수행평가(OOP)

경복대학교
소프트웨어융합과
배 희호 교수

수행평가

- 우리 회사(Company)에는 직무에 따라 정규직(Regular), 영업직(Salesman), 점장직(Manager), 일용직(Temporary)의 4종류의 Salary System을 가지고 있다.
  - 일반적인 정규직은 호봉(grade)에 따른 기본급을 지급함
  - 일용직은 정규직이 아니고, 아니고 일당제(Daily Wage System)로 기본급을 산정함
  - 영업직은 정규직으로 판매수당(Commission)을 판매금액(Sale)을 기준으로 추가 지급함
  - 점장직은 정규직으로 기본급을 기준으로 일정 금액의 Incentive를 추가 지급함

수행평가

- 처리 조건
  - 최대 입력 자료의 건수는 10건 이하로 함
  - 사번(id)은 6자리 숫자형 Code로 입력
  - 사원이름(name)은 3자리 이름
  - Test Data의 첫 번째는 반드시 본인의 이름을 등록할 것
  - 정규직의 호봉(grade)은 문자로 (1-5) 입력
  - 호봉에 따른 기본급 산정표

1급	2급	3급	4급	5급
1,600,000	1,800,000	2,100,000	2,400,000	2,800,000

수행평가

- 처리 조건
  - 영업직은 판매 금액에 따른 commission을 추가로 지급함
    - Commission율은 개인별로 다름 (입력 받음)
  - 점장직은 기본급에 따른 incentive가 추가로 지급됨

기본급	incentive
1,800,000 이하 (2급 이하)	6%
1,800,000 초과 2,400,000 이하 (3급 4급)	5%
2,400,000 초과 (5급)	4%

수행평가

- 처리 조건
  - 급여액은 기본급 + 각종 수당
  - 급여액에 따라 세율

급여액 기준	세율
1,000,000 이하	2%
1,000,000 초과 3,000,000 이하	3%
3,000,000 초과	4%

  - 지급액은 급여액에서 세금을 공제한 금액 임

수행평가

- 처리 조건
  - 출력 양식을 참조해서 출력 할 것
  - 출력 시 지급액이 많은 사람부터 작은 사람으로 정렬
  - 지급액의 총합을 표시하여라
  - 영업직 사원의 커미션 금액 산출 내역서를 별도로 출력

수행평가

사원(추상 클래스)

일용직

영업직

정규직

점장직

수행평가

- 사원 클래스

클래스 Employee

사원이름(name)

사번(bunho)

기본급(salary)

급여액(gross)

세금(tax)

지급액(net)

Data 출력하기

수행평가

- 일용직 클래스

일용직(Temporary) – 일당제

클래스(Employee)

일수(day)

일당(ildang)

호봉이 없음

기본급 계산하기

급여액 계산하기

지급액 계산하기

수행평가

- 정규직 클래스

정규직(Regular) -고정급

클래스(Employee)

호봉(grade)

기본급 계산하기

급여액 계산하기

지급액 계산하기

수행평가

- 영업직 클래스

영업직(Salesman) – 성과급

클래스(Regular)

클래스(Employee)

판매금(sales)

커미션
(commission)

보너스(bonus)

기본급 계산하기

급여액 계산하기

지급액 계산하기

수행평가

- 점장직 클래스

점장직(Manager) – 성과급

클래스(Regular)

클래스(Employee)

인센티브
(incentive)

기본급 계산하기

급여액 계산하기

지급액 계산하기

수행평가

- Data 입력

```
[일용직] 이대한님의 일당 입력 : 78900
[일용직] 이대한님의 작업 일수 입력 : 25

[정규직] 진접읍님의 급수 입력 : 4

[영업직] 최명길님의 급수 입력 : 5
[영업직] 최명길님의 판매 금액 입력 : 34568000
[영업직] 최명길님의 커미션 비율 입력 : 1.5

[영업직] 정통파님의 급수 입력 : 3
[영업직] 정통파님의 판매 금액 입력 : 3453900
[영업직] 정통파님의 커미션 비율 입력 : 1.6

[정규직] 코로나님의 급수 입력 : 5

[점장직] 한민국님의 급수 입력 : 3
```

수행평가

실행 결과

경복주식회사 급여 대장

사번	이름	급	day	일당	기본급	인센티브	커미션	급여액	세금	지급액	비고
424561	최명길	5급	0	0원	2,800,000원	0원	518,520원	3,318,520원	132,740원	3,185,780원	영업직
348988	코로나	5급	0	0원	2,800,000원	0원	0원	2,800,000원	84,000원	2,716,000원	정규직
234567	홍길동	4급	0	0원	2,400,000원	0원	0원	2,400,000원	72,000원	2,328,000원	정규직
348967	진접읍	4급	0	0원	2,400,000원	0원	0원	2,400,000원	72,000원	2,328,000원	정규직
124567	한송이	3급	0	0원	2,100,000원	0원	109,656원	2,209,656원	66,289원	2,143,367원	영업직
245778	한민국	3급	0	0원	2,100,000원	105,000원	0원	2,205,000원	66,150원	2,138,850원	점장직
124655	정통파	3급	0	0원	2,100,000원	0원	55,262원	2,155,262원	64,657원	2,090,605원	영업직
123456	경복대	급	23	92,500원	2,127,500원	0원	0원	2,127,500원	63,825원	2,063,675원	일당제
456213	이대한	급	25	78,900원	1,972,500원	0원	0원	1,972,500원	59,175원	1,913,325원	일당제
345678	한국인	2급	0	0원	1,800,000원	108,000원	0원	1,908,000원	57,240원	1,850,760원	점장직

지급액 합계 : 22,758,362 원

수행평가

- 실행 결과

영업직 사원 커미션 산출 내역

사번	이름	급	판매 실적	요율	커미션 금액
424561	최명길	5급	34,568,000원	1.5 %	518,520
124567	한송이	3급	4,569,000원	2.4 %	109,656
124655	정통파	3급	3,453,900원	1.6 %	55,262

수행평가

- Class Diagram을 그려보자
  - Data 관리를 하는 회사(Company) 클래스가 필요함

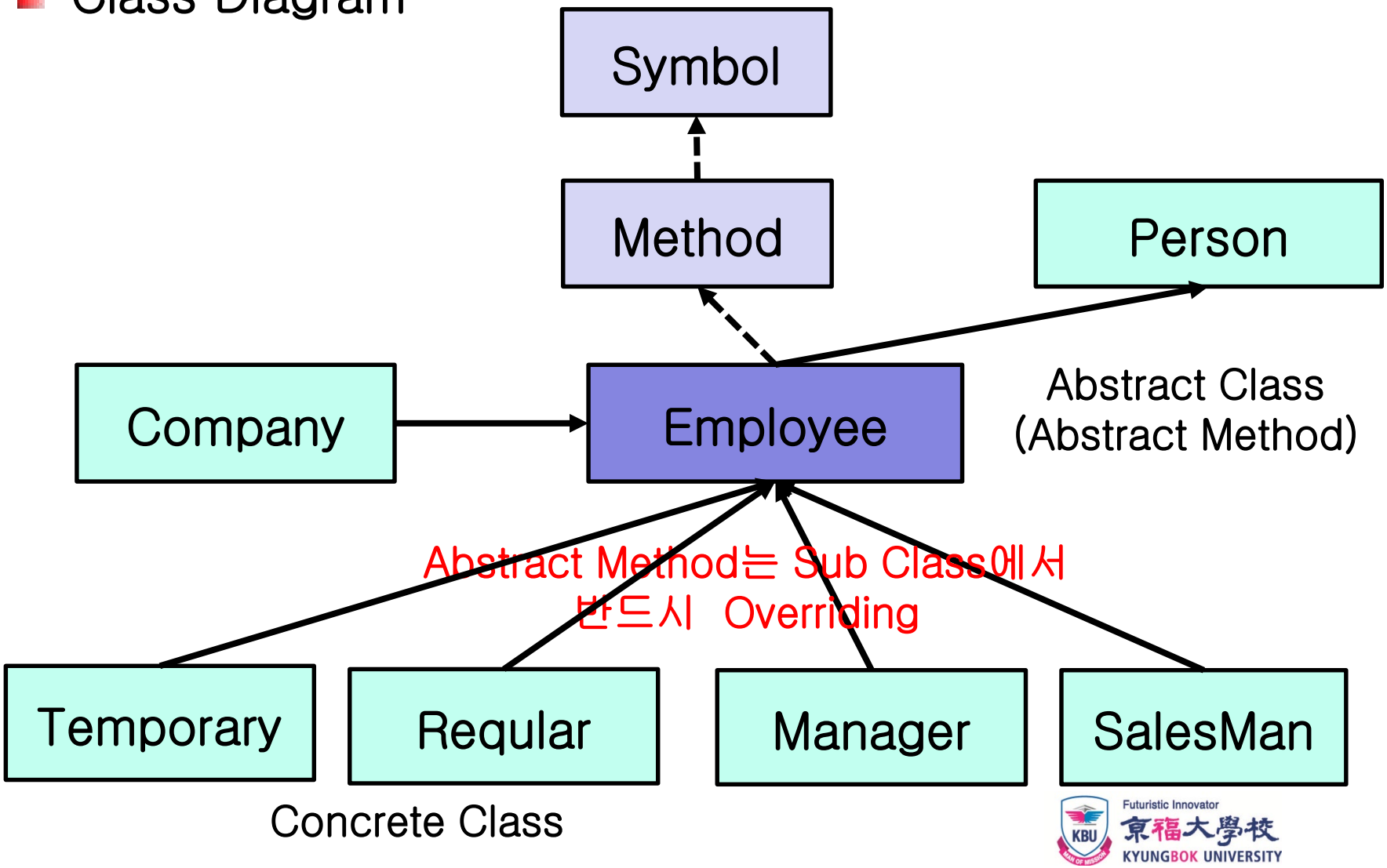
- Data 관리
  - Data 입출력
  - Data 정렬

수행평가

- 제출 방법
  - Program의 최종 파일을 Project Folder를 압축 파일로 제출 할 것
  - 반드시 실행 화면을 Capture한 것을 해당 폴더에 저장하고 압축할 것
  - 폴더의 모든 파일을 압축할 것
  - 압축 시 파일 이름은 이름-학번.zip으로 할 것
  - 예) 홍길동이면서 학번이 1602345이면 파일 이름을 “수행2-홍길동-1602345.ZIP”으로 함

수행평가

- Class Diagram



수행평가

- SymbolicDefine Interface

```
public interface SymbolDefine {
    String[] type = {"일용직", "정규직", "영업직", "점장직"};

    int SALARY1 = 1600000;
    int SALARY2 = 1800000;
    int SALARY3 = 2100000;
    int SALARY4 = 2400000;
    int SALARY5 = 2800000;

    int TAX1 = 1000000;
    int TAX2 = 3000000;

    int BONUS1 = 1800000;
    int BONUS2 = 2400000;
}
```

수행평가

- Method Interface

```
public interface Method extends SymbolDefine{
    void inputData(String type) throws IOException;
    int salary();
    int gross();
    int managerBonus();
    int salesBonus();

    default int tax() {
        int tax;
        int temp = gross();
        if (temp <= TAX1) {
            tax = (int) (temp * (2.0f / 100));
        } else if (temp <= TAX2)
            tax = (int) (temp * (3.0f / 100));
        else
            tax = (int) (temp * (4.0f / 100));

        return tax;
    }
```

수행평가

- Method Interface

```
    default int net() {
        return gross() - tax();
    }
}
```

수행평가

- Person Class

```
public class Person {
    final private String name;
    final private String bunho;

    public Person(String name, String bunho) {
        this.name = name;
        this.bunho = bunho;
    }

    public String getName() {
        return name;
    }

    @Override
    public String toString() {
        return String.format("%6s %3s", bunho, name);
    }
}
```

수행평가

- Employee Class

```
abstract public class Employee extends Person implements Method {
    private char grade;

    public Employee(String name, String bunho) {
        super(name, bunho);
    }

    public void setGrade(char grade) {
        this.grade = grade;
    }
```

수행평가

- Employee Class

```
public void inputData(String type) throws IOException {
    Scanner keyboard = new Scanner(System.in);
    while (true) {
        System.out.printf("[%s] %s님의 급수 입력 : ", type, getName());
        grade = keyboard.next().charAt(0);
        if (grade >= '1' && grade <= '5')
            break;
        else {
            System.err.print("ERROR : 급수는 1 ~ 5급 입니다");
            System.in.read();
        }
    }
}
```

수행평가

```
public int salary() {
    int salary = 0;
    switch (grade) {
        case '1':
            salary = SALARY1;
            break;
        case '2':
            salary = SALARY2;
            break;
        case '3':
            salary = SALARY3;
            break;
        case '4':
            salary = SALARY4;
            break;
        case '5':
            salary = SALARY5;
    }
    return salary;
}
```

수행평가

- Employee Class

```
 @Override
public int gross() {
    return salary();
}

@Override
public int managerBonus() {
    return 0;
}

@Override
public int salesBonus() {
    return 0;
}
```

수행평가

- Employee Class

```
    public String toStringParent() {
        return String.format("%s %2c급", super.toString(), grade);
    }

    @Override
    public String toString() {
        return String.format("%s %2c급 %2d %,7d원 %,9d원 %,9d원
                                                %,9d원 %,9d원 %,7d원 %,9d원",
                super.toString(), grade, 0, 0, salary(),
                                managerBonus(), salesBonus(), gross(), tax(), net());
    }
}
```

수행평가

- Temporary Class

```
public class Temporary extends Employee {
    private int workday;
    private int dailyPay;

    public Temporary(String name, String bunho) {
        super(name, bunho);
        setGrade(' ');
    }
```

수행평가

- Temporary Class

```
@Override
public void inputData(String type) throws IOException {
    Scanner keyboard = new Scanner(System.in);
    while (true) {
        System.out.printf("[%s] %s님의 일당 입력 : ", type, getName());
        dailyPay = keyboard.nextInt();
        if (dailyPay > 0)
            break;
        else {
            System.err.println("ERROR : 양수임");
            System.in.read();
        }
    }
```

수행평가

- Temporary Class

```
        while (true) {
            System.out.printf("[%s] %s님의 작업 일수 입력 : ", type, getName());
            workday = keyboard.nextInt();
            if (workday >= 0 && workday <= 31)
                break;
            else {
                System.err.println("ERROR : 31일을 넘지 못함");
                System.in.read();
            }
        }
    }
```

수행평가

- Temporary Class

```
    @Override
    public int salary() {
        return workday * dailyPay;
    }

    @Override
    public String toString() {
        return String.format("%s %2d %,7d원 %,9d원 %,9d원 %,9d원 %,9d원
                                        %,7d원 %,9d원 일당제",
                super.toStringParent(), workday, dailyPay, salary(),
                                managerBonus(), salesBonus(), gross(), tax(), net());
    }
}
```

수행평가

- Regular Class

```
public class Regular extends Employee{

    public Regular(String name, String bunho) {
        super(name, bunho);
    }

    @Override
    public String toString() {
        return String.format("%s 정규직", super.toString());
    }
}
```

수행평가

- Manager Class

```
public class Manager extends Employee {

    public Manager(String name, String bunho) {
        super(name, bunho);
    }

    @Override
    public int managerBonus() {
        int incentive;
        int salary = salary();
        if (salary <= BONUS1)
            incentive = (int) (salary * (6.0f / 100));
        else if (salary <= BONUS2)
            incentive = (int) (salary * (5.0f / 100));
        else
            incentive = (int) (salary * (4.0f / 100));
        return incentive;
    }
```

수행평가

- Manager Class

```
    @Override
    public int gross() {
        return salary() + managerBonus();
    }

    @Override
    public String toString() {
        return String.format("%s 점장직",super.toString());
    }
}
```

수행평가

- Manager Class

```
public class SalesMan extends Employee {
    private int sales;
    private double commission;

    public SalesMan(String name, String bunho) {
        super(name, bunho);
    }

    public int getSales() {
        return sales;
    }

    public double getCommission() {
        return commission;
    }
```

수행평가

- Manager Class

```
@Override
public void inputData(String type) throws IOException {
    super.inputData(type);
    Scanner keyboard = new Scanner(System.in);
    while (true) {
        System.out.printf("[%s] %s님의 판매 금액 입력 : ", type, getName());
        sales = keyboard.nextInt();
        if (sales >= 0)
            break;
        else {
            System.err.println("ERROR : 판매 금액은 양수 입니다");
            System.in.read();
        }
    }
```

수행평가

- Manager Class

```
        while (true) {
            System.out.printf("[%s] %s님의 커미션 비율 입력 : ", type, getName());
            commission = keyboard.nextDouble();
            if (commission >= 0.0)
                break;
            else {
                System.err.println("ERROR : 양수 입니다");
                System.in.read();
            }
        }
    }
```

수행평가

- Manager Class

```
    @Override
    public int salesBonus() {
        return (int) (sales * (commission / 100));
    }

    @Override
    public int gross() {
        return salary() + salesBonus();
    }

    @Override
    public String toString() {
        return String.format("%s 영업직", super.toString());
    }
}
```

수행평가

- Company Class

```
public class Company {
    private String name;
    private Employee[] employees;

    public Company(String name, Employee[] employees) {
        this.name = name;
        this.employees = employees;
    }
```

수행평가

- Company Class

```
private void sort() {
    Employee temp;
    for (int i = 0; i < employees.length - 1; i++) {
        for (int j = i + 1; j < employees.length; j++) {
            if (employees[i].net() < employees[j].net()) {
                temp = employees[j];
                employees[j] = employees[i];
                employees[i] = temp;
            }
        }
    }
}

private int netTotal() {
    int total = 0;
    for (int i = 0; i < employees.length; i++)
        total += employees[i].net();
    return total;
}
```

수행평가

- Company Class

```
public void display() {
    sort();
    System.out.println("₩t₩t₩t₩t" + name + " 급여 대장 ");
    line(105);
    System.out.println(" 사번    이름   급  day    일당     기본급      인센티브
                                커미션     급여액      세금     지급액     비고");
    line(105);
    for (int i = 0; i < employees.length; i++) {
        System.out.println(employees[i]);
    }
    line(105);
    System.out.printf("₩t₩t₩t₩t₩t 지급액 합계 : %,18d 원₩n", netTotal());
    line(105);
```

수행평가

```
        System.out.printf("₩n₩n₩n₩n₩t₩t₩t₩t 영업직 사원 커미션 산출 내역₩n");
        line(60);
        System.out.println(" 사번    이름   급        판매 실적      요율      커미션 금액");
        line(60);
        for (int i = 0; i < employees.length; i++) {
            if (employees[i] instanceof SalesMan)
                System.out.printf("%s %,16d원 %4.1f %% %,10d₩n",
                        (employees[i]).toStringParent(),
                          ((SalesMan) employees[i]).getSales(),
                        ((SalesMan) employees[i]).getCommission(),
                                  employees[i].salesBonus());
        }
        line(60);
    }
    private void line(int count) {
        for (int i = 0; i < count; i++)
            System.out.print("*");
        System.out.println();
    }
}
```

수행평가

- Main Class

```
public class Main implements SymbolDefine{
    public static void main(String[] args) throws IOException {
        Employee[] employees = new Employee[]{
                new Temporary("경복대", "123456"),
                new Regular("홍길동", "234567"),
                new SalesMan("한송이", "124567"),
                new Manager("한국인", "345678"),
                new Temporary("이대한", "456213"),
                new Regular("진접읍", "348967"),
                new SalesMan("최명길", "424561"),
                new SalesMan("정통파", "124655"),
                new Regular("코로나", "348988"),
                new Manager("한민국", "245778")  };
```

수행평가

- Main Class

```
        for (int i = 0; i < employees.length; i++) {
            if (employees[i] instanceof Temporary)
                employees[i].inputData(type[0]);
            else if (employees[i] instanceof Regular)
                employees[i].inputData(type[1]);
            else if (employees[i] instanceof SalesMan)
                employees[i].inputData(type[2]);
            else
                employees[i].inputData(type[3]);
             System.out.println();
        }

        Company company = new Company("경복주식회사", employees);
        company.display();
    }
}
```