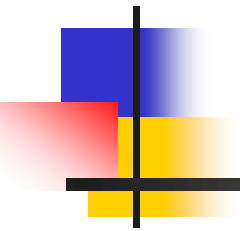

JAVA HomeWork
객체 지향

경복대학교
소프트웨어융합과
배희호 교수

Report 0(읽어보기)

- 트로이와 그리스 군대의 병사들이 전투하는 컴퓨터 프로그램을 절차 지향과 객체 지향 방법으로 만들어보자

Report 0(읽어보기)

- 절차 지향 방법
  - 여러 명의 병사를 절차적 기법으로 일일이 제어하려면 많은 수의 복잡한 함수를 ‘if’나 ‘switch’ 같은 조건을 통해서 호출해야 하기 때문에 논리가 복잡해지는 것을 피할 수 없다
  - 이러한 Algorithm은 병사의 수가 적당한 수준일 때는 상관없지만 그 수가 수천이나 수만에 이르게 되면 현실적으로 넘을 수 없는 한계에 봉착하여 불가능 함
  - ‘객체 지향(Object Oriented)’ 기법은 이러한 한계를 극복할 수 있는 방법론을 제공하기 위해서 고안되었음

Report 0(읽어보기)

- 객체 지향 방법
  - 객체 지향에 익숙한 프로그래머라면 다양한 '조건'을 이용하는 절차적 프로그래밍의 방법과 달리 '칼로 찌르기', '방패로 막기', '달리기', '고함지르기', 혹은 '죽기'처럼 병사가 취할 만한 여러 가지 '동작(action)'을 '메소드(method)'로 구현하고, 키, 체중, 나이와 같이 병사 개인의 모습을 묘사하는 데이터는 '속성(attribute)'에 담아서 'Soldier'라는 이름의 클래스(class)를 정의할 것이다
  - 그리고 전쟁 장면을 연출할 때는 Soldier 클래스의 '인스턴스(instance)'를 무수하게 생성시킴으로써 웅장한 모습을 나타낼 것이다

Report 0(읽어보기)

- 절차 지향
  - 1950년대 중반 고급 언어의 출현과 더불어 시작
  - 동사(함수, 메소드) 중심의 언어
  - FORTRAN, COBOL, Pascal, C
  - Data를 서로 구별하는 방법이 없음
- 객체 지향
  - 1980년대 Software 위기를 극복하는 대안으로 시작
  - 객체, 클래스, 상속 중심의 언어
  - 클래스 내부에 Data(명사)를 배치시키고, 이와 관련된 메소드 (함수)를 배치시키는 방법
  - Data 중심으로 명령과 Data를 결합
  - Java, C++, Smalltalk

Report 0(읽어보기)

- 절차 지향
  - CPU의 행동 (Code) 중심
  - 처리해야할 명령어를 결정하고, 필요한 Data 요청
  - 대규모 Programming 환경에 대응이 미흡
    - 관련 Data나 함수를 분류하는데 문제가 많음
  - 기초적인 Programming 방식을 단순하여 성능이 좋음

- 객체 지향
  - Data (Status) 중심
  - 관리하는 Data를 정의하고, 이에 필요한 명령어를 추가
  - 성능보다는 개발의 편리성과 효율성을 높이는데 목적이 있음

Report 0(읽어보기)

- 객체 지향 Programming 설계
  - Object를 통하여 Data를 분리하여 내부적인 Module화 가능
  - Object와 Message를 중심으로 한 Program 설계
  - Object 별로 Method와 Data를 정의
  - Object 사이의 Message 전송(Method 호출)을 적용
  - Object 중심으로 큰 규모의 Program 제작 가능

- 객체 지향 Programming의 규칙
  - Encapsulation – Object 내부의 Method와 외부 Method를 구별
  - 가능한 Object들 간의 독립적인 형태로 구현
  - Class 재사용 시 독립적 Class의 재사용 유리

Report 0(생각하기)

- House(집)을 객체 지향을 위하여 추상화 해보자

- 명사 : 지붕 색, 창문 수, 주인 이름, 평수, 층수, 건축 년도 등
- 동사 : 청소하다, 수리하다, (집을) 팔다, 사다 등

Report 1(실습 하기)

- 자신만의 Person 클래스를 Class Diagram을 만들어 보자
  - Person은 이름(name)과 나이(age)는 필수로 가진다
  - 이름과 나이 외에 성별 변수(gender)를 추가한다

Report 1(실습 하기)

- Class Member 변수(Field 선언)

```
public class Person {
    private String name;        // String은 문자열 타입
    private int age;
    private String gender;
}
```

Report 1(실습 하기)

- 생성자

```
public Person() {
      this(“ ", 0, “ ");
}

public Person(String name, int age, String gender) {
     this.name = name;
     this.age = age;
     this.gender = gender;
}
```

Report 1(실습 하기)

- setter와 getter 메소드

```
public String getName() {
   return name;
}
public void setName(String name) {
   this.name = name;
}
public String getGender() {
   return sex;
}
public void setGender(String gender) {
   this.gender = gender;
}
public int getAge() {
   return age;
}
public void setAge(int age) {
   this.age = age;
}
```

Report 1(실습 하기)

- 자신만의 Person 클래스를 만들어 보자
  - Main 클래스를 만들고 다음 형식처럼 적용해보자

```
public static void main(String args[]){
        Person man = new Person(“홍길동", "남자", 27);

        man.setName(“홍길동”);
        man.setAge(man.getAge( )  + 1);
        System.out.println(man.getName() + "/" +
                                man.getGender( ) + "/" + man.getAge( ));
}
```

Report 1(실습 하기)

- toString() 메소드를 만들어 .println( ) 테스트 해보자
  - Person 클래스에 다음 코드를 추가하자

```
public String toString() {
   return(/* 이곳에 출력하고 싶은 문자열을 작성 */);
}
```

```
@Override
public String toString() {
    return " [name = " + name + ", age = " + age + ", gender = "
            + gender + "]";
}
```

Report 1(실습 하기)

```
@Override
public String toString() {
    return " [name = " + name + ", age = " + age + ", gender = "
            + gender + "]";
}
```

```
public static void main(String args[]){
        Person man = new Person("함영식", "남자", 27);
        man.setName("함영철");
        man.setAge(man.getAge( )  + 1);
        System.out.println(man.getName() + "/" +
                    man.getGender( ) + "/" + man.getAge( ));
        System.out.println(man);
}
```

Report 2

- int 형의 반지름(radius) Field를 가지는 Circle 클래스를 작성하라
- equals() 메소드를 재정의하여 두 개의 Circle 객체의 반지름(radius)이 같으면 두 Circle 객체가 동일한 것으로 판별하도록 하라
- Circle 클래스의 생성자는 1개의 인자를 가지며 radius Field를 인자로 받아 초기화한다

Report 3

- 다음과 같이 클래스가 정의되어 있다고 가정하자. 이 클래스의 객체를 생성하고 Field를 10과 1.2345로 초기화하며 각 Field의 값을 출력하는 Code를 완성하여라.

```
class Number {
    private int ivalue;
    private float fvalue;
}
```

```
public static void main(String[] args) {
    Number number = new Number(10, 1.2345f);

    System.out.println(number);
}
```

Report 4

- 학생을 나타내는 클래스 Student를 만들어보자. 학생은 이름(name)과 학번(rollno), 나이(age)를 가진다. Student 클래스를 작성하고 객체를 생성하여 테스트하라.

Report 5(학번 짝수)

- 은행 계좌(account)라는 현실 세계의 사물을 데이터적인 측면과 기능적인 측면으로 추상화하여 클래스를 만들어보아라
  - 데이터(명사)
    1) 계좌번호
    2) 비밀번호
    3) 이름
    4) 잔액
  - 기능(동사)
    1) 입금하다
    2) 출금하다
    3) 잔액을 조회하다

Report 5(학번 홀수)

- 은행 계좌(account)라는 현실 세계의 사물을 데이터적인 측면과 기능적인 측면으로 추상화하여 클래스를 만들어보아라
  - 데이터(명사)
    1) 계좌번호
    2) 비밀번호
    3) 이름
    4) 잔액
    5) 이율
  - 기능(동사)
    1) 입급하다
    2) 출금하다
    3) 잔액을 조회하다
    4) 이율을 조정하다

Report 6

- Keyboard부터 특정 연도를 입력 받아 윤년인지 아닌지를 판별하는 Program을 객체 지향으로 작성하라
  - 윤년 판단 기준
    - 연수가 4로 나누어 떨어지는 해는 우선 윤년으로 하되, 그 중에서 100으로 나누어 떨어지는 해는 평년으로 하고, 다만 400으로 나누어 떨어지는 해는 다시 윤년으로 판정한다
  - 프로그램은 2개의 클래스로 작성

Report 제출 방법

- 보고서는 기본적으로 PPT 파일에 작성한다
  - 문제
  - 문제 해결에 필요한 이론적인 내용
  - 소스 파일을 텍스트 형태로 PPT 파일에 복사하여 완성할 것
  - 실행 결과는 실행 결과 화면을 캡쳐하여 PPT 파일에 넣어서 작성할 것
  - 숙제를 한 이후의 느낀 점, 하고싶은 말, 또는 불평~~
- 보고서와 소스/바이트 코드를 하나로 묶은 ZIP 파일을 e-강의실에 업로드 할 것
  - 소스 코드, 바이트 코드 반드시 포함
  - 파일명: X차-홍길동-1401234.zip