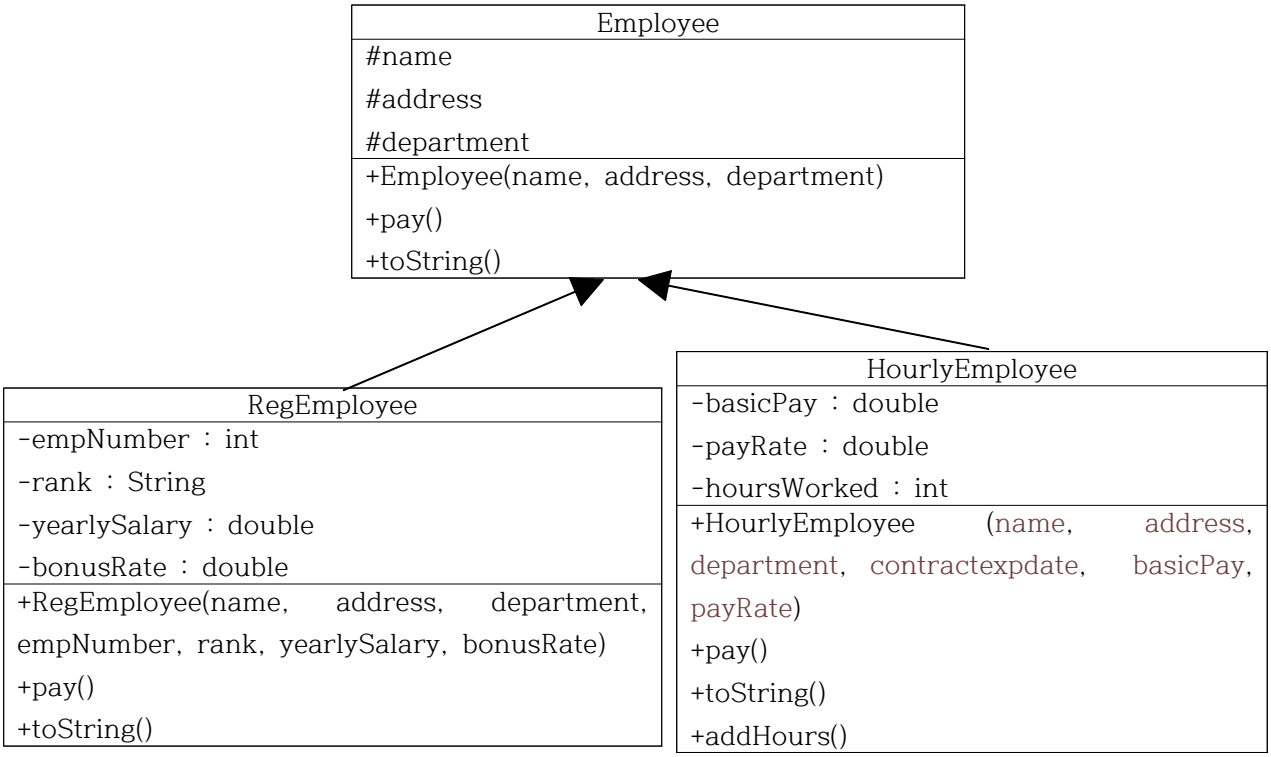
Lab2. 다형성을 이용한 직원 관리

출제자 : 정은미 교수

문제 : 직원(일반, 정규, 시간당)별 월급을 계산해 주는 프로그램을 작성하시오.

목표 : 클래스 설계, 상속을 통한 다형성에 대한 이해 및 구현

1. 클래스



1.1 클래스 구성

- ✓ **Employee class** : 일반직원 클래스
- ✓ **RegEmployee class** : 연봉을 받는 정규 직원 클래스
- ✓ **Hourlyclass** : 시간당 임금을 받는 직원 클래스

1.2 일반 직원 클래스 Employee를 정의한다.

- ✓ 필드
  - ✓ 이름(name)
  - ✓ 주소(address)
  - ✓ 부서(department)
- ✓ 메소드
  - ✓ 생성자 :멤버변수를 초기화
  - ✓ pay() : **추상메소드**
  - ✓ toString():결과 출력

1.3 정규 직원 클래스 RegEmployee를 정의한다.(일반직원 클래스를 상속받아 필요한 부분만 구현)

- ✓ 필드
  - ✓ 이름(name)
  - ✓ 주소(address)
  - ✓ 부서(department)
  - ✓ 직원번호(empNumber)
  - ✓ 직급(rank)
  - ✓ 연봉(yearlySalary)
  - ✓ 보너스율(bonusRate)
- ✓ 메소드
  - ✓ 생성자 :멤버변수를 초기화
  - ✓ pay() : 연간 총 봉급액을 되돌려 준다
    - ✓ 연봉*보너스율 리턴한다.
  - ✓ toString: 결과 출력

1.4 시간당 임금을 받는 클래스 HourlyEmployee를 정의한다.(일반 직원 클래스를 상속받아 필요한 부분만 구현)

- ✓ 멤버변수
  - ✓ 이름(name)
  - ✓ 주소(address)
  - ✓ 부서(department)
  - ✓ 임시직원의 기본 임금(basicPay)
  - ✓ 시간당 임금(payRate)
  - ✓ 근무시간(hoursWorked)
- ✓ 멤버함수
  - ✓ 생성자 :멤버변수를 초기화
  - ✓ pay() : 월급계산
    - ✓ 월급=기본임금+근무시간*시간당임금
  - ✓ toString() : 결과 출력
  - ✓ addHours() : 임시직원의 추가 근무시간을 근무시간에 누적시킨다.

♣ 위와 같이 메소드를 구현해야 하며, 각 메소드의 실행 결과(리턴되는 값의 형식)는 '실행 및 출력 예제'를 참조하기 바람.

2. EmployeeDriver클래스는 다음과 같다.

```
/*  EmployeeDriver.java
    Employee, RegEmployee, TempEmployee, HourlyEmployee와 Intern 클래스를 시험하는
드라이버 프로그램이다.*/
class EmployeeDriver
{
    Employee[] employeeList;

  //-----------------------------------------------------------------
    //  직원들의 목록을 만든다
    //-----------------------------------------------------------------
    public EmployeeDriver()
    {
        employeeList = new Employee[4];

   employeeList[0] = new RegEmployee ("선미", "서소문 123", "마케팅", 9352345, "이사", 7000.0, 1.4);

        employeeList[1] = new RegEmployee ("종미", "대치 456", "인사", 9874321, "과장", 4000.0, 1.2);

        employeeList[2] = new HourlyEmployee ("단형", "유성 678", "기획", "07-07-31", 2000.0, 10.0);

        employeeList[3] = new HourlyEmployee ("범수", "신사 987",  "총무", "08-12-31", 1600.0, 5.0);

        ((HourlyEmployee)employeeList[2]).addHours (30);
        ((HourlyEmployee)employeeList[3]).addHours (40);
        ((HourlyEmployee)employeeList[2]).addHours (50);
        ((HourlyEmployee)employeeList[3]).addHours (60);
        ((HourlyEmployee)employeeList[2]).addHours (20);
        ((HourlyEmployee)employeeList[3]).addHours (30);
    }

    //-----------------------------------------------------------------
    //  모든 직원들에게 봉급을 지불한다
    //-----------------------------------------------------------------
    public void payday ()
    {
        double amount;

        for (int count=0; count < employeeList.length; count++)
        {
            System.out.println (employeeList[count]);

            amount = employeeList[count].pay();  // polymorphic

            System.out.println ("지급액: " + amount);

            System.out.println ("-----------------------------------");
        }

    }
    //-----------------------------------------------------------------
    //  회사의 직원들을 만들고 그들에게 봉급을 지불한다.
    //-----------------------------------------------------------------
    public static void main (String[] args)
    {
            EmployeeDriver personnel = new EmployeeDriver();

            personnel.payday();
    }
}
```

3. 실행 및 출력 예제

```
이름: 선미
주소: 서소문 123
소속부서: 마케팅
직원번호: 9352345
직급: 이사
연봉: 7000.0
보너스 지급율: 1.4
지급액: 9800.0
-----------------------------------
이름: 종미
주소: 대치 456
소속부서: 인사
직원번호: 9874321
직급: 과장
연봉: 4000.0
보너스 지급율: 1.2
지급액: 4800.0
-----------------------------------
이름: 단형
주소: 유성 678
소속부서: 기획
시간당 임금: 10.0
근무 시간: 100
지급액: 1000.0
-----------------------------------
이름: 범수
주소: 신사 987
소속부서: 총무
시간당 임금: 5.0
근무 시간: 130
지급액: 650.0
-----------------------------------
```

* 반드시 위에서 설명된 내용에 따라 구현해야 하며 특별히 기술되지 않은 부분은 여러분의 상상력과 프로그래밍 실력에 맡깁니다.