수업 **복습 노트**

강사 : **이상훈**
학생 : **여인준**

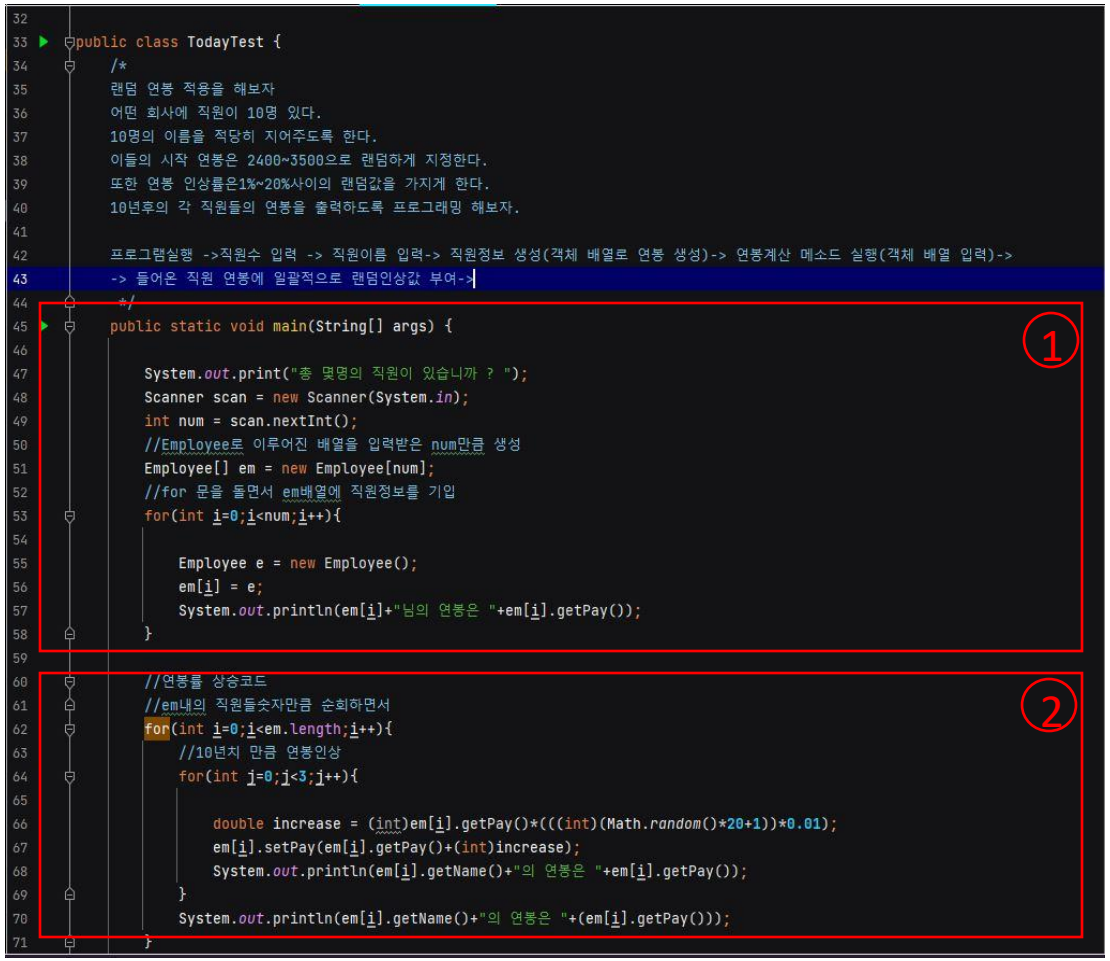
-10일차 수업-

수업 복습(객체 배열 복습)

```java
import java.util.Scanner;

class Employee{

    Scanner scan;
    private String name;

    private int pay;

    public Employee(){
        System.out.print("직원 이름을 입력해주세요 : ");
        Scanner scan = new Scanner(System.in);
        this.name= scan.nextLine();
        pay =(int)(Math.random()*1100+2400);
    }

    public int getPay() { return pay; }
    public void setPay(int pay){this.pay = pay;}
    public String getName() { return name; }

```

1.Employee 클래스를 만들고, 필요하다고 생각한 이름(name)과 연봉(pay)을 객체변수로 설정했다.
2.입력을 받아 작성하고 싶었기에 Scanner를 줘서 생성자를 만들었다.
3.private로 설정한 객체변수에 대한 setter메소드와 getter메소드를 만들었다.

수업 복습(객체 배열 복습)



```
32
33  public class TodayTest {
34      /*
35      랜덤 연봉 적용을 해보자
36      어떤 회사에 직원이 10명 있다.
37      10명의 이름을 적당히 지어주도록 한다.
38      이들의 시작 연봉은 2400~3500으로 랜덤하게 지정한다.
39      또한 연봉 인상률은1%~20%사이의 랜덤값을 가지게 한다.
40      10년후의 각 직원들의 연봉을 출력하도록 프로그래밍 해보자.
41
42      프로그램실행 ->직원수 입력 -> 직원이름 입력-> 직원정보 생성(객체 배열로 연봉 생성)-> 연봉계산 메소드 실행(객체 배열 입력)->
43      -> 들어온 직원 연봉에 일괄적으로 랜덤인상값 부여->
44       */
45      public static void main(String[] args) {
46
47          System.out.print("총 몇명의 직원이 있습니까 ? ");
48          Scanner scan = new Scanner(System.in);
49          int num = scan.nextInt();
50          //Employee로 이루어진 배열을 입력받은 num만큼 생성
51          Employee[] em = new Employee[num];
52          //for 문을 돌면서 em배열에 직원정보를 기입
53          for(int i=0;i<num;i++){
54
55              Employee e = new Employee();
56              em[i] = e;
57              System.out.println(em[i]+"님의 연봉은 "+em[i].getPay());
58          }
59
60          //연봉률 상승코드
61          //em내의 직원들숫자만큼 순회하면서
62          for(int i=0;i<em.length;i++){
63              //10년치 만큼 연봉인상
64              for(int j=0;j<3;j++){
65
66                  double increase = (int)em[i].getPay()*(((int)(Math.random()*20+1))*0.01);
67                  em[i].setPay(em[i].getPay()+(int)increase);
68                  System.out.println(em[i].getName()+"의 연봉은 "+em[i].getPay());
69              }
70              System.out.println(em[i].getName()+"의 연봉은 "+(em[i].getPay()));
71          }
```

①

1.입력을 받아 직원수를 정하고,
2.정한 직원수(num) 만큼 Employee클래스를 데이터로 가질 배열(em)을 생성.
3.for 문을 직원수 만큼 돌면서 Employee객체를 생성하고, 그 생성된 객체를 배열(em)에 넣는다.

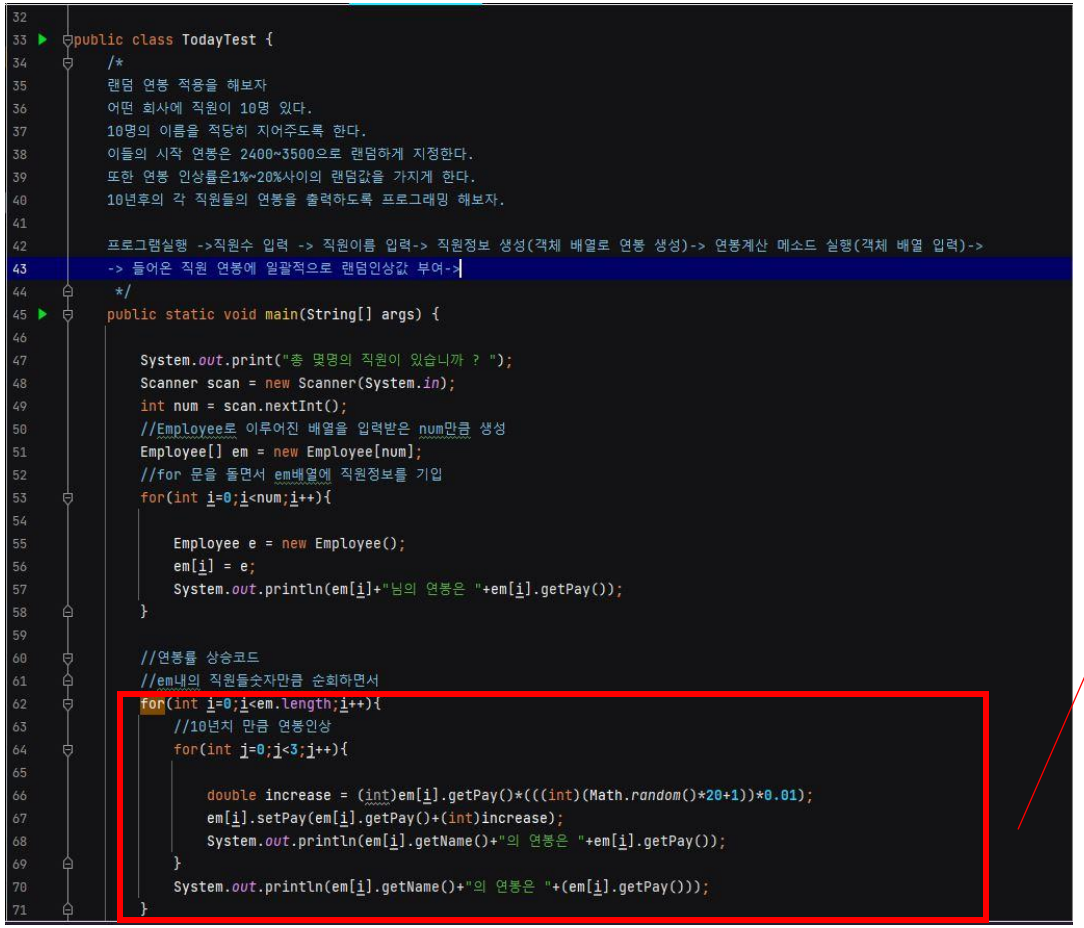
②

1.for문을 직원수(em.length)만큼 돌면서,
2.그 안에 for문(즉, 직원한사람마다 개별적인 for문 을실행)을 10번 (사진은 3번) 돈다.
3.이중 for문 안의 내용은 매년 연봉인상률(increase)을 현재 연봉(pay)에 더하는 코드이다. Employee의 pay가 private기 때문에 get,set 메소드가 유용하게 쓰인다.

생각 할 만 한것

1 . 연봉 인상률(increase)코드를 좀 더 최적화 할 순 없을까?
2. 메소드의 목적을 생각 했을때 좀 더 세분화된 메소드를 만들어 재활용성을 높일 수 있을 것 같다.

수업 복습(메소드 세분화)



```java
public class TodayTest {
    /*
    랜덤 연봉 적용을 해보자
    어떤 회사에 직원이 10명 있다.
    10명의 이름을 적당히 지어주도록 한다.
    이들의 시작 연봉은 2400~3500으로 랜덤하게 지정한다.
    또한 연봉 인상률은1%~20%사이의 랜덤값을 가지게 한다.
    10년후의 각 직원들의 연봉을 출력하도록 프로그래밍 해보자.

    프로그램실행 ->직원수 입력 -> 직원이름 입력-> 직원정보 생성(객체 배열로 연봉 생성)-> 연봉계산 메소드 실행(객체 배열 입력)->
    -> 들어온 직원 연봉에 일괄적으로 랜덤인상값 부여->
     */
    public static void main(String[] args) {

        System.out.print("총 몇명의 직원이 있습니까 ? ");
        Scanner scan = new Scanner(System.in);
        int num = scan.nextInt();
        //Employee로 이루어진 배열을 입력받은 num만큼 생성
        Employee[] em = new Employee[num];
        //for 문을 돌면서 em배열에 직원정보를 기입
        for(int i=0;i<num;i++){

            Employee e = new Employee();
            em[i] = e;
            System.out.println(em[i]+"님의 연봉은 "+em[i].getPay());
        }

        //연봉률 상승코드
        //em내의 직원들숫자만큼 순회하면서
        for(int i=0;i<em.length;i++){
            //10년치 만큼 연봉인상
            for(int j=0;j<3;j++){

                double increase = (int)em[i].getPay()*(((int)(Math.random()*20+1))*0.01);
                em[i].setPay(em[i].getPay()+(int)increase);
                System.out.println(em[i].getName()+"의 연봉은 "+em[i].getPay());
            }
            System.out.println(em[i].getName()+"의 연봉은 "+(em[i].getPay()));
        }
```

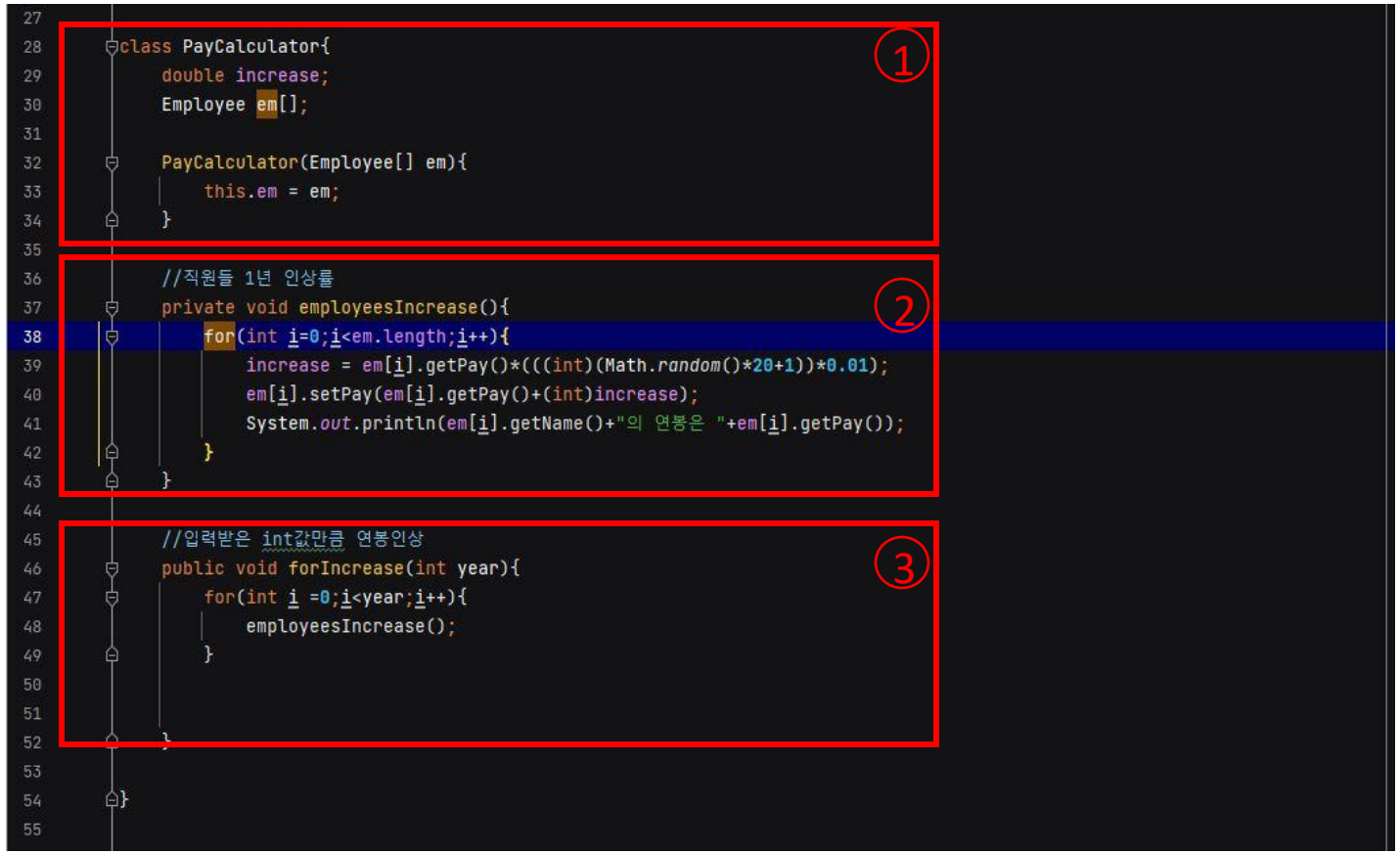
연봉을 인상하는 부분의 메소드를 세분화 할 수 있을 것 같았다. 지금 코드의 작동원리는

1. 연봉인상률(increase)을 만들고 그걸 원래 연봉(pay)에 더한 다음 for문을 인상하는 횟수만큼 돈다
2. 그 다음 직원수(em.length)만큼 for문을 돈다.

세분화한 메소드의 생각은

1. 직원들(em.length)의 1년후 연봉인상금액을 구한다.
2. n값을 집어넣어 n년후의 직원들 연봉인상금액을 구한다.

수업 복습(메소드 세분화)



```
27
28      class PayCalculator{
29          double increase;
30          Employee em[];
31
32          PayCalculator(Employee[] em){
33              this.em = em;
34          }
35
36          //직원들 1년 인상률
37          private void employeesIncrease(){
38              for(int i=0;i<em.length;i++){
39                  increase = em[i].getPay()*(((int)(Math.random()*20+1))*0.01);
40                  em[i].setPay(em[i].getPay()+(int)increase);
41                  System.out.println(em[i].getName()+"의 연봉은 "+em[i].getPay());
42              }
43          }
44
45          //입력받은 int값만큼 연봉인상
46          public void forIncrease(int year){
47              for(int i =0;i<year;i++){
48                  employeesIncrease();
49              }
50
51
52          }
53
54      }
55
```

①

PayCalulator라는 클래스를 만들고 생성자에선 em(Employee 배열)을 this.em에 초기화 해준다.

②

employeeIncrease()라는 메소드를 만들고 직원들의 1년인상된 연봉을 계산하고 보여준다.

③

forIncrease 라는메소드를 만들고 입력값으로 int를 받는다. 받은 int만큼 employeeIncrease()메소드를 실행한다.

수업 복습(메소드 세분화)+추가(평균 구하는 메소드)

```java
public int averagePay(){
    int sum = 0;
    int average = 0;
    for(int i=0;i<em.length;i++){
        sum+=em[i].getPay();
    }
    average= sum / em.length;
    return average;
```

평균 구하는 메소드를 추가했다

1.sum과 average를 지역변수로 두었는데 average같은경우 클래스 객체변수로 두었을때 메소드 연속실행시 초기화가 되지 않아서 지역변수로 설정했다.
2.for문안의 코드는 sum에 em배열의 pay를 더하는 식이다.
3.return값을 두어 average를 이용할수 있게 해봤다.

```java
//입력받은 int값만큼 연봉인상
public void showIncrease(int year){
    for(int i =0;i<year;i++){
        System.out.println(i+"년차 연봉 평균은"+averagePay());
        employeesIncrease();

    }


}
```

원래 메소드였던 forIncrease를 평균또한 보여주기에 shoIncrease라고 이름을 바꿨다.

1.averagePay가 단순히 배열의 평균값을 리턴하는것을 이용해서 for문에 넣어봤다.

```
총 몇명의 직원이 있습니까 ? 3
직원 이름을 입력해주세요 : 직원1
직원1님의 연봉은 3326
직원 이름을 입력해주세요 : 직원2
직원2님의 연봉은 2811
직원 이름을 입력해주세요 : 직원3
직원3님의 연봉은 3152
0년차 연봉 평균은3096
직원1의 연봉은 3558
직원2의 연봉은 3345
직원3의 연봉은 3750
1년차 연봉 평균은3551
직원1의 연봉은 4091
직원2의 연봉은 3478
직원3의 연봉은 3862
2년차 연봉 평균은3810
직원1의 연봉은 4868
직원2의 연봉은 3721
직원3의 연봉은 4364

Process finished with exit code 0
```

정상적으로 작동하는것 같다.

수업 복습(메소드 세분화)

```java
public class TodayTest {
    /*
    랜덤 연봉 적용을 해보자
    어떤 회사에 직원이 10명 있다.
    10명의 이름을 적당히 지어주도록 한다.
    이들의 시작 연봉은 2400~3500으로 랜덤하게 지정한다.
    또한 연봉 인상률은1%~20%사이의 랜덤값을 가지게 한다.
    10년후의 각 직원들의 연봉을 출력하도록 프로그래밍 해보자.

    프로그램실행 ->직원수 입력 -> 직원이름 입력-> 직원정보 생성(객체 배열로 연봉 생성)-> 연봉계산 메소드 실행(객체 배열 입력)->
    -> 들어온 직원 연봉에 일괄적으로 랜덤인상값 부여->
     */
    public static void main(String[] args) {

        System.out.print("총 몇명의 직원이 있습니까 ? ");
        Scanner scan = new Scanner(System.in);
        int num = scan.nextInt();
        //Employee로 이루어진 배열을 입력받은 num만큼 생성
        Employee[] em = new Employee[num];
        //for 문을 돌면서 em배열에 직원정보를 기입
        for(int i=0;i<num;i++){

            Employee e = new Employee();
            em[i] = e;
            System.out.println(em[i]+"님의 연봉은 "+em[i].getPay());
        }

        //연봉를 상승코드
        //em내의 직원들숫자만큼 순회하면서
        for(int i=0;i<em.length;i++){
            //10년치 만큼 연봉인상
            for(int j=0;j<3;j++){

                double increase = (int)em[i].getPay()*(((int)(Math.random()*20+1))*0.01);
                em[i].setPay(em[i].getPay()+(int)increase);
                System.out.println(em[i].getName()+"의 연봉은 "+em[i].getPay());
            }
            System.out.println(em[i].getName()+"의 연봉은 "+(em[i].getPay()));
        }
```

```java
PayCalculator p = new PayCalculator(em);
p.forIncrease( year: 3);
```

생각 한 점

실제 구동 되는 메인함수에서는 5줄정도 되는 코드가 2줄로 줄어들었다. 하지만 전체적인 코드량으로 봤을땐 메소드를 정의하느라 코드가 더 늘어났다. 리소스상 어떤게 유리한지 아직은 잘모르겠다.

복습 후 느낀점 및 질문사항

느낌점.
1.아직 전반적인 지식부족으로 인해 응용력이 많이 떨어진다고 느꼈다. 점점 배우는 내용이 많아지면 차차 나아질거라 생각한다.
2. 메소드를 세분화 하는 정도또한 답이 있는것 같지는 않았다.
매우 세분화가 되면 모든걸 메소드로 할 수도 있을것 같지만 굳이 작업량이 많지 않은코드에선 일반적인 코드작성이 더 편해 보이기도 했다.
이부분은 아직 실제 작업물에선 코드가 어떻게쓰이는지 잘 모르기때문에 좀 더 배워 봐야 할 것 같다.

Q1. 메소드를 세분화를 했을때 실행하는 부분에서 코드는 짧아졌지만 전체적인 코드의 양으로 봤을땐 많아진 느낌이였습니다. 어떤게 리소스?메모리? 부분에서 효율적인 코드인지 아직 가늠이 되지 않는것 같습니다.