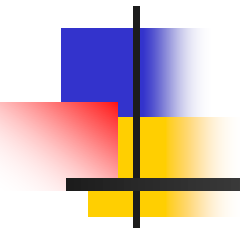

급여계산
(클래스 상속)

경복대학교
소프트웨어융합과
배희호 교수

급여 계산

- 다음에서 제시하는 처리 조건에 따라 사원이름(name), 사번(bunho), 호봉(hobong), 수당(extra_pay)을 객체 배열에 입력 받은 후, 아래의 처리 조건에 의하여 급여(salary)를 계산하는 프로그램을 작성하시오.

급여 계산

실행 결과

```
**************************************************************************************
        이름                    사번   호봉       수당         본봉       세금        급여
**************************************************************************************
 박정환(2012년  7월 20일)      1422    4     890,000      720,000    1,610    1,608,390
 황정민(2012년  4월 20일)      1456    4     890,000      720,000    1,610    1,608,390
 홍길동(1993년 10월  5일)      1112    1     890,000      650,000    1,540    1,538,460
 김대한(1999년  5월 13일)      3456    5     789,000      750,000    1,539    1,537,461
 이나라(2010년  3월  7일)      1345    2     789,000      680,000    1,469    1,467,531

 박대국(2011년  8월 10일)      1789    2     789,000      680,000    1,469    1,467,531
 하충민(2012년  3월 20일)      4521    1     789,000      650,000    1,439    1,437,561
 길대연(2013년 12월 20일)      1345    5     678,900      750,000    1,428    1,427,472
 나정산(2011년  4월 19일)      1362    3     678,000      700,000    1,378    1,376,622
 이민국(1987년  6월 22일)      5634    1     678,900      650,000    1,328    1,327,572
**************************************************************************************
                                      7,861,800    6,950,000       14,796,990
**************************************************************************************
```

급여 계산

- 입력 설계
  - 최대 입력 자료의 건수는 10건 이상으로 함
  - 사번(bunho)은 4자리 숫자형 문자로 입력
  - 사원이름(name)은 3자리 이름
    - 테스트 데이터의 첫 번째는 반드시 본인의 이름을 등록할 것
  - 호봉(hobong)은 문자로 (1 ~ 5) 입력
  - 수당(extra_pay)은 0 ~ 1,000,000원까지로 입력

급여 계산

- 입력 예시

사번	사원이름	호봉	수당
9111	홍길동	1	400,000
2233	이한국	3	650,000

  - 레코드 별 입력 순서는 “사번, 사원이름, 호봉, 수당”으로 한다
  - 자료의 입력은 사번, 사원이름, 호봉은 초기화하여 받을 수 있음

급여 계산

- 처리 조건
  - 호봉에 따른 지급액

1호봉	2호봉	3호봉	4호봉	5호봉
1,650,000	1,680,000	1,700,000	1,720,000	1,750,000

  - 총 지급액에 따라 세율

총 지급액 기준	세율
1,700,000미만	5%
1,700,000이상 2,300,000미만	8%
2,300,000이상	10%

급여 계산

- 처리 조건
  - 지급액은 호봉에 따라 지급
  - 총 지급액은 (지급액 + 수당)으로 함
  - 세금은 (총 지급액 * 세율)로 하여 100원 이하는 버림
  - 실 지급액은 (총 지급액 – 세금)으로 함
  - 합계(Total)는 수당, 총계, 세금, 실 지급액을 각각 누적한 금액 임
  - 실 지급액(NET)이 많은 사람부터 작은 사람으로 정렬

급여 계산

클래스 Man

사원이름(name)

사번(bunho)

생일 (birthday)

데이터 출력하기

급여 계산

클래스 Employee

호봉(hobong)

수당(extra_pay)

데이터 입력하기　데이터 출력하기

지급액 계산하기　총 지급액 계산하기　세금 계산하기

실 지급액 계산하기

급여 계산

클래스 Company(회사)

정보 출력하기

합계 구하기　정렬하기

Object Type 변환

- 기본형 변수처럼 참조 변수도 형 변환(Casting)이 가능
- 단, 서로 상속 관계에 있는 Class 사이에서만 가능

```
Parent parent = new Child();
```

- 위의 대입 연산에서 왼쪽 항(부모, Parent)과 오른쪽 항(자식, child)의 Object 유형이 서로 다른 경우, 두 유형이 서로 상속 관계에 있고 왼쪽 Object가 오른쪽 Object의 Super Class인 경우에만 암묵적 Type 변환이 일어남
- Sub Class에서 Super Class 유형으로 할당하는 것은 가능하나 그 반대의 경우에는 명시적 형 변환을 해야 함

Object Type 변환

- 암묵적 형 변환
  - 자식 Class가 부모 Class에게 상속받은 기능만 사용하도록 제한
  - 주의할 점은 기능의 제한이지 기능의 변경은 아니라는 것
  - 모든 Object는 자신의 상위 형태로 암묵적 형 변환이 가능
  - 즉 Override된 기능만 사용 가능하고, 추가적으로 구현한 기능은 사용할 수 없음

Object Type 변환

- 명시적 형 변환
  - 부모 Class의 Object를 자식 Class 형태로 변환하는 것
  - 형 변환을 위해서는 다음과 같이 변환할 Class 이름을 명시적으로 지정해 주어야 함

```
ChildClass child = (ChildClass) parent;
```

  - Object가 최초로 생성될 때 자식 Class 형태로 생성되고, 부모 형태로 암묵적 형 변환이 된 상태를 다시 원래의 자식 Class 형태로 되돌릴 경우에만 가능

```
ChildClass child1 = new ChildClass();
ParentClass parent = child1;      // 암묵적 형 변환
ChildClass child2 = (ChildClass) parent;//명시적 형 변환
```

Object Type 변환

- 상속 관계의 클래스에서 객체의 형변환 가능

```
class Acast {
                    int a =1; }
class Bcast extends Acast  {
                    int b = 2; }
class Ccast extends Bcast  {
                    int c = 3; }

class TestCasting {
   public static void main(String[] args) {
      Acast refA;            // Acast 타입의 객체 refA 선언
      refA = new Ccast();
      /* Acast 타입의 객체 참조 변수 refA에 Ccast 클래스의 객체를
         생성하여 할당 */
      System.out.println("refA.a의 값은 " + refA.a);
   }
}
```

출력결과
refA.a의 값은 1

Object Type 변환

- 앞의 프로그램을 다음과 같이 수정

```
class TestCasting {
    public static void main(String[] args) {
     Acast refA;
     refA = new Ccast();
     System.out.println("refA.a의 값은 "+ refA.c );
     // Ccast 클래스의 멤버인 c에 접근 시도. 에러 발생
    }
  }
```

```
TestCasting.java:13: No variable c defined in class Acast.
    System.out.println("refA.a의 값은 "+refA.c);
                                        ^
1 error
```

Object Type 변환

- 앞의 프로그램을 반대로 다음과 같이 수정

```
class TestCasting  {
      public static void main(String[] args)  {
          Ccast refC = new Acast();
         // 에러 발생
         System.out.println("refC.a의 값은 "+refC.a);
    }
  }
```

```
TestCasting.java:12: Incompatible type for declaration.
Explicit cast needed to convert Acast to Ccast.
    Ccast refC = new Acast();
          ^
1 error
```

급여 계산(I)

Date 클래스

```
public class Date {
    private int year;
    private int month;
    private int day;

    public Date(int year, int month, int day) {
        this.year = year;
        this.month = month;
        this.day = day;
    }

    @Override
    public String toString() {
        return String.format("(%4d년 %2d월 %2d일)", year, month, day);
    }
}
```

급여 계산(I)

- Employee.JAVA

```
public class Employee extends Man{
    private char hobong;
    private int extra_pay;

    public Employee(String name, String bunho, Date birthday, char hobong) {
        super(name, bunho, birthday);
        this.hobong = hobong;
    }

    public Employee(String name, String bunho, Date birthday,
                                                    char hobong, int extra_pay) {
        this(name, bunho, birthday, hobong);
        this.extra_pay = extra_pay;
    }

    public void setExtra_pay(int extra_pay) {
        this.extra_pay = extra_pay;
    }
```

급여 계산(I)

- Employee.JAVA

```
public int getExtra_pay() {
    return extra_pay;
}

public int input() throws IOException {
    Scanner keyboard = new Scanner(System.in);
    int pay;
    while (true) {
        System.out.printf("%s님의 수당 입력 : ", getName());
        pay = keyboard.nextInt();
        if (pay >= 0 && pay <= 1000000)
            break;
        else {
            System.err.println("입력 오류");
            System.in.read();
        }
    }
    return pay;
}
```

급여 계산(I)

```
public int gross() {
    int gross = 0;
    switch(hobong) {
        case '1':
            gross = 650000;
            break;
        case '2':
            gross = 680000;
            break;
        case '3':
            gross = 700000;
            break;
        case '4':
            gross = 720000;
            break;
        case '5':
            gross = 750000;
    }
    return gross;
}
```

급여 계산(I)

- Employee.JAVA

```
public int net() {
    return extra_pay + gross() - tax();
}

public int tax() {
    int total = extra_pay + gross();
    int tax;
    if (total < 700000) {
        tax = (int)((double)total * 0.05D * 1.0D / 100.0D);
    } else if (total < 750000) {
        tax = (int)((double)total * 0.08D * 1.0D / 100.0D);
    } else {
        tax = (int)((double)total * 0.1D * 1.0D / 100.0D);
    }
    return tax;
}
```

급여 계산(I)

- Employee.JAVA

```
    @Override
    public String toString() {
        return  super.toString() + String.format(" %3c %,10d %,10d %,6d %,10d",
                hobong, extra_pay, gross(), tax(), net());
    }
}
```

급여 계산(I)

- Company.JAVA

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

    public Company(Employee[] employees) {
        this.employees = employees;
    }

    public Employee getEmployees(int index) {
        return employees[index];
    }

    public int getLength() {
        return employees.length;
    }
```

급여 계산(I)

- Company.JAVA

```
private int sudangtotal() {
    int temp = 0;
    for (int i = 0; i < employees.length; i++)
        temp += employees[i].getExtra_pay();

    return temp;
}

private int bonbongtotal() {
    int temp = 0;
    for (int i = 0; i < employees.length; i++)
        temp += employees[i].gross();

    return temp;
}
```

급여 계산(I)

- Company.JAVA

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

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[i];
                employees[i] = employees[j];
                employees[j] = temp;
            }
        }
    }
}
```

급여 계산(I)

```
private String total() {
    return String.format("₩t₩t₩t₩t₩t₩t₩t %,10d  %,10d ₩t %,10d",
            sudangtotal(), bonbongtotal(), nettotal());
}

public void display() {
    sort();
    line();
    System.out.println("        이름              사번  호봉      수당
                                                    본봉     세금      급여");
    line();
    for(int i = 0; i < employees.length; ++i) {
        if (i != 0 && i % 5 == 0)
            System.out.println();
        System.out.println(employees[i]);
    }
    line();
    System.out.println(total());
    line();
}
```

급여 계산(I)

- Company.JAVA

```
    private void line() {
        System.out.println("**********************************************
                                                    **********************************");
    }
}
```

급여 계산(I)

- Main.JAVA

```
public static void main(String[] args) throws IOException {
    Employee[] person = new Employee[]{
            new Employee("홍길동", "1112", new Date(1993, 10, 5),'1'),
            new Employee("이나라", "1345", new Date(2010, 3, 7),'2'),
            new Employee("나정산", "1362", new Date(2011, 4, 19),'3'),
            new Employee("박정환", "1422", new Date(2012, 7, 20),'4'),
            new Employee("김대한", "3456", new Date(1999, 5, 13),'5'),
            new Employee("이민국", "5634", new Date(1987, 6, 22),'1'),
            new Employee("황정민", "1456", new Date(2012, 4, 20),'4'),
            new Employee("길대연", "1345", new Date(2013, 12, 20),'5'),
            new Employee("박대국", "1789", new Date(2011, 8, 10),'2'),
            new Employee("하충민", "4521", new Date(2012, 3, 20), '1') };
```

급여 계산(I)

- Main.JAVA

```
        for (int i = 0; i < person.length; i++)
            person[i].setExtra_pay(person[i].input());

        Company company = new Company(person);
        company.display();
    }
}
```

급여 계산(II)

- Company.JAVA

```
public class Company {
    private ArrayList<Employee> employees;

    public Company(ArrayList<Employee> employees) {
        this.employees = employees;
    }

    private int sudangtotal() {
        int temp = 0;
        for (int i = 0; i < employees.size(); i++)
            temp += employees.get(i).getExtra_pay();

        return temp;
    }
```

급여 계산(II)

- Company.JAVA

```
private int bonbongtotal() {
    int temp = 0;
    for (int i = 0; i < employees.size(); i++)
        temp += employees.get(i).gross();

    return temp;
}

private int nettotal() {
    int temp = 0;
    for (int i = 0; i < employees.size(); i++)
        temp += employees.get(i).net();

    return temp;
}
```

급여 계산(II)

- Company.JAVA

```
private void sort() {
    Collections.sort(employees, new netComparator().reversed());
}

private class netComparator implements Comparator<Employee> {
    @Override
    public int compare(Employee o1, Employee o2) {
        return o1.net() - o2.net();
    }
}

private String total() {
    return String.format("₩t₩t₩t₩t₩t₩t₩t %,10d  %,10d ₩t %,10d",
            sudangtotal(), bonbongtotal(), nettotal());
}
```

급여 계산(II)

```
    public void display() {
        sort();
        line();
        System.out.println(" 이름    사번  호봉      수당        본봉    세금      급여");
        line();
        for(int i = 0; i < employees.size(); ++i) {
            if (i != 0 && i % 5 == 0)
                System.out.println();
            System.out.println(employees.get(i));
        }
        line();
        System.out.println(total());
        line();
    }

    private void line() {
        System.out.println("-----------------------------------------------------");
    }
}
```

급여 계산(II)

- Main.JAVA

```
public class Main {
    public static void main(String[] args) throws IOException {
        ArrayList<Employee> employees = new ArrayList<>();
        employees.add(new Employee("홍길동", "1112", new Date(1993, 10, 5),'1'));
        employees.add(new Employee("이나라", "1345", new Date(2010, 3, 7),'2'));
        employees.add(new Employee("나정산", "1362", new Date(2011, 4, 19),'3'));
        employees.add(new Employee("박정환", "1422", new Date(2012, 7, 20),'4'));
        employees.add(new Employee("김대한", "3456", new Date(1999, 5, 13),'5'));
        employees.add(new Employee("이민국", "5634", new Date(1987, 6, 22),'1'));
        employees.add(new Employee("황정민", "1456", new Date(2012, 4, 20),'4'));
        employees.add(new Employee("길대연", "1345", new Date(2013, 12, 20),'5'));
        employees.add(new Employee("박대국", "1789", new Date(2011, 8, 10),'2'));
        employees.add(new Employee("하충민", "4521", new Date(2012, 3, 20), '1'));
```

급여 계산(II)

- Main.JAVA

```
        for (int i = 0; i < employees.size(); i++)
            employees.get(i).setExtra_pay(employees.get(i).input());

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