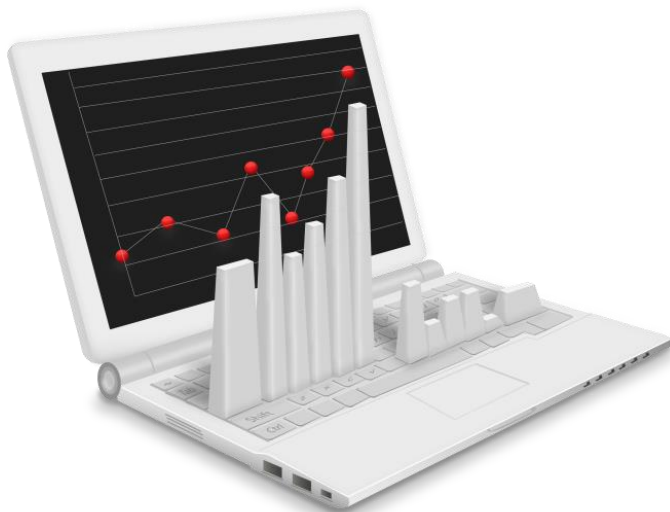


프로그래밍 언어론

구조적 데이터 타입

컴퓨터공학과
조은선



구조적 데이터 타입

학 습 목 표

- 대표적인 구조적 데이터 타입인 Record 타입과 Union 타입에 대해 이해한다.

학 습 내 용

- Record 타입의 개념
- Record 타입의 연산
- Union 타입의 개념
- Union 타입의 종류



목 차

- 들어 가기
- 학습하기
 - Record 타입
 - Union 타입
- 평가하기
- 정리하기



알고가기



다음 중, 한 학생의 정보를 컴퓨터에 나타내고 빠르게 처리하기에 가장 적당하다고 생각되는 타입의 C 변수 선언을 고르시오.

- ① `char x;`
- ② `double y;`
- ③ `char z[100];`
- ④ `struct {
 char name[20];
 char student_id[8];
 char tel_number[11];
} w;`



Record (레코드) 타입

Record (레코드)

- 서로 다른 타입의 이질적인 데이터 원소들의 집단
- 개개의 원소는 이름에 의해서 식별

예 `employee.name, employee.age`

- COBOL에서 도입 대부분의 언어에서 제공
- C, C++, C# 의 struct : 캡슐화된 구조체
- Java 의 class 로 표현 가능

Record 연산

- Assignment, 비교 (같은지 다른지), 초기화 등



| 배열과 Record 타입의 비교

배열과 **record** 타입의 비교

→ record 타입의 원소 타입들은 서로 다름

배열은 서로 동일

→ 원소 접근이 record가 빠름

배열은 주소계산 필요

예) `a[i]`와 `a.id`의 접근 방법 비교

논점

→ record 타입 개체의 필드 참조 구문 형식은?

`OF` (COBOL), `.`(대부분), `%` 등

→ record 타입 개체의 필드 참조시 record 이름 일부가 생략 가능 한가?

`with` (PASCAL)



| Union(공용체) 타입

Union 타입

- 프로그램 실행 중에 다른 시기에 다른 타입의 값을 저장할 수 있는 타입 (메모리 효율 증대)

예

C 의 union

```
union number {
    int value;
    char data[4];
} x;
x.value = 321; ...
strcpy(x.data, "123"); // 앞의 321을 덮어쓰
x.value++               // 의미는?
```

논점

- 타입 검사가 요구되어야 하는가?
- 지원해야 하는가? (Java/C# 없음 – 안전성이 메모리 효율보다 중요)



| Free Union 타입

Free Union(자유 공용체)

→ C/C++의 `union`

→ 타입 검사를 위한 언어적 지원 없음

불완전!

예

```
union T {  
    int intData;  
    char * pointerData;  
} x;  
  
x.intData = 1;  
  
int i;  
x.pointerData = &i;
```



| Discriminated Union

Discriminated Union(판별공용체)

- Ada/Pascal 등
- 타입 지시자 (tag나 discriminant)를 통한 타입 검사

시간/공간
모두
비효율적

예

[태그를 통한 판별 공통체의 예 - PASCAL]

```
type intreal =
  record tagg : Boolean of
    true : (blint : integer);
    false : (blreal : real);
  end;
var blurb : intreal;
blurb.tagg := false;    { real 로 여김 } blurb.blreal := 47.0; { OK }
blurb.tagg := true;     { int 로 여김 }  blurb.blint  := 47;   { OK }
```

A

2

평가하기

마지막으로 내가 얼마나 이해했는지를 한번 확인해 볼까요?
총 2문제가 있습니다.

START



평가하기 1

1. 다음 중 record 타입의 설명으로 틀린 것은?

- ① record 타입의 원소 타입들은 서로 다르나 배열 타입의 원소들은 서로 같다.
- ② 개개의 원소는 index로 식별된다.
- ③ of, ., %, 등을 사용하여 필드를 접근할 수 있다.
- ④ C의 struct 가 이에 해당한다.

확인



평가하기 2

2. Union 타입의 각 종류와 특징과 거리가 먼 것은?

- ① Free Union – 타입검사를 위한 언어적인 지원이 없다.
- ② Discriminated Union – 타입 지시자 (tag나 discriminant)를 통한 타입 검사가 가능하다.
- ③ Free Union– 시간, 메모리 모두 Discriminated Union 에 비해 효율적이다.
- ④ Discriminated Union– C 나 C++ 의 union이 여기에 해당한다.

확인



정리하기

Record(레코드) 타입

Record 타입은 서로 다른 타입의 데이터 원소들의 집단으로서, 구조적인 데이터를 나타내기 적절하므로 프로그램에서 자주 사용된다.

이름으로 접근되며, 원소 타입이나 접근 속도 등 각 특징이 배열과 비교된다.

Union(공용체) 타입

Union 타입은 메모리 효율을 높이기 위해 다른 타입의 값을 하나의 메모리 공간에서 공유할 수 있도록 한다.

신뢰성 측면에서 좋지 않은 것으로 알려져 있고, 특히 Free Union은 타입 검사를 위한 장치가 없어 C#, Java 등에서는 지원되지 않고 있다.