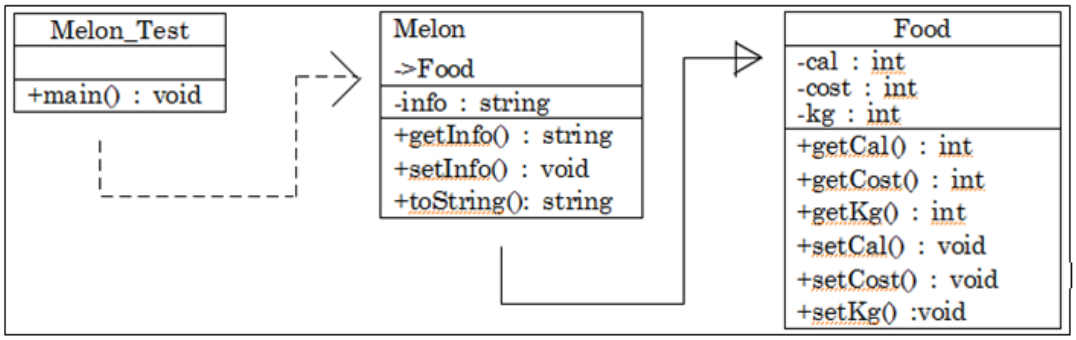


2. 일반적인 음식을 나타내는 Food 클래스를 상속받아서 멜론을 나타내는 Melon 클래스를 정의하세요.

- Food 클래스는 칼로리, 가격, 중량이라는 3개의 필드를 갖는다.

- Melon 클래스는 경작지 정보(예를 들어 원산지는 "아프리카"라는 문자열 정보) 필드를 추가로 갖는다.

- 아래 UML 다이어그램을 참고해서 각 클래스의 멤버를 구성한다.



위와 같은 클래스들의 객체를 각각 만들고 각 객체의 모든 정보(속성)를 출력하는 Melon_Test 클래스를 정의하세요.

2-1. 분석

Food 부모클래스와 Melon 자식클래스를 프로그래밍하고 각 객체의 정보를 출력하는 MelonTest를 정의한다.

부모 클래스는 칼로리, 가격, 중량이라는 필드를 갖는다. 각각의 속성에 설정자와 접근자를 설정해준다. 칼로리, 가격, 중량을 가지는 생성자를 만들어 준다.

Food클래스를 상속받는 Melon클래스를 생성한다. Melon클래스는 원산지 정보를 추가로 갖는다. 원산지에 대한 설정자와 접근자를 설정한다. 가격, 칼로리, 중량은 상속받고 원산지를 포함한 생성자와 toString()메소드를 만들어준다.

MelonTest클래스에 각 객체를 생성하고 속성의 정보를 담아 출력한다.

2-2. 소스코드

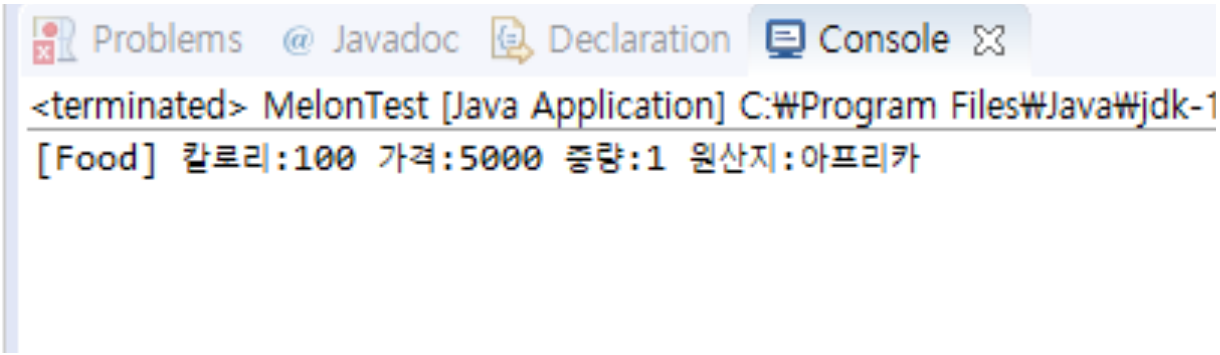
[클래스코드]

```
class Food {
	private int cal; //칼로리
	private int cost; //가격
	private int kg; //중량

	public int getCal() {return cal;}
	public void setCal(int cal) {this.cal = cal;}
	public int getCost() {return cost;}
	public void setCost(int cost) {this.cost = cost;}
	public int getKg() {return kg;}
	public void setKg(int kg) {this.kg = kg;}

	public Food(int cal, int cost, int kg) {
		this.cal = cal;
		this.cost = cost;
		this.kg = kg;
	}
}
 class Melon extends Food{
	 private String info;

	 public String getInfo() {return info;}
	 public void setInfo(String info) {this.info = info;}

	 public Melon(int cal, int cost, int kg, String info) {
		 super(cal, cost, kg);
		 setInfo(info);
	 }
	 @Override
	 public String toString() {
		 return "[Food]"+" 칼로리:"+getCal()+" 가격:"+getCost()+"
중량:"+getKg()+" 원산지:"+getInfo();
	 }
 }
```

[실행코드]

```
public class MelonTest {
	public static void main(String[] args) {
		Melon m = new Melon(100, 5000, 1, "아프리카");
		System.out.println(m);
	}
}
```

2-3. 결과화면


Problems
Javadoc
Declaration
Console
<terminated> MelonTest [Java Application] C:₩Program Files₩Java₩jdk-1
[Food] 칼로리:100 가격:5000 중량:1 원산지:아프리카