6주차 과제 및 실습

과제 및 실습

- **과제는 문제 및 프로그램 실습으로 나뉘어 있습니다.**

- **제출 파일 : 보고서+소스코드**
  - 보고서 : 한글 또는 word 파일
  - 소스코드 : java 파일만 제출
  - 위 두 파일을 학번_이름_실습주차.zip으로 압축하여 제출

  (3개의 실습문제가 있다면 exec_2019111998_1.java, … exec_2019111998_3.java, 2019111998_이선호_2주차.hwp를 zip파일로 압축해서 제출)

- **보고서 내용**
  - 과제 문제 : 해답, 해답의 이유
  - 실습 문제 : 해답소스코드, 프로그램 설명, 결과 화면(결과 캡처)
    - 프로그램 설명 : 작성한 소스코드의 내용, 소스코드 내 주석으로 대체 가능
  - 보고서 파일명 : 학번_이름_실습주차.hwp
    - ex) 2023111010_홍길동_2주차.hwp

- **소스코드**
  - 파일명 : exec_학번_문제번호.java(2번 문제의 경우 exec_2019111998_2.java)

- **제출 : eclass**
  - 과제 연장 제출을 희망할 경우 직접 또는 이메일을 통해 조교에게 요청

프로그램 예제1. 배열을 이용한 객체 생성

```
import java.util.*;

public class Book {
    String title;
    String author;
    public Book() {
        this("", "");
        System.out.println("생성자1 호출됨");
    }

    public Book(String title) {
        this(title, "작자미상");
        System.out.println("생성자2 호출됨");
    }

    public Book(String title, String author) {
        this.title = title; this.author = author;
        System.out.println("생성자3 호출됨");
    }

    void show() { System.out.println("제목 : " + title + ", 저자 : " + author); }

    public static void main(String [] args) {
        Book[] book = new Book[3];
        Scanner scanner = new Scanner(System.in);

        for(int i = 0; i < book.length; i++) {
            System.out.print("서적명 : ");
            String title = scanner.next();
            System.out.print("저자명 : ");
            String author = scanner.next();

            if(title.equals("없음") && author.equals("없음"))
                book[i] = new Book();
            else if(!(title.equals("")) && author.equals("없음"))
                book[i] = new Book(title);
            else if(!title.equals("") && !author.equals(""))
                book[i] = new Book(title, author);
            book[i].show();
        }
        scanner.close();
    }
}
```

```
서적명 : 홍길동
저자명 : 없음
생성자3 호출됨
생성자2 호출됨
제목 : 홍길동, 저자 : 작자미상
서적명 : 없음
저자명 : 없음
생성자3 호출됨
생성자1 호출됨
제목 : , 저자 :
서적명 : 홍길동
저자명 : 허균
생성자3 호출됨
제목 : 홍길동, 저자 : 허균
```

프로그램 예제2. 메소드 오버로딩의 예

Exec2) 이름이 같은 메소드의 사용

메소드의 이름이 같을때는 매개변수의 개수나 자료형이 달라야한다.

(반환형으로 구분하지 않는다.)

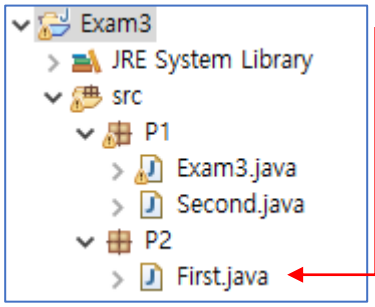
```
class Overloading{
    public int sum(int x, int y) { return x + y; }
    public double sum(double x, double y) { return x + y; }
    public int sum(int x, int y, int z) { return x + y + z; }
}

public class Exam2 {
    public static void main(String[] args) {
        Overloading ol = new Overloading();
        System.out.println("1 + 2의 결과"+ ol.sum(1, 2) );
        System.out.println("1.1 + 2.1의 결과"+ ol.sum(1.1, 2.1) );
        System.out.println("1 + 2 + 3의 결과"+ ol.sum(1, 2, 3) );
    }
}
```

프로그램 예제3. Package와 접근제한자.

Exec3) 패키지를 만들고 클래스를 생성하기.

클래스를 생성할때 패키지를

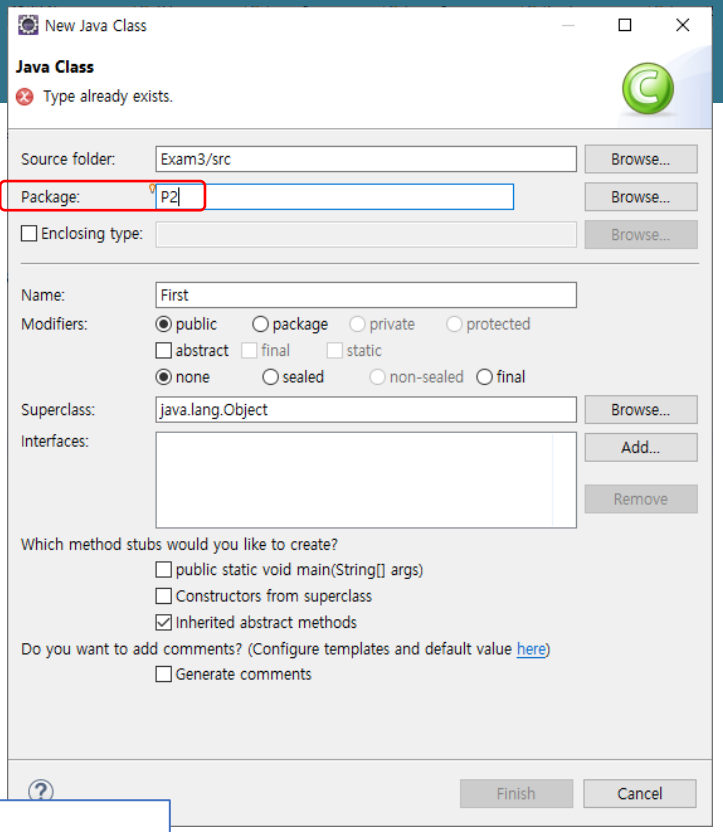



```java
package P2;

import P1.*;

public class First {
    void f() {
        Second b = new Second();
        b.n = 3;
        b.g();
    }
}
```

다른 패키지의 클래스를 사용할 때 import 필수




```java
package P1;

public class Second {
    public int n;
    public int g() {
        return ++n;
    }
}
```

같은 패키지 안에 있음.

```java
package P1;

class Third{
    public void k() {
        Second b = new Second();
        b.n = 7;
        System.out.println(b.g());
    }
}

public class Exam3 {
    public static void main(String[] args) {
        Third t = new Third();
        t.k();
    }
}
```

접근제한자가 없는 디폴트 클래스는 다른 클래스 파일 내에 생성가능하다.

과제문제 : 다음의 문제를 풀고 정답의 이유도 함께 작성하세요.

1. this 키워드의 사용 방법이 아닌 것은?

① 객체 안에서 맴버 변수를 접근할 때 사용한다.

② 생성자를 호출할 때 사용할 수 있다.

③ 생성자의 매개변수가 맴버변수와 동일할 때 구분하기 위해 사용할 수 있다.

④ 생성자의 접근지정자는 public이므로 this로 자유로이 호출한다.

2. 가비지 컬렉션에 대한 설명으로 옳은것은?

① 더 이상 참조되지 않는 객체를 가비지라고 한다.

② 자바에서 참조되지 않는 객체는 운영체제가 판단한다.

③ 가비지의 처리를 위해 사용자가 강제로 가비지 처리 명령을 실행 할 수 있다.

④ 가비지 컬렉터는 가용 메모리 관리를 위해 운영체제에 의해 실행된다.

3. 접근지정자의 설명으로 맞는 것은?

① 객체의 접근지정자는 public과 private를 사용할 수 있다.

② 객체의 맴버에 대한 접근 지정자는 public, private, protected를 사용할 수 있다.

③ 접근지정자를 위한 키워드는 반드시 명시적으로 넣어야 한다.

④ private는 동일한 패키지 안의 다른 클래스에서도 접근할 수 없다.

실습 문제1

exec1) 패키지를 만들어 Rect 클래스와 Exam 클래스를 생성하라.

```java
package w6;

class Rect{
    int x1, y1;
    int x2, y2;

    Rect(){                    // 기본 생성자 호출

    }
    Rect(int x, int y){        // 한 점을 입력받는 생성자 호출

    }
                               // 두 점을 입력받는 생성자 호출
    Rect(int x1, int y1, int x2, int y2){


    }
    private int getWidth() {     // 사각형 폭을 구하는 메소드

    }
    private int getHeight() {    // 사각형 높이를 구하는 메소드

    }
    public int getArea() {       // 사각형 면적을 구하는 메소드

    }
    public int getRound() {      // 사각형 둘레를 구하는 메소드

    }
}

public class Exam1 {
    public static void main(String[] args) {
        Rect r1 = new Rect();
        System.out.println("사각형의 면적은 : " + r1.getArea() + ", 사각형의 둘레는 : " + r1.getRound());
        Rect r2 = new Rect(30, 30);
        System.out.println("사각형의 면적은 : " + r2.getArea() + ", 사각형의 둘레는 : " + r2.getRound());
        Rect r3 = new Rect(10, 10, 50, 50);
        System.out.println("사각형의 면적은 : " + r3.getArea() + ", 사각형의 둘레는 : " + r3.getRound());
    }
}
```

생성자는 점을 없을때는 (0,0) (1,1)을 대각선으로 가지는 사각형, 점이 하나일때는 (0,0), (x,y)를 대각선으로 가지는 사각형, 점이 두개 일때는 (x1, y1), (x2, y2)를 대각선으로 가지는 사각형을 생성한다.

Rect 클래스의 메소드들은 주석에 맞추어 작성한다.

실행했을 때의 보기와 같은 결과가 나오도록 작성하시오.

```
(0, 0), (1, 1)점을 이용한 사각형이 만들어졌습니다.
사각형의 면적은 : 1, 사각형의 둘레는 : 4
(0, 0), (30, 30)점을 이용한 사각형이 만들어졌습니다.
사각형의 면적은 : 900, 사각형의 둘레는 : 120
(10, 10), (50, 50)점을 이용한 사각형이 만들어졌습니다.
사각형의 면적은 : 1600, 사각형의 둘레는 : 160
```

실습 문제2

Exec2) 다음의 코드에서 틀린 부분을 찾아내고 이유를 설명하시오.

1)

```
import java.util.Scanner;
class Sample{
    public int a;
    private int b;
    int c;
}

public class Exec2 {
    public static void main(String[] args) {
        Sample sample = new Sample();
        sample.a = 10;
        sample.b = 10;
        sample.c = 10;
    }
}
```

2)

```
public class Exec2 {
    int x;
    void f(int a)
    {
        x = 1;
    }
    int f(int b)
    {
        return x + b;
    }
}
```

실습 문제3

- 다음과 같은 계산기를 완성하라.

```
import java.util.Scanner;

class Calculator{
    int a, b;
    public void SetValue(int a, int b) {

    }
    public int run(String op) {

    }
}

public class Exec3 {
    public static void main(String[] args) {
        Scanner s = new Scanner(System.in);
        int a, b;
        String op;

        System.out.println("수식을 입력하세요. ex)1 + 2");
        a = s.nextInt();
        op = s.next();
        b = s.nextInt();

        Calculator c = new Calculator();
        c.SetValue(a, b);
        System.out.println(a + " " + op + " " + b + " = " + c.run(op));

        s.close();
    }
}
```

SetValue() : 두개의 피연산자를 객체의 변수 a, b에 저장

run() : 연산자를 받아서 적절한 사칙연산을 수행함.

```
수식을 입력하세요. ex)1 + 2
20 * 3
20 * 3 = 60
```