

Basic Computer Programming

Lecture 11

Electrical & Electronics Engineering
Chung-Ang University

Contents

- Understand the struct.
- Understand how to declare and initialize the struct.
- Understand the difference between the declaration of the struct and the declaration of the structural variable.
- The struct can be pointed with a pointer.
- Understand and use the union.
- Create a user-defined data type using typedef.

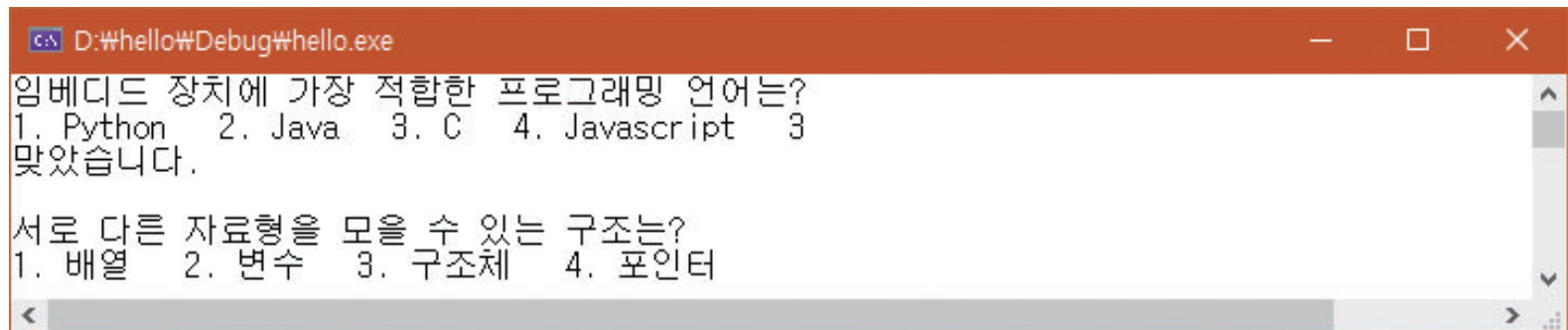
The programs we will make in this chapter

- Express the points as structures and calculate the distance between them.



```
Microsoft Visual Studio 디버그 콘솔
점의 좌표를 입력하시오(x y): 10 10
점의 좌표를 입력하시오(x y): 20 20
두 점사이의 거리는 14.142136입니다.
```

- Write a four-point selection quiz system using the arrangement of the structures.

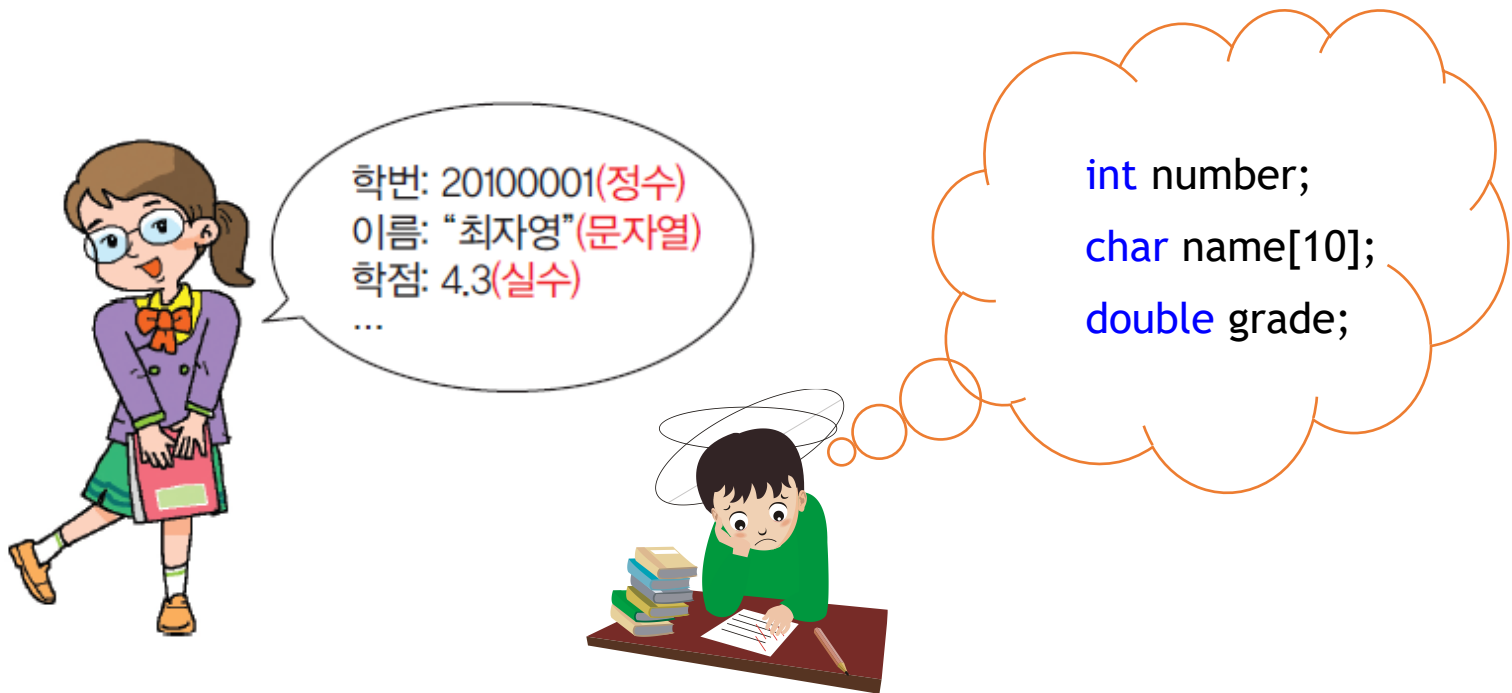


```
D:\hello#Debug#hello.exe
임베디드 장치에 가장 적합한 프로그래밍 언어는?
1. Python 2. Java 3. C 4. Javascript 3
맞았습니다.

서로 다른 자료형을 모을 수 있는 구조는?
1. 배열 2. 변수 3. 구조체 4. 포인터
```

struct

- How do I bring together data about students?
- For example, data about students can be thought of ID numbers, names, and credits, but they are all different types. The student number is Integer type, the name is string, and the credit is double type.



struct

- The above data are not tied together, so it is quite inconvenient to handle. It would be convenient to handle data about a student together. In this case, the structure is used as follows.

```
int number;  
char name[10];  
double grade;
```



```
struct student {  
    int number;  
    char name[10];  
    double grade;  
};
```



Interim check



중간점검

1. 구조체와 배열의 차이점을 이야기해보라.
2. 복소수, 날짜, 화면의 좌표, 사각형 등을 표현하는데 필요한 데이터를 나열해보라.



Declaration of struct

태그(tag)

```
struct student {  
    int    number;    // 학번  
    char   name[10];  // 이름  
    double grade;     // 학점  
};
```

멤버(member)

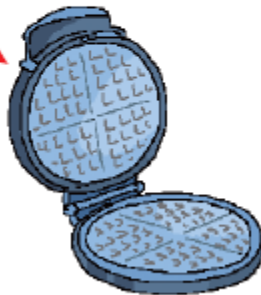


구조체를 사용하면
연관된 변수들을 묶을 수
있습니다.

Declaration of struct

- A structural declaration is not a variable declaration.

구조체를 정의하는 것은 와플이나 붕어빵을 만드는 틀을 정의하는 것과 같다.



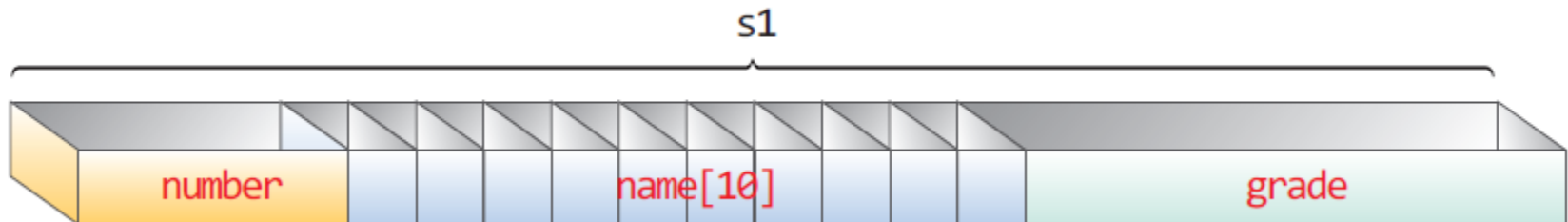
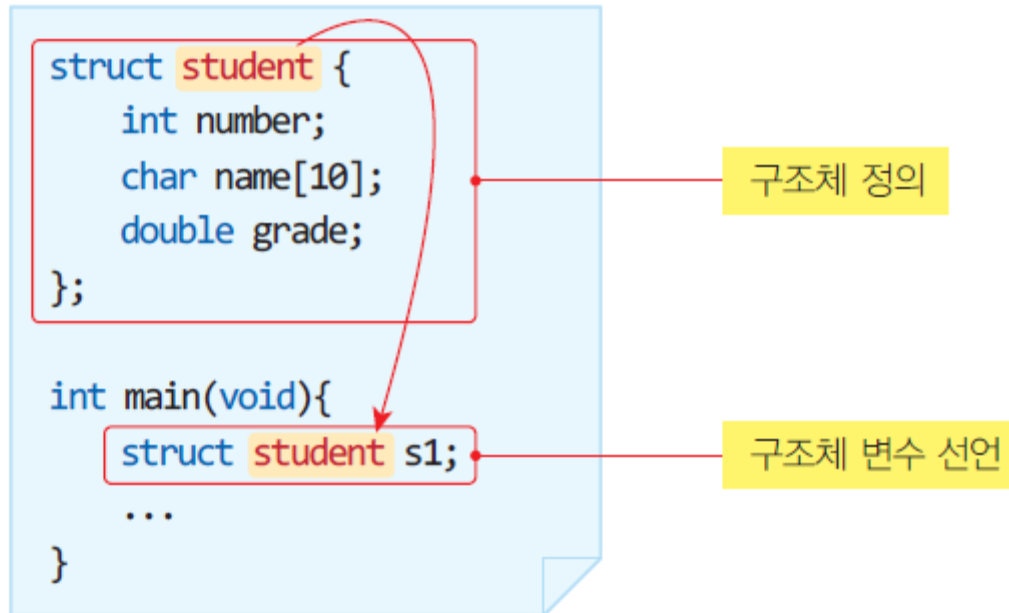
구조체

와플이나 붕어빵을 실제로 만들기 위해서는 구조체 변수를 선언하여야 한다.



구조체 변수

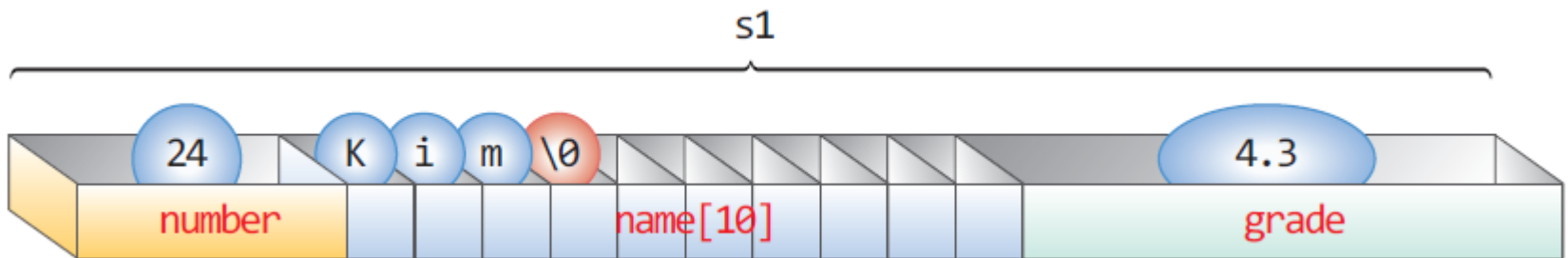
Structural variable declaration



Initialization of variable

- Use brackets to list the initial values.

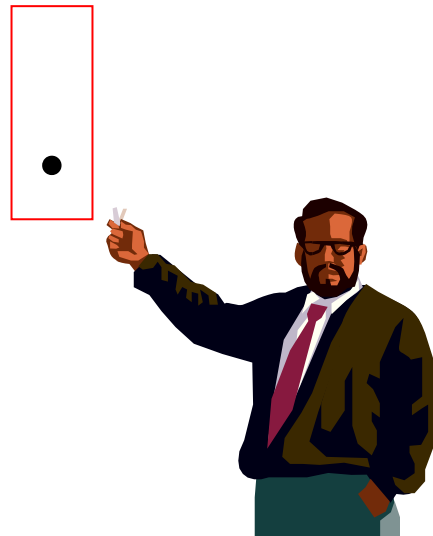
```
struct student {  
    int number;  
    char name[10];  
    double grade;  
};  
struct student s1 = { 24, "Kim", 4.3 };
```



Reference of Structural Members

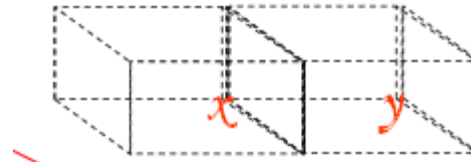
- To refer to structural members: Use the operator '.'

```
s1.number = 20170001;    // 정수 멤버  
strcpy(s1.name, "Kim");  // 문자열 멤버  
s1.grade = 4.3;          // 실수 멤버
```

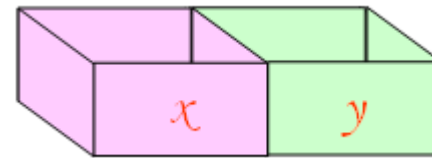


Declaration of struct and declaration of structural variables

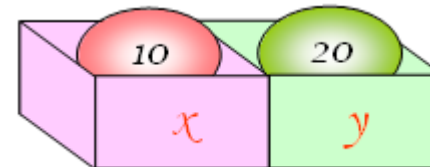
```
struct point {  
    int x;  
    int y;  
};
```



```
struct point p1;
```



```
p1.x = 10;  
p1.y = 20;
```



Example #1

```
#include <stdio.h>
#include <stdlib.h>

struct student {
    int number;
    char name[10];
    double grade;
};
```



학번: 20100001(정수)
이름: "최자영"(문자열)
학점: 4.3(실수)
...

Example #1

```
int main(void)
{
    struct student s;

    s.number = 20170001;
    strcpy(s.name, "홍길동");
    s.grade = 4.3;

    printf("학번: %d\n", s.number);
    printf("이름: %s\n", s.name);
    printf("학점: %f\n", s.grade);
    return 0;
}
```



The screenshot shows the 'Microsoft Visual Studio 디버그 콘솔' (Debug Console) window. It displays the output of the program, which matches the printf statements in the code above. The output is as follows:

```
학번: 20170001
이름: 홍길동
학점: 4.300000
```

Example #2

```
#include <stdio.h>
// 2차원 공간의 점을 구조체로 나타낸다.
struct point {
    int x;
    int y;
};

int main(void)
{
    struct point p = { 1, 2 }; // ①
    struct point q = { .y = 2, .x = 1 }; // ②
    struct point r = p; // ③
    r = (struct point){ 1, 2 }; // ④ C99 버전

    printf("p=(%d, %d) \n", p.x, p.y);
    printf("q=(%d, %d) \n", q.x, q.y);
    printf("r=(%d, %d) \n", r.x, r.y);
    return 0;
}
```

Microsoft Visual Studio 디버그 콘솔

```
p=(1, 2)
q=(1, 2)
r=(1, 2)
```

Interim check



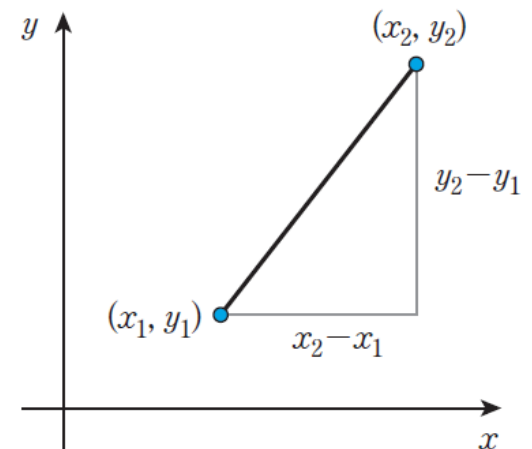
중간점검

1. 구조체 안에 선언된 각각의 변수들을 _____ 이라고 한다.
2. 구조체의 선언에 사용하는 키워드는 _____ 이다.
3. 구조체의 태그는 왜 필요하며, 태그를 사용하는 경우와 사용하지 않은 경우가 어떻게 다른가?
4. 구조체의 선언만으로 변수가 만들어지는가?
5. 구조체의 멤버를 참조하는 연산자는 무엇인가?

Lab: Let's express the point as a struct

- After declaring a struct representing two points, the coordinates of the points are input from the user.
- Let's calculate the distance between these points.

```
Microsoft Visual Studio 디버그 콘솔
점의 좌표를 입력하시오(x y): 10 10
점의 좌표를 입력하시오(x y): 20 20
두 점사이의 거리는 14.142136입니다.
```



Sol:

```
#include <stdio.h>
#include <math.h>

struct point {
    int x;
    int y;
};

int main(void)
{
    struct point p1, p2;

    double xdiff, ydiff;
    double dist;
```

Sol:

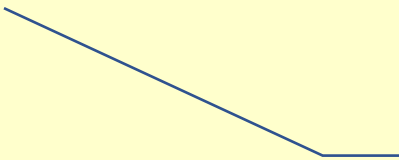
```
printf("점의 좌표를 입력하시오(x y): ");  
scanf("%d %d", &p1.x, &p1.y);  
  
printf("점의 좌표를 입력하시오(x y): ");  
scanf("%d %d", &p2.x, &p2.y);  
  
xdiff = p1.x - p2.x;  
ydiff = p1.y - p2.y;  
  
dist = sqrt(xdiff * xdiff + ydiff * ydiff);  
  
printf("두 점사이의 거리는 %f입니다.\n", dist);  
  
return 0;  
}
```

substitution of structural variables

- Substituting structures is possible and recommended.

```
struct point {  
    int x;  
    int y;  
};  
struct point p1 = { 10, 20 };  
struct point p2 = { 30, 40 };
```

p2 = p1;



Substitution operations between structural variables are possible.

Comparison of Structural Variables

- It is impossible to compare structures.

```
struct point {  
    int x;  
    int y;  
};  
struct point p1 = { 10, 20 };  
struct point p2 = { 30, 40 };  
  
if (p1 == p2)           // 컴파일 오류  
{  
    printf("p1와 p2이 같습니다.")  
}
```

Interim check



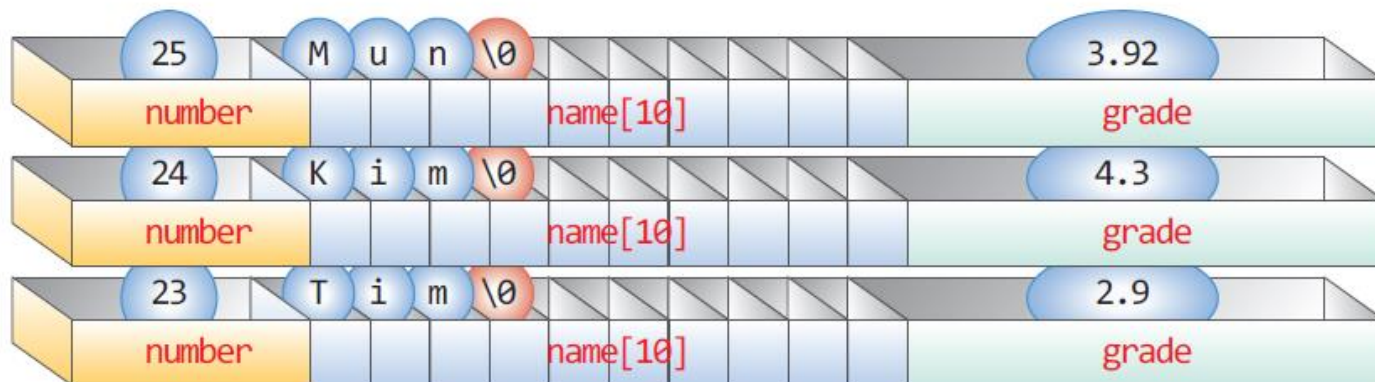
중간점검

1. 구조체의 변수끼리 허용되는 연산에는 어떤 것들이 있는가?
2. 구조체 태그와 구조체 변수의 차이점은 무엇인가?
3. 구조체 멤버로 구조체를 넣을 수 있는가?
4. 구조체는 배열을 멤버로 가질 수 있는가?

Structural array

```
struct student {  
    int number;  
    char name[20];  
    double grade;  
};
```

```
struct student list[100]; // 구조체 배열 선언
```



Structural array

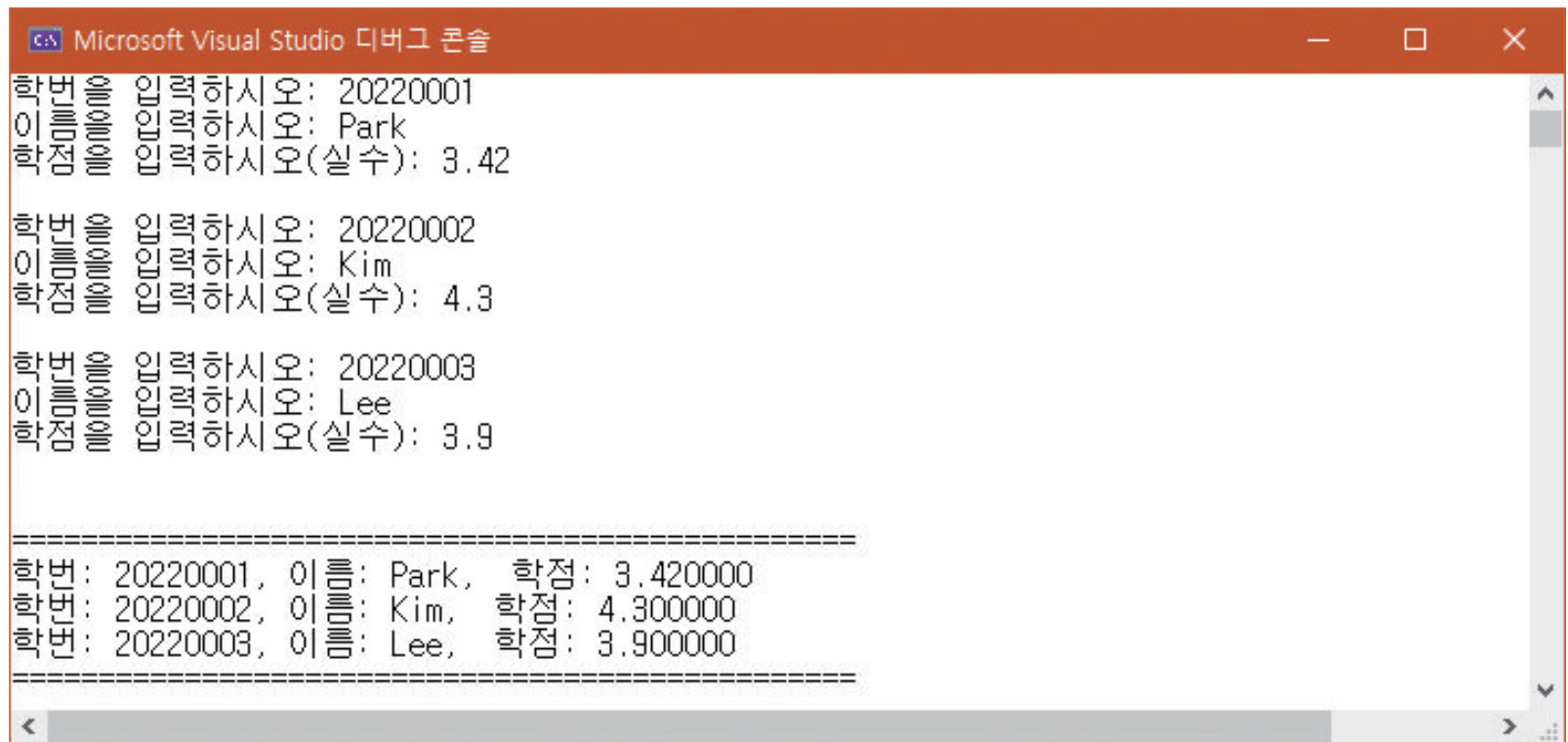
```
struct student {  
    int number;  
    char name[20];  
    double grade;  
};  
  
int main(void)  
{  
    struct student list[100]; // 구조체의 배열 선언  
  
    list[2].number = 27;  
    strcpy(list[2].name, "홍길동");  
    list[2].grade = 178.0;  
}
```


Structural array

```
struct student list[3] = {  
    { 1, "Park", 172.8 },  
    { 2, "Kim", 179.2 },  
    { 3, "Lee", 180.3 }  
};
```

Example #1

- Students' data are input using an iterative structure. The data are stored in an array of structures.



The screenshot shows the Microsoft Visual Studio 디버그 콘솔 (Debug Console) window. It displays the input and output of a program that reads student data iteratively and stores it in an array of structures. The input consists of three sets of student data, each containing a student ID, name, and score. The output shows the data stored in the array, formatted as a table with columns for student ID, name, and score.

```
Microsoft Visual Studio 디버그 콘솔

학번을 입력하시오: 20220001
이름을 입력하시오: Park
학점을 입력하시오(실수): 3.42

학번을 입력하시오: 20220002
이름을 입력하시오: Kim
학점을 입력하시오(실수): 4.3

학번을 입력하시오: 20220003
이름을 입력하시오: Lee
학점을 입력하시오(실수): 3.9

=====
학번: 20220001, 이름: Park, 학점: 3.420000
학번: 20220002, 이름: Kim, 학점: 4.300000
학번: 20220003, 이름: Lee, 학점: 3.900000
=====
```

```
#include <stdio.h>
#define SIZE 3

struct student {
    int number;
    char name[20];
    double grade;
};

int main(void)
{
    struct student list[SIZE];
    int i;

    for (i = 0; i < SIZE; i++)
    {
        printf("학번을 입력하시오: ");
        scanf("%d", &list[i].number);
        printf("이름을 입력하시오: ");
        scanf("%s", list[i].name);
        printf("학점을 입력하시오(실수): ");
        scanf("%lf", &list[i].grade);
        printf("\n");
    }
}
```

Example #1

```
printf("\n=====\\n");  
    for (i = 0; i < SIZE; i++)  
        printf("학번: %d, 이름: %s, 학점: %f\\n", list[i].number, list[i].name,  
list[i].grade);  
    printf("=====\\n");  
    return 0;  
}
```

Interim check



중간점검

1. 상품 5개의 정보를 저장할 수 있는 구조체의 배열을 정의해보라. 상품은 번호와 이름, 가격을 멤버로 가진다.

Lab: Create a Quiz System

- Let's write a program that stores multiple-choice questions, outputs them to the user, receives input from the user, and outputs the correct answer.

```
D:\hello\Debug\hello.exe
임베디드 장치에 가장 적합한 프로그래밍 언어는?
1. Python 2. Java 3. C 4. Javascript 3
맞았습니다.

서로 다른 자료형을 모을 수 있는 구조는?
1. 배열 2. 변수 3. 구조체 4. 포인터
```



Sol:

```
#define _CRT_SECURE_NO_WARNINGS
#include<stdio.h>
#include<stdlib.h>
#define SIZE 100

struct QUESTION {
    char question[SIZE];
    char item1[SIZE];
    char item2[SIZE];
    char item3[SIZE];
    char item4[SIZE];
    int solution;
};

struct QUESTION bank[100] = {
    {"임베디드 장치에 가장 적합한 프로그래밍 언어는?", "1. Python", "2. Java", "3. C", "4. Javascript", 3 },
    {"서로 다른 자료형을 모을 수 있는 구조는?", "1. 배열", "2. 변수", "3. 구조체", "4. 포인터", 3 },
};
```

Sol:

```
int main(void)
{
    int select, i;
    for (i = 0; i < 2; i++) {
        printf("%s\n", bank[i].question);
        printf("%s  ", bank[i].item1);
        printf("%s  ", bank[i].item2);
        printf("%s  ", bank[i].item3);
        printf("%s  ", bank[i].item4);
        scanf("%d", &select);
        if (select == bank[i].solution)
            printf("맞았습니다.\n\n");
        else
            printf("틀렸습니다.\n\n");
    }
    return 0;
}
```


Struct and function

- When a structure is used as an argument or return value, the "call by value" principle applies.

```
int equal(struct student s1, struct student s2)
{
    if (strcmp(s1.name, s2.name) == 0)
        return 1;
    else
        return 0;
}
```

Struct and function

- It can be used as the return value of the function.

```
struct student make_student(void)
{
    struct student s;

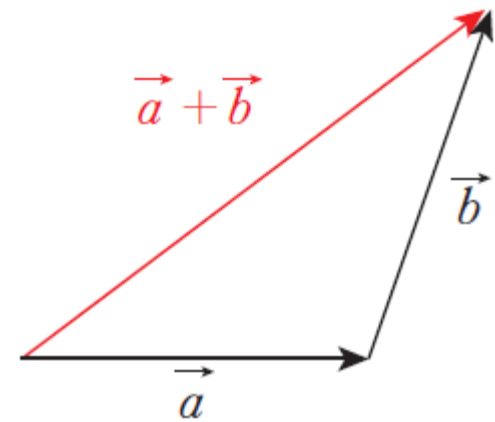
    printf("나이:");
    scanf("%d", &s.age);
    printf("이름:");
    scanf("%s", s.name);
    printf("키:");
    scanf("%f", &s.grade);

    return s;
}
```

Example #1

- Let's produce a function `get_vector_sum()` that finds the sum of the two vectors.

```
Microsoft Visual Studio 디버그 콘솔
벡터의 합은 (7.000000, 9.000000)입니다.
```



Example #1

```
#include <stdio.h>

struct vector {
    float x;
    float y;
};

struct vector get_vector_sum(struct vector a, struct vector b);

int main(void)
{
    struct vector a = { 2.0, 3.0 };
    struct vector b = { 5.0, 6.0 };
    struct vector sum;

    sum = get_vector_sum(a, b);
    printf("벡터의 합은 (%f, %f)입니다.\n", sum.x, sum.y);

    return 0;
}
```

Example #1

```
struct vector get_vector_sum(struct vector a, struct vector b)
{
    struct vector result;

    result.x = a.x + b.x;
    result.y = a.y + b.y;

    return result;
}
```

Interim check



중간점검

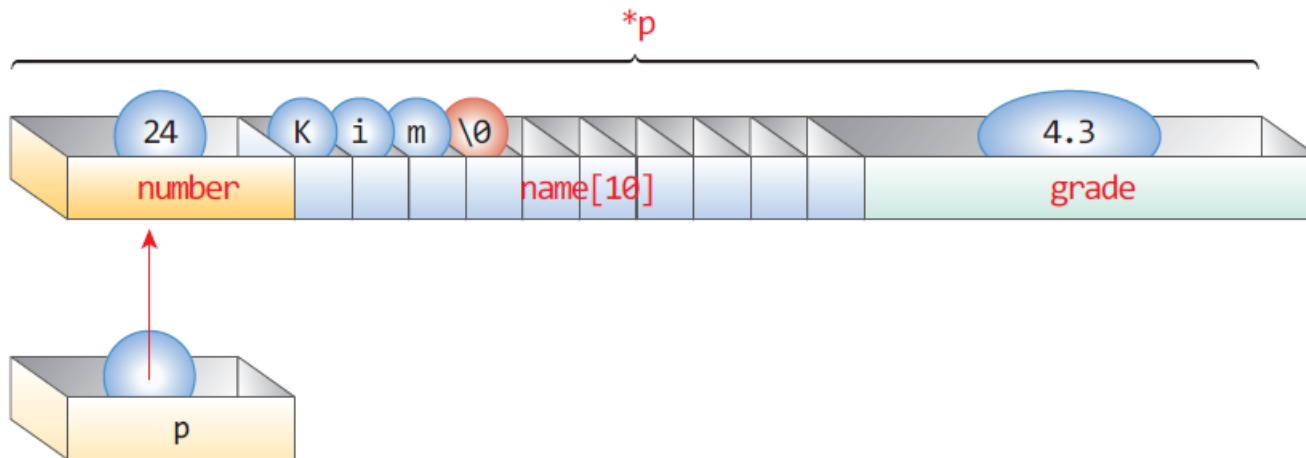
1. 구조체를 함수의 인수로 전달하면 원본이 전달되는가? 아니면 복사본이 전달되는가?
2. 원본 구조체를 포인터로 함수에 전달하는 경우, 원본 구조체를 훼손하지 않게 하려면 어떻게 하면 되는가?

Struct and pointer

```
struct student s = { 20070001, "홍길동", 4.3 };  
struct student* p;
```

```
p = &s;
```

```
printf("학번=%d 이름=%s 학점=%f \n", s.number, s.name, s.grade);  
printf("학번=%d 이름=%s 학점=%f \n", (*p).number, (*p).name, (*p).grade);
```



-> operator

```
p->number;
```

```
printf("학번=%d 이름=%s 학점=%f\n", p->number, p->name, p->grade);
```


Example

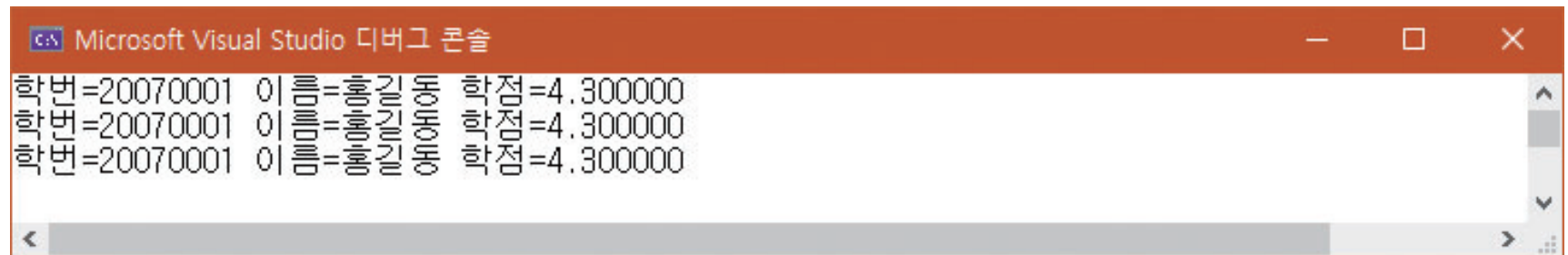
```
#include <stdio.h>

struct student {
    int number;
    char name[20];
    double grade;
};

int main(void)
{
    struct student s = { 20070001, "홍길동", 4.3 };
    struct student* p;

    p = &s;
    printf("학번=%d 이름=%s 학점=%f \n", s.number, s.name, s.grade);
    printf("학번=%d 이름=%s 학점=%f \n", (*p).number, (*p).name, (*p).grade);
    printf("학번=%d 이름=%s 학점=%f \n", p->number, p->name, p->grade);
    return 0;
}
```

results



```
Microsoft Visual Studio 디버그 콘솔
학번=20070001 이름=홍길동 학점=4.300000
학번=20070001 이름=홍길동 학점=4.300000
학번=20070001 이름=홍길동 학점=4.300000
```

Union

- Having multiple variables share the same memory area

```
union example {  
    char c;      // 같은 기억 공간 공유  
    int i;       // 같은 기억 공간 공유  
};
```



Example

```
#include <stdio.h>

union example {
    int i;
    char c;
};

int main(void)
{
    union example data;

    data.c = 'A';
    printf("data.c:%c  data.i:%i\n", data.c, data.i);

    data.i = 10000;
    printf("data.c:%c  data.i:%i\n", data.c, data.i);
}
```



The screenshot shows the Microsoft Visual Studio Debug Console window. The title bar is orange and contains the text "Microsoft Visual Studio 디버그 콘솔". The console area is white and displays the output of the program. The first line shows "data.c:A" and "data.i:-858993599". The second line shows "data.c:+" and "data.i:10000". The console has a scrollbar on the right and a horizontal scrollbar at the bottom.

```
data.c:A  data.i:-858993599
data.c:+  data.i:10000
```

Interim check



중간점검

1. 공용체의 선언에 사용하는 키워드는 _____ 이다.
2. 공용체에 할당되는 메모리의 크기는 어떻게 결정되는가?

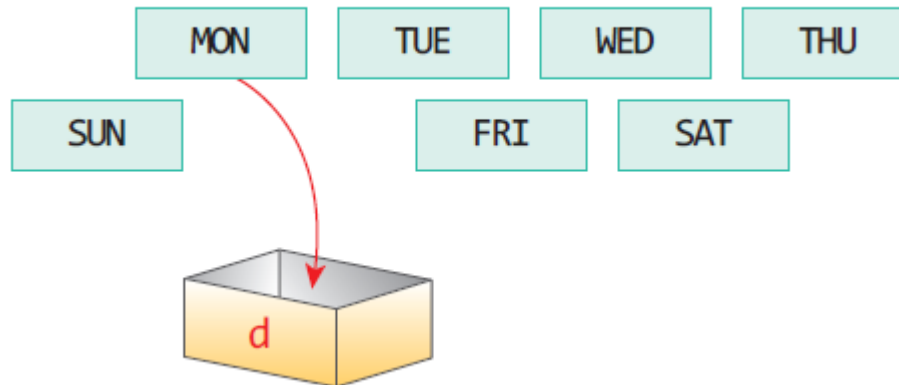
enum

Syntax 11.1 열거형

형식 `enum` 태그 { 값1, 값2, ... };

예 `enum` levels { low, medium, high };

설명 태그라는 이름을 가지는 열거형을 정의한다. 이 열거형은 정의된 값만을 가질 수 있다.



enum initialization

```
enum levels { low = 1, medium, high };
```

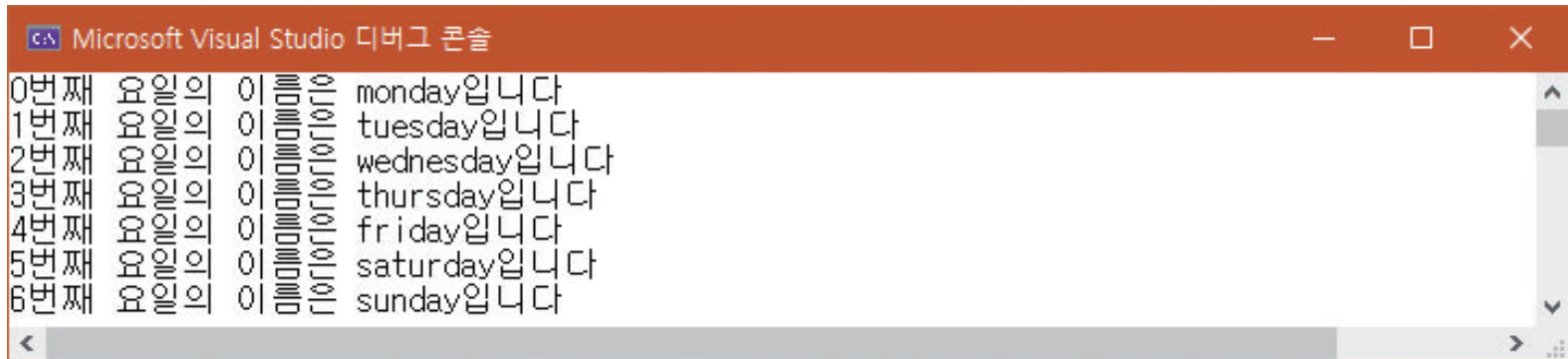
```
enum levels { low = 10, medium = 20, high = 30 };
```

```
enum levels english;
```

```
english = high;
```

Example

- Let's define an enum representing the days of the week and print it out as follows.



```
0번째 요일의 이름이 monday입니다
1번째 요일의 이름이 tuesday입니다
2번째 요일의 이름이 wednesday입니다
3번째 요일의 이름이 thursday입니다
4번째 요일의 이름이 friday입니다
5번째 요일의 이름이 saturday입니다
6번째 요일의 이름이 sunday입니다
```


Sol

```
#include <stdio.h>

enum days { MON, TUE, WED, THU, FRI, SAT, SUN };

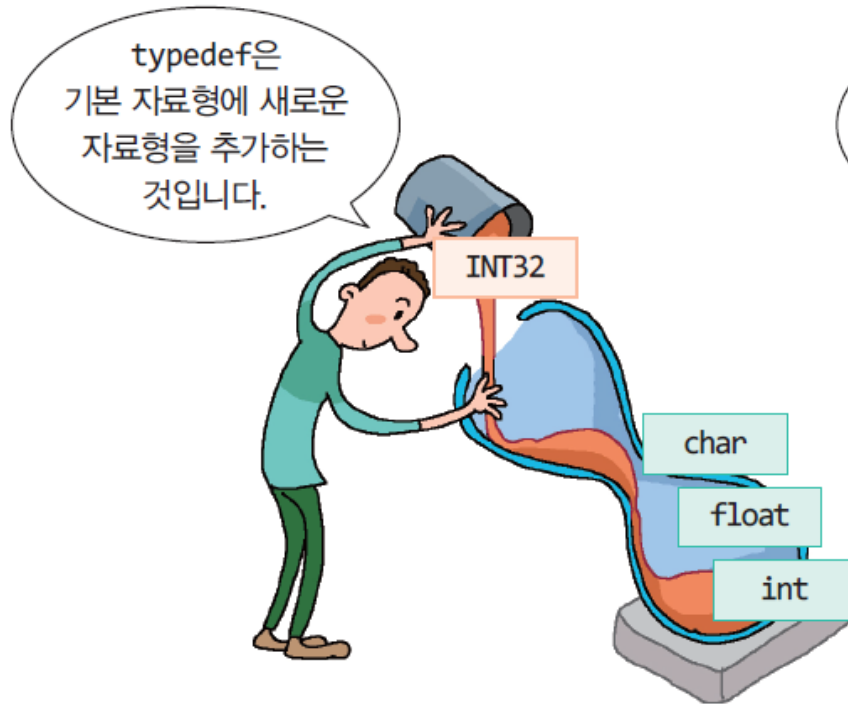
// 포인터들의 배열을 만들고 문자열 상수로 초기화한다.
char* days_name[] = {
    "monday", "tuesday", "wednesday", "thursday", "friday",
    "saturday", "sunday" };

int main(void)
{
    enum days d;

    for (d = MON; d <= SUN; d++)
        printf("%d번째 요일의 이름은 %s입니다\n", d, days_name[d]);
    return 0;
}
```

typedef

- Define a new data type



typedef

Syntax 11.2 typedef

형식 `typedef old_type new_type;`

예 `typedef unsigned char BYTE;`

설명 `old_type`은 기존의 자료형이다. `new_type`은 새롭게 정의하려고 하는 자료형의 이름이다. 위의 문장은 새로운 자료형 `new_type`을 정의하는 것으로 그 내용은 `old_type`과 같다는 의미가 된다.

기존의 자료형

새로운 자료형

Typedef example

```
typedef int  INT32;  
typedef unsigned int  UINT32;  
  
INT32 i; // int i;와 같다.  
UINT32 k; // unsigned int k;와 같다.
```

```
typedef struct point {  
    int x;  
    int y;  
} POINT;
```

Example #1

- The point in the two-dimensional space is expressed as a structure, and then the structure is defined as a new type, POINT, using typedef.



A screenshot of the Microsoft Visual Studio Debug Console window. The title bar is orange and contains the text "Microsoft Visual Studio 디버그 콘솔" along with standard window control buttons. The console area is white and displays the text `(2, 3)+(10, 10)->(12, 13)`. A horizontal scrollbar is visible at the bottom of the console area.

Sol

```
#include <stdio.h>

typedef struct point {
    int x;
    int y;
} POINT;

POINT translate(POINT p, POINT delta);

int main(void)
{
    POINT p = { 2, 3 };
    POINT delta = { 10, 10 };
    POINT result;

    result = translate(p, delta);
    printf("(%d, %d)+(%d, %d)->(%d, %d)\n", p.x, p.y, delta.x, delta.y, result.x,
result.y);

    return 0;
}
```

Sol

```
POINT translate(POINT p, POINT delta)
{
    POINT new_p;

    new_p.x = p.x + delta.x;
    new_p.y = p.y + delta.y;

    return new_p;
}
```

Interim check



중간점검

1. typedef의 용도는 무엇인가?
2. typedef의 장점은 무엇인가?
3. 사원을 나타내는 구조체를 정의하고 이것을 typedef을 사용하여 `employee`라는 새로운 타입으로 정의해보자.