12주차 2차시 객체지향의 개념

【학습목표】

1. 객체와 객체지향기법을 설명할 수 있으며, 캡슐화의 개념을 파악할 수 있다.
2. 클래스와 인스탄스 각각의 구성을 파악할 수 있으며, 추상화와 상속의 개념을 설명할 수 있다.

학습내용1 : 객체와 메시지

1. 객체와 메시지

1) 객체(object)
- 실세계에 존재하는 물체(사람, 자동차, 고양이), 추상적 개념(시험 성적, 좌석 예약)처럼 사람이 하나의 단위로 인식할 수 있는 것을 객체로 나타냄.
- 그 상태(state)를 나타내는 「자료부분」,
- 행동(behavior) 양식을 나타내는 「연산부분」으로 구분해 표현 가능함
- 예로서 개(dog)의 경우
    - 자료부분 ⇨이름, 품종, 색깔
    - 연산부분 ⇨멍멍 짖거나, 상대를 공격하거나, 이리저리 뛰는 행위.

- 소프트웨어의 경우 : 「상태」를 「변수」로, 「행동」을「메소드」로 표현하여 「객체는 일련의 변수 값과 메소드의 묶음으로」 생각할 수 있음.

2) 객체지향기법
- 실세계를 객체 단위로 모델링하여 객체모델(object model)을 작성함.
    - 속성(attribute) 객체가 가지는 자료 혹은 특징을 의미함.
    - 프로시저(procedure)·연산(operation) 객체가 수행하는 행위를 의미함.
    - 메소드(method) 객체를 소프트웨어로 구현시킨 연산을 의미함.

3) 메시지(message) 정의된 객체 사이에 정보를 교환하는 수단임.

4) 동일한 용어 ({group }, {thing })에 대한 호칭의 달라짐의 예

분야	Group	Thing
프로그래밍 언어	데이터 타입	변(상)수
객체지향모델링 언어	클래스	객체
데이터베이스와 정보모델링	엔티티 타입	엔티티
실제 예(대학)	교수	정성실

학습내용2 : 캡슐화

1. 캡슐화(encapsulation)

1) 객체를 이용하여 자료와 연산들을 하나의 단위로 묶는 일을 의미함.
- 「개념적 모델」과「대상물」을 일관적으로 취급하여 「프로그램을 모듈화」하면 정확도가 제고됨.
- 다른 모듈·객체의 변화에 대하여 그 종속성이 낮아져 프로그램의 수정이 용이해짐.

2) 캡슐화로 객체의 자료구조와 연산을 수행하는 구체적인 방법
- 사용자가 알지 못하게 하는 정보은폐(information hiding)가 이루어짐

학습내용3 : 클래스와 인스탄스

1. 클래스와 인스탄스

1) 객체들의 유사성을 바탕으로 하여 추상화 계층으로 「클래스(class)」를 정의함.
- 「동일한 자료구조와 연산을 가진 객체집단」을 「클래스(class)」 혹은 「객체유형 (object's type)」 이라고 함.

2) 클래스는 추상화된 속성들을 기술해야 되는데 이러한 속성을 가지는 실객체(real object)를 「클래스의 인스탄스(instance)」라고 함

3) 템플릿(template)은 클래스가 자기에게 속하는 속성을 정의해야 하는데, 이 속성을 정의하는 틀(frame)을 의미함

4) 실질적인 정보처리는 클래스가 담당하지 않고 인스탄스에 의해서 이루어짐. 그러므로 모든 객체는 특정한 하나의 클래스에 소속되어야 함

① 클래스의 구성
- 클래스명 > 군 인
- 속성(자료) > 성명, 성별, 소속, 계급
- 매소드(기능) > 교육신청, 교육받기, 평가결과조회

② 인스탄스의 구성(하나의 예)
- 클래스명 > 군 인
- 속성(자료) > 이기자, 남 자, 2중대, 일등병

③ 매소드(기능) ⇨교육신청, 교육받기, 평가결과조회

2. 다형성(polymorphism)

1) 서로 다른 클래스에 존재하는 객체이지만 유사한 활동을 하는 경우를 의미함
- 연산이 동일하더라도 클래스가 다르기 때문에 전혀 다른 내용을 수행함.
    - 다수의 클래스가 동일한 이름의 자료구조와 연산을 가지더라도, 실제 수행되는 연산은 메시지를 수신하는 객체가 선택하므로 문제가 없음.
    - 메시지를 수신하는 객체는 자신의 클래스에 정의된 연산을 찾아서 필요한 처리를 행함.

학습내용4 : 추상화

1. 분류(classification)

- 객체를 그 구조·속성·기능 등의 특징을 고려하여 해당 클래스로 구분하는 것을 의미함

2. 추상화(abstraction)

- 객체들의 성질을 분해하여 공통된 성질을 추출해서 상위에 존 재 할 상위 클래스(super class)를 만드는 것을 의미

3. 특수화(specialization)

- 상위 클래스에서 하위 클래스를 도출하는 것을 의미함.
- 상위 클래스로부터 다수의 하위 클래스로 구분하는 것을 서브 클래싱(subclassing)이라고 함

학습내용5 : 상속

1. 상속

1) 클래스를 공통·유사한 속성을 중심으로 구분 가능함
- 공통적인 속성에 대해서 추상화된 클래스로 정의하고, 「기존의 유사한 클래스」들이 「새로 정의한 클래스를 공동으로 사용함으로써」 프로그램의 작성이 편해짐.
- 이러한 원리를 상속(inheritance)라고 함.

1) 상속과정
- 상속해 주는 상위의 클래스를 상위 클래스(superclass), 상속을 받는 하위의 클래스를 하위 클래스(subclass)라고 함

2) 프로그래밍 언어에 따라 상속 시에
- 「친자관계를 표현하는 호칭」이 다소 다름
- Smalltalk의 경우 부모를「superclass」, 자식을「subclass」라고 함.
- Eiffel의 경우 부모를「ancestor」, 자식을「descendent」라고 함.
- C++의 경우 부모를『「base class」, 자식을「derived class」라고 함.

3) 객체는 일반화(generalization)와 특수화(specialization) 개념을 이용
- 상위객체와 하위객체를 만들면 상속에 의해서 계층구조(inheritance hierarchy)를 형성함

4) 단일상속(single inheritance)
- 계층구조에서 대부분의 클래스는 상위클래스를 한 개만 가지며, 이로 인해서 상위클래스 하나로부터만 상속을 받게 되는 경우를 의미함

5) 다중상속(multiple inheritance)
- 두 개 이상의 상위클래스로부터 상속을 받는 경우임
예) 겸임교수는 교수속성과 전문가속성을 상속받아야 함

<각 객체는 메시지(message)로 연결됨 그러므로 하위객체가 멀리 위치한 상위객체 에서 상속을 받기 위해서는 상위객체를 찾아 갈 수 있는 포인터(pointer)가 있어야 함>

1) 기존 개발방법론으로는 다양한 요구에 대응 불가하여 소프트웨어 위기(software crisis) 초래 가능성 대두함.

2) 소프트웨어에 대응하기 위한 하나의 방법론이 객체지향기법(object oriented methodology)임.
- 1967년 재사용성(reusability) 제고를 목적으로 Simula언어를 통해서 소개됨.
- 1983년 Smalltalk에 이르러 현 사용 형태의 객체지향 개념이 정립됨.
- 기존 개발방법론은 자료, 처리방법을 분리해서 취급하여 재사용의 제한, 유지보수비용. 시간 과다, 사용자 요구에 대한 효과적 대응 불가 등을 초래함

3) 객체지향방법론의 출발은 객체(object)로부터 시작됨.

학습내용2 : 통합 개발환경의 제기

1. 개요

1) 객체지향방법론만으로 소프트웨어의 모든 문제해결이 불강능하여 통합개발환경 구축이 요구됨.

2) CASE, 정보지장소(information repository), 코드 생성기(codegenerator), 비주얼 프로 그래밍, 객체지향 이터베이스, 추론기관 (inference engine), 비절차적 언어(nonporocedual language), 정보공학(information engineering), 객체지향방법론 등등.

2. 정보저장소

1) 투입된 정보가 머무르는 장소를 의미한다.
- 이것은 정보의 저장에 이용되는 서류철이나 주소록 등으로 비유될 수 있다.
- 정보저장소에는 감각등록기, 작업기억, 장기기억 등의 세 요소가 포함된다.

【학습정리】

1. 객체와 메시지를 알아본다.

2. 캡슐화에 대하여 이해한다.

3. 클래스와 인스탄스를 파악한다.