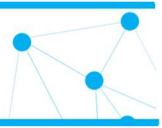
05 프로그래밍 언어와 소프트웨어 개발

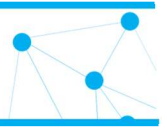
5.1 프로그래밍 언어와 컴파일러
5.2 프로그래밍 언어의 역사와 종류
5.3 프로그래밍 언어의 구조와 컴파일 과정
5.4 절차적 프로그래밍 언어
5.5 객체지향 언어
5.6 소프트웨어 공학과 소프트웨어 개발 방법론
5.7 응용 소프트웨어
5.8 ICT 기술과 관련학문 분야

5.4 절차적 프로그래밍 언어



- 절차적 언어의 개념
- 절차적 언어의 유형

5.4.1 절차적 언어의 개념



- 초창기 프로그래밍 언어 대부분 절차적 언어
- 구조적 프로그래밍 개념
  - 컴퓨터 수행작업은 입력, 연산, 데이터 처리 및 저장, 출력 등 작업의 반복 ⇒ 이 과정을 체계적인 구조로 표현한 것이 구조적 프로그래밍 개념
  - 특징 : 반복 및 제어 구조의 표현을 체계적으로 할 수 있음
    - 반복 구조: for, while 등
    - 선택 구조: if-then-else, switch-case 등

5.4.2 절차적 언어의 유형

- 초창기 절차적 언어: COBOL, FORTRAN
  - FORTRAN : 1954년 과학 및 공학 계산을 위한 최초의 고수준 프로그래밍 언어
  - COBOL : 1960년대 초반 데이터 처리를 주목적

Example of output:

```
PROGRAM output_example
INTEGER :: ix
REAL:: theta
ix=1
theta=3.141593
WRITE(*,*) 'ix =', ix
WRITE(*,*) 'theta = ', theta, 'COS(theta) =', COS(theta)
END PROGRAM output_example
Result printed is: ix=1
                   theta=3.141593 COS(theta) = -1.0
```

(가) FORTRAN 프로그램

```
      .1....:..2....:..3....:....4....:....5....:..6....:...
00010 *
00020 * Module Name :      GetExtensionVersion
00030 *
00031 *   All Rights Reserved, CopyRight(C) FUJITSU LIMITED 1999-20(
00040 *
00060  identification division.
00070  program-id. "GetExtensionVersion".
00080  environment division.
00090  data division.
00100   linkage section.
00110      copy IsapiInf.
00120 *
00130  procedure division with stdcall linkage using ISAPI-INF0.
00140      move 1 to program-status.
00150      exit program.
```

(나) COBOL 프로그램

- C언어
  - 1970년대 초반 벨 연구소에서 UNIX 운영체제를 작성하기 위하여 개발
  - 고수준 프로그램뿐만 아니라 저수준 하드웨어의 제어가 가능하므로 응용프로그램 개발 뿐만 아니라 시스템 프로그램 개발까지 폭넓게 사용

addition of nubmers.c

```c
#include<stdio.h>
int main()
{
    double a, b , result =0 ;
    printf("Enter first number :  ");
    scanf("%lf",&a);

    printf("Enter Second Number :  ");
    scanf("%lf",&b);

    result = a + b;
    printf("Addition of two numbers is :  %lf", result);

    return 0;
}
```

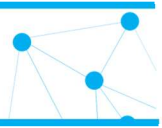
(가) C 언어 프로그램

```
10 INPUT "What is your name: "; U$
20 PRINT "Hello "; U$
30 INPUT "How many stars do you want: "; N
40 S$ = ""
50 FOR I = 1 TO N
60 S$ = S$ + "*"
70 NEXT I
80 PRINT S$
90 INPUT "Do you want more stars? "; A$
100 IF LEN(A$) = 0 THEN GOTO 90
110 A$ = LEFT$(A$, 1)
120 IF A$ = "Y" OR A$ = "y" THEN GOTO 30
130 PRINT "Goodbye "; U$
140 END
```

(나) BASIC 프로그램

Beginner - 초보자용
Python과 함께 자주 등장하는 사용

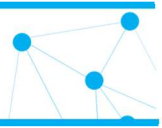
5.5 객체지향 언어



- 객체지향 언어의 개념
- 객체지향 언어의 종류

완전히 바뀐 패러다임

5.5.1 객체지향 언어의 개념



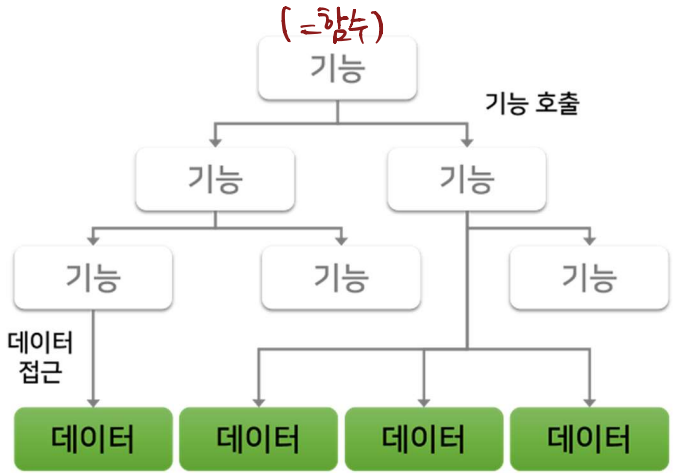
- 절차적 언어의 특징
  - 작업 순서대로 프로그램을 작성
  - 프로그램은 명령어의 목록

모듈화프로그래밍
너하나부터는프로그램의단위 : 처리가초점
메인과서브프로그램으로나뉨

- 객체지향(Object-oriented) 언어는 시스템을 여러 개의 독립된 단위인 객체(Object)들의 모임으로 파악
  - C++, Java, Python 등이 대표적 객체지향 언어

비교

절차형 언어 vs. 객체지향 언어



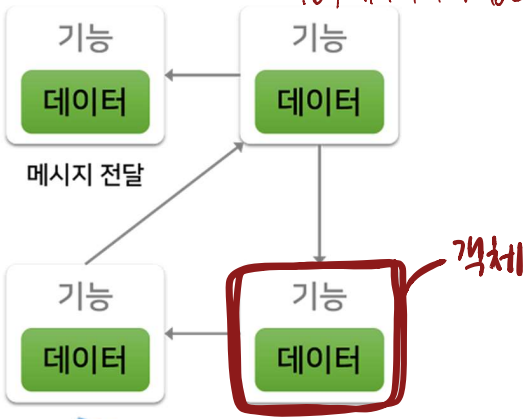
+) 데이터를 수정하면 그 데이터를 사용하는 모든 기능이 영향을 받음 오류발생↑

기능과 데이터의 불일치

빈약한 유지 보수성

시스템을 일일이 수정하게 고칠 수 있다는 뜻.

데이터를 감춰둔다. = 기능과 데이터를 하나로 묶고 기능이 데이터에게 접근하도록 함



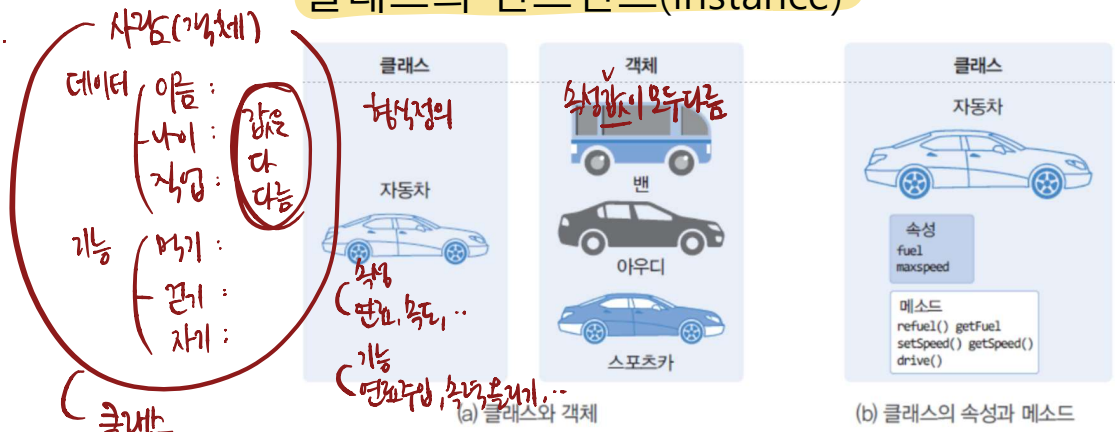
기능과 데이터 캡슐화

개선된 유지 보수성

+ 재사용성 : 객체를 다른 시스템에서도 쓸 수 있음

- 클래스, 객체 및 메소드의 개념

  - 객체 : 인간이 이해하기 쉬운 사물이나 개념을 표현
  - 클래스(Class)
    - 객체의 형식을 의미, 객체의 속성값(데이터)과 그 객체의 처리 방법(메소드)을 하나로 묶어서 정의
  - 객체(Object)
    - 클래스의 인스턴스(Instance)



(a) 클래스와 객체 (b) 클래스의 속성과 메소드

- 객체지향의 특징
  - 추상화, 캡슐화, 상속 및 다형성

1. 추상화(Abstraction)
   - 불필요한 정보는 숨기고 중요한 정보만을 표현함으로써 프로그램을 간단히 만드는 것을 자료추상화라고 함
   - 자료추상화를 통해 정의된 자료형을 추상자료형 (abstract data type)이라고 함
   - 클래스는 추상자료형으로 애플리케이션 프로그램이 꼭 필요로 하는 속성과 메소드만을 추려서 표현

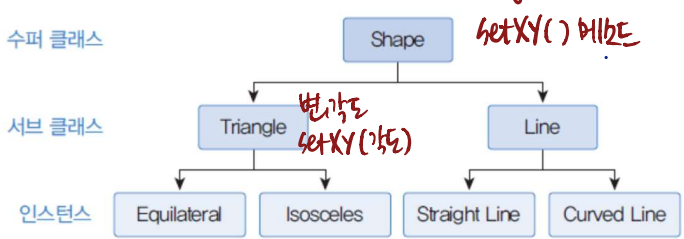
2. 캡슐화(Encapsulation)

- 캡슐화란 속성과 기능을 하나로 묶되, 추상화하여 객체의 세부 내용을 사용자가 보지 못하도록 은폐하는 것
- 객체 안에 속성과 메소드가 존재하고 이들에 대한 접근을 통제
- 객체들 간에 서로 독립적으로 존재하여 전체 프로그램의 구조를 이해하기 쉬워짐 객체끼리 메세지를 주고받는것 파악

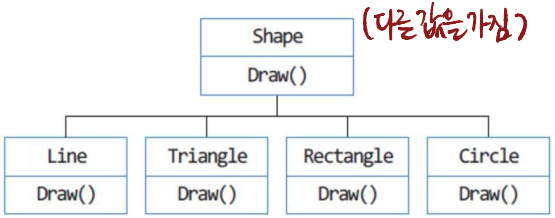
➢ 객체를 정의한다는 말은 객체가 가진 속성, 기능, 그리고 다른 객체와의 관계를 정의하는 것

* 모듈화프로그래밍
: 잘게 서브프로그램으로 나누는것
→ 객체지향에서의 객체 = 모듈

3. 상속(Inheritance)
   - 하나의 클래스가 상위 클래스의 모든 속성 및 메소드를 상속
4. 다형성(Polymorphism)
   - 기본 클래스와 가족으로 가지는 클래스
   - 실제 호출되어 실행되는 객체는 상황에 따라 기본 클래스가 아니고 가족 중 한 객체

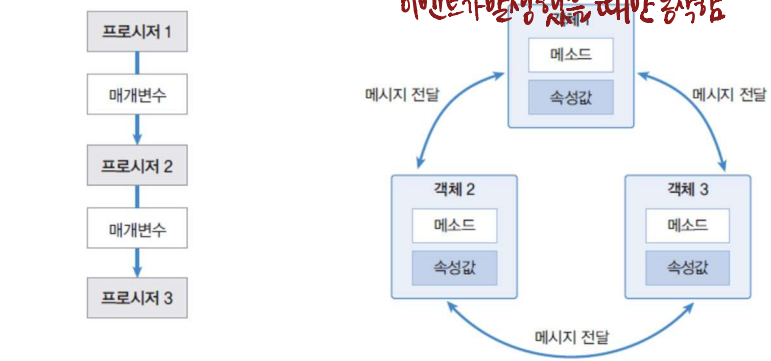


(a) 상속 개념



(b) 다형성 개념

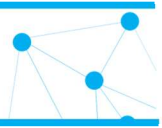
- 절차적 언어와 객체지향 언어의 차이점
  - 절차적 언어는 다른 프로시저를 호출할 때 매개변수 이용
  - 객체지향 언어는 한 객체가 다른 객체에 메시지(Message)를 보냄 ⇒ 이벤트기반(Event Driven) 프로그래밍 기법



(a) 절차적 언어에서의 파라미터 값 전달

(b) 객체지향 언어에서 메시지 전달

5.5.2 객체지향 언어의 종류



- C++, Java, Python
  - C++ : C 언어를 확장하여 객체지향 개념을 적용
  - 자바(Java)
    - 1995년 Sun Microsystems사가 개발
    - Java 컴파일러는 일종의 기계어인 바이트 코드(Byte Code)로 번역, 자바 가상머신(JVM) 에서 실행
  - 파이썬(Python)
    - 1991년 귀도 반 로섬 개발, 비영리재단에서 관리
    - 인터프리터 방식, 동적 타이핑, 작성하기 용이, 라이브러리가 풍부