

## 2장. 추상 자료형

### 학습목표

- 프로그램 설계를 위해 추상 자료형을 정의해야 하는 이유를 이해한다.
- 추상 자료형의 구성요소가 무엇인지 파악한다.
- 객체지향 언어가 추상 자료형을 정의하기에 유리한 이유를 이해한다.
- 정보의 은닉, 인캡슐레이션 등 용어의 의미를 파악한다.

# Section 01 추상 자료형 개념-추상 자료형

## 👤 자료구조보다 더 중요한 개념

## 👤 추상 자료형

- 抽象 資料型
- ADT: Abstract Data Type
- 추상 데이터 타입

## 👤 추상 자료형

- 프로그램의 대상이 되는 사물이나 현상을 추상적으로 정의
- 추상적 정의이기 때문에 일반적인 정의
- 특수한 경우에 모두 사용가능
- 추상적 정의이기 때문에 세부 구현내용을 몰라도 사용가능

## 추상 자료형: 사용과 구현의 분리

### 사용자와 구현자를 분리

- 자료형이 추상적으로 정의되어 있다.
- 구체적인 구현내용은 구현자만 알고 있다.
- 사용자로서는 구현내용을 몰라도 불러다 쓰기만 하면 된다.

# 추상화의 특징

## 👤 단순 자료형

- `Int x, char y;` → 볼트, 너트, 못 단위로 집을 짓기

## 👤 추상 자료형

- 리스트 → 화장실, 침실, 거실 단위로 집을 짓기

## 👤 추상화

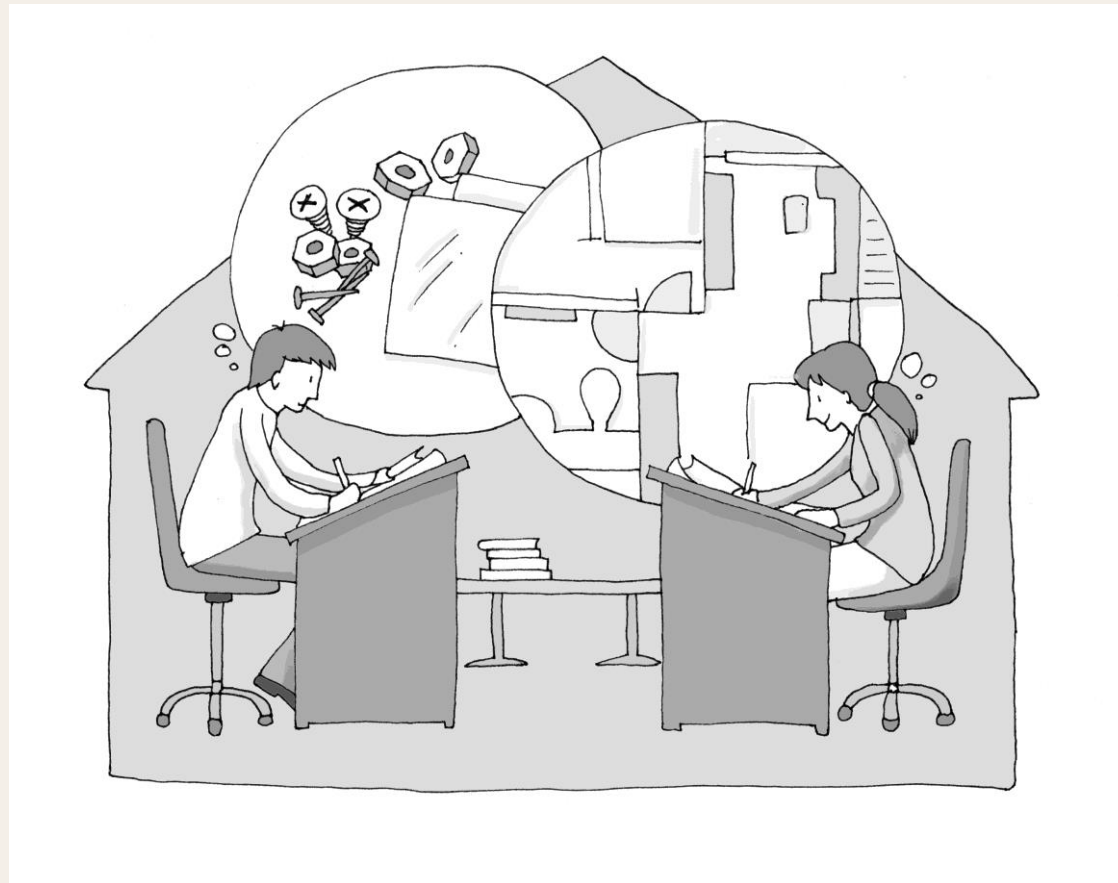
- 단지 추상적으로 그런 것이 있다고 가정하고 집을 설계하라.
- 구현이 가능할지, 누가 구현할지, 언제까지 될지는 고려치 말라.
- 추상화 수준을 높임으로써 프로그램 설계가 용이해 짐

## 👤 일반화

- 일반적으로 필요할 것 같은 작업을 정의
- 특수한 경우의 작업, 구체적 작업은 회피
- 범용성이 높아짐

# 추상화

## 👤 건물을 추상화한 예



## 👤 얼음 제조기의 추상화

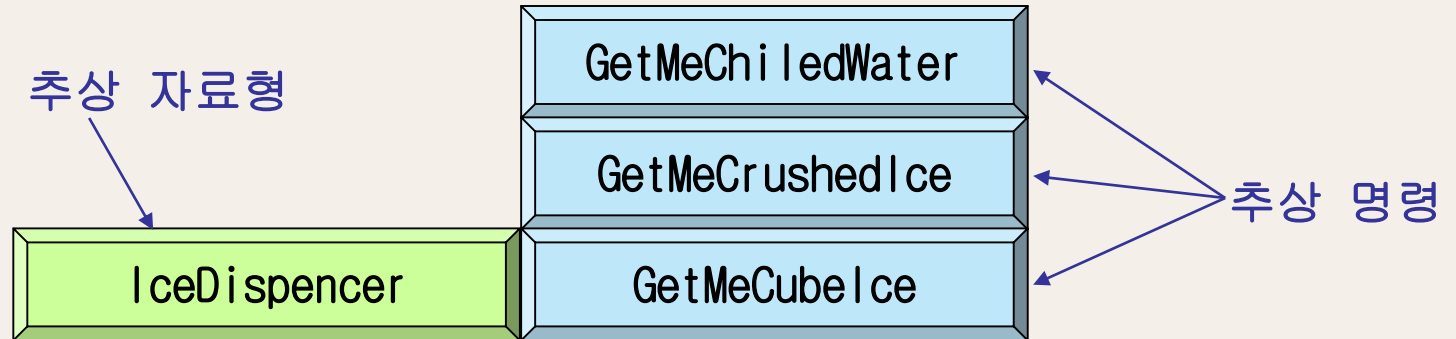
### ADT IceDispenser

- : GetMeChilledWater();      냉수 주시요.
- : GetMeCrushedIce();      부순 얼음 주시요.
- : GetMeCubeIce();      각진 얼음 주시요.



# 추상화

📌 필요한 작업만 명시하면 됨.



[그림 2-3] 추상 자료형 얼음 제조기

# 구현자 관점

👤 작업을 수행하기 위한 구체적인 방법과 위한 데이터 집합

👤 **ADT IceDispenser**

- Data

: Water, Motor, Button 물, 모터, 버튼

- Operation

: GetMeChilledWater() 냉수 주시요의 구체적 구현

{ }

: GetMeCrushedIce() 부순 얼음 주시요의 구체적 구현

{ }

: GetMeCubeIce() 각진 얼음 주시요의 구체적 구현

{ }

👤 자료구조

- 물, 모터, 버튼 등의 데이터 집합
- 데이터는 그 데이터를 가공할 작업을 전제로 설계되어야 함
- 작업이 가장 효율적으로 수행될 수 있도록 조직화된 데이터 집합
- 구조체(Structure)로 가져갈지 아니면 배열(Array)로 가져갈지
- 사용자 입장에서는 자료구조에는 관심이 없음



## 추상 자료형을 보는 관점

### 👤 사용자 관점과 구현자 관점의 추상 자료형

| 사용자 관점의 추상 자료형 | 구현자 관점의 추상 자료형                      |
|----------------|-------------------------------------|
| 작업명과 용도        | 1) 작업의 구체적 구현방법<br>2) 구현을 위한 데이터 집합 |

## Section 02 추상 자료형과 C

### 👤 인터페이스 파일과 구현파일의 분리

- 헤더 파일(.h): 인터페이스 파일, 소스 파일(.c): 구현 파일

### 👤 헤더 파일의 예

typedef struct

|              |            |
|--------------|------------|
| { int Water; | 물의 양       |
| int Motor;   | 모터 회전수     |
| int Button;  | 버튼 1, 2, 3 |

} materialType;

|                            |           |
|----------------------------|-----------|
| void GetMeChilledWater( ); | 냉수 주시요    |
| void GetMeCrushedIce( );   | 부순 얼음 주시요 |
| void GetMeCubeIce( );      | 각진 얼음 주시요 |

### 👤 헤더 파일

- 함수 프로토타입만 보여 줌
- 블랙 박스(정보의 은닉, 구현을 볼 수 없음)
- 계약서 역할(작업의 정의를 자세하고 정확하게 기술)

### 👤 절차적 언어

- 독자적 추상 자료형(Ice Dispencer)을 인정하지 않음
- 따라서 추상 자료형 이름을 선언할 수 없음
- 필요한 작업명과 작업이 가해지는 데이터 타입만을 명시

## Section 03 추상 자료형과 C++

### 개념적 일치

- 사용과 구현의 분리
- 작업을 기준으로 정의된 데이터 타입
- 사용자에게 구현방법을 숨김

### 선언의 일치

- 추상 자료형 선언 = 클래스 선언
- 추상 자료형의 작업 = 객체 클래스의 메시지

# 인캡슐레이션

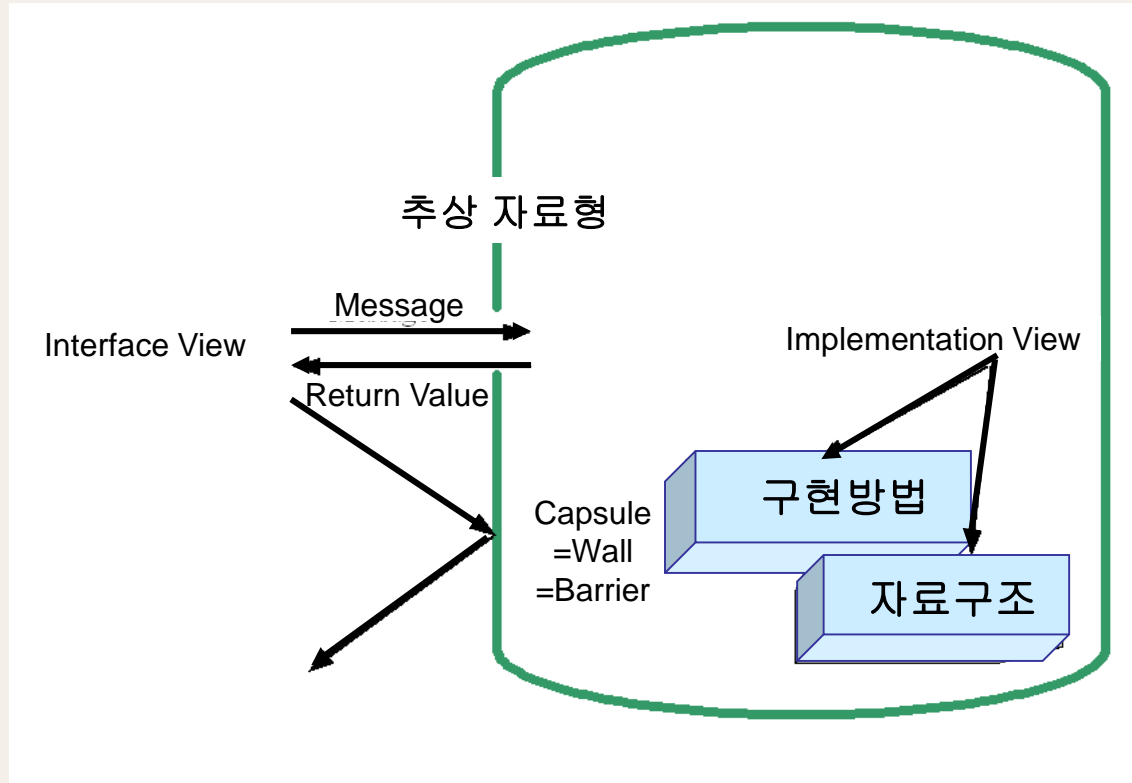
## 👤 첫째 의미

- 캡슐 벽(Wall, Barrier)때문에 밖에서 안을 들여다 볼 수 없음 (정보의 은닉)
- 객체 밖의 사용자가 객체 안의 구현방법이나 자료구조를 알 수 없음
- 캡슐 외부와 내부 사이에 열린 유일한 창구는 메시지

## 👤 둘째 의미

- 하나 이상의 요소를 캡슐이 둘러싸고 있음
- 작업 구현방법과 자료구조를 묶어서 캡슐화
- 작업을 떠나서 자료구조를 생각할 수는 없기 때문에 묶어서 정의

# 인캡슐레이션



[그림 2-4] 정보 은닉과 인캡슐레이션

## 👤 추상 자료형 “얼음 제조기”

```
class IceDispenser           클래스 얼음 제조기
{ public:
    void GetMeChilledWater( ); 냉수 주시요
    void GetMeCrushedIce( );  부순 얼음 주시요
    void GetMeCubeIce( );     각진 얼음 주시요
private:
    int Water, Motor, Button;  메시지 실행을 위한 데이터 집합
}
```

## 👤 멤버 함수는 멤버 데이터의 상태를 바꿔가며 원하는 기능을 수행

- GetMeCrushedIce()는 Water가 얼음으로 바뀌도록 상태 변화를 유도

# 용어 정리

## 👤 객체지향 관련 용어 정리

|          |         |   |   |   |  |
|----------|---------|---|---|---|--|
| 추상 자료형   | ADT     | = | Operation   | + | Collection of Data                     |
| 일반 용어    | Program | = | Algorithm   | + | Data Structure                         |
| 객체지향 방법론 | Class   | = | Message(Interface View) or<br>Method(Implementation View) | + | State Variable or<br>Instance Variable |
| C++      |         |   | Member Function   | + | Member Data                            |

[표 2-2] ADT = 작업 + 데이터 집합



**Thank you**

---