

Object oriented programming In C++ (ACE 1004)

Structure 1

Prof. 최학남

xncui@inha.ac.kr

Office: high-tech 401

Contents

- Definition of **structure**
- Declaration of **structure**
- Initialization of **structure**
- Access the **structure**

What is the Structure

- Arrays require that all elements be of the **same data type**
- Many times it is necessary to group information of different data types.
 - ✓ An example is a materials list for a product. The list typically includes a name for each item, a part number, dimensions, weight, and cost.

.

What is the Structure

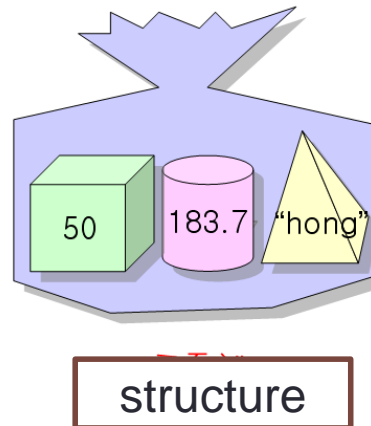
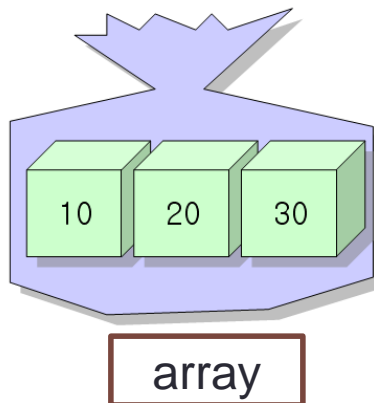
➤ Structure

Data types

fundamental : char, int, float, double etc.

derived : array, structure etc.

- ✓ Can store combinations of **different types of data**
- ✓ A **struct** is a derived data type composed of members that are each **fundamental** or **derived data types**.



Definition of structure

➤ Definition form for structure

```
struct structure_tag_name {  
    data type member_name;  
    data type member_name;  
    ...  
};
```

Create the new data types

The diagram shows a C++ struct definition for a 'student' structure. The code is as follows:

```
struct student {  
    int number; // Student ID  
    char name[10]; // Name  
    double height; // Height  
};
```

Annotations in the diagram:

- A red arrow points from the text "(tag)" to the word "student" in the struct definition, which is highlighted with a pink hexagonal background.
- A red arrow points from the text "(member)" to the variable "height" in the struct definition, which is highlighted with a green hexagonal background.

Definition of structure

```
// point A(x,y)
struct point {
    int x;           // x axis
    int y;           // y axis
};
```

```
// complex value
struct complex {
    double real;      // real
    double imag;      // imaginary
};
```

```
// date
struct date {
    int month;
    int day;
    int year;
};
```

```
// Rectangular
struct rect {
    int x;
    int y;
    int width;
    int height;
};
```

```
// employee
struct employee {
    char name[20];    // name
    int age;           // age
    int gender;        // gender
    int salary;        // salary
};
```

Definition of structure

```
// point A(x,y)
struct point {
    int x;           // x axis
    int y           // y axis
};
```

```
// complex value
struct complex {
    double real;     // real
    double imag;     // imaginary
};
```

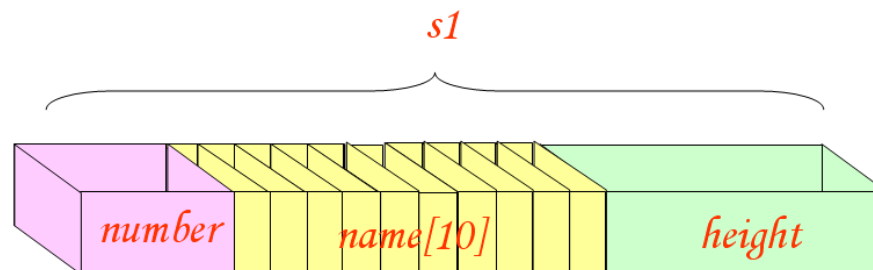
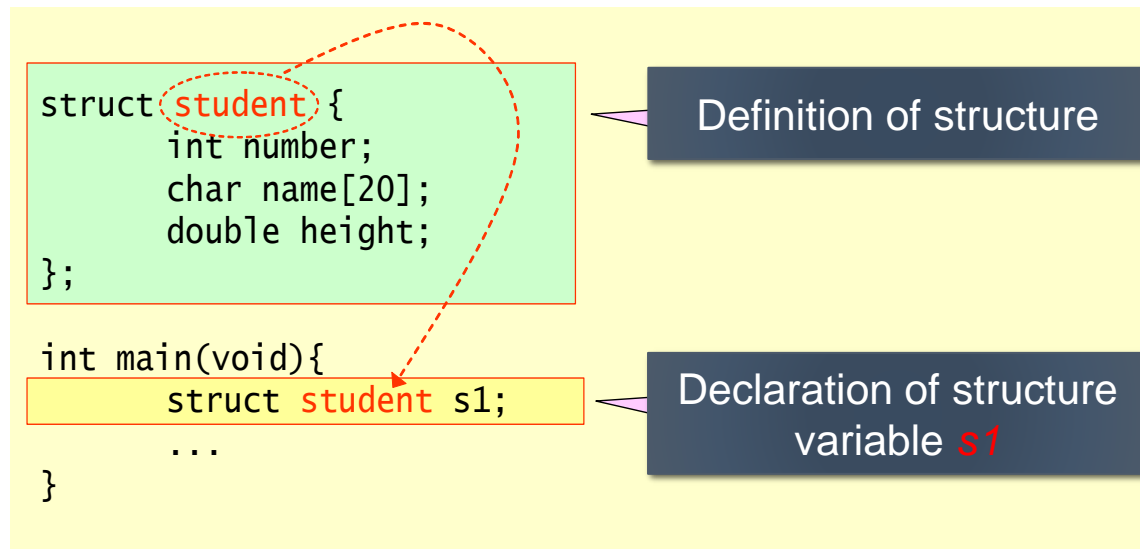
```
// date
union date {
    int month;
    int day;
    int year;
};
```

```
// Rectangular
struct rect {
    int x;
    int y;
    int width;
    int height;
};
```

```
// employee
struct employee {
    char name[20];   // name
    int age;         // age
    int gender;      // gender
    int salary;      // salary
};
```

Declaration of structure

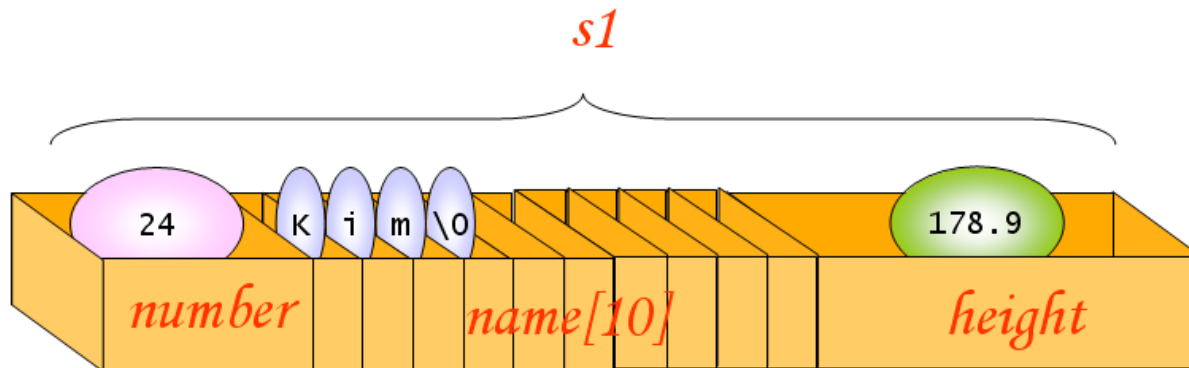
- The definition of structure and the declaration of structure variables are different.



Initialization of structure

```
struct student {  
    int number;  
    char name[10];  
    double height;  
};  
struct student s1 = { 24, "Kim", 178.9 };
```

Using {...}



Access the structure member

- Access the structure member using “.”

```
s1.number = 26;  
s1.name= "Kim";  
s1.height = 183.2;
```

EX #1

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

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

Declaration(definition) of structure

```
int main(void)
```

```
{
```

```
    struct student s;
```

Declaration of structure variable

```
    s.number = 22072174;
    strcpy(s.name, "name");
    s.height = 170.2;
```

Access the structure member

```
    printf("sid: %d\n", s.number);
    printf("name: %s\n", s.name);
    printf("height: %5.1f\n", s.height);
```

```
sid: 22072174
name: name
height: 170.2
```

```
    return 0;
```

```
}
```



EX #2

```
#include<iostream>
struct student {
    int korean;
    int english;
    int math;
};
int main() {
    student x, y;
    x.korean = 80;
    x.english = 90;
    x.math = 70;
    // print the data in x
    printf("%d %d %d\n", x.korean, x.english, x.math);
    // read new data
    scanf_s("%d %d %d", &y.korean, &y.english, &y.math);
    // print the new data
    printf("%d %d %d\n", y.korean, y.english, y.math);
    return 0;
}
```



Structure array

➤ Declaration of structure array

```
struct student {  
    int number;  
    char name[20];  
    double height;  
};  
void main()  
{  
    struct student list[100];           // declaration of structure array  
  
    list[2].number = 27;  
    strcpy(list[2].name, "hong");  
    list[2].height = 178.0;  
}
```

➤ Initialization of structure array

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

EX#3 Structure array

```
#define SIZE 3

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

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

    for(i = 0; i < SIZE; i++)
    {
        printf("Enter the student ID: ");
        scanf("%d", &list[i].number);
        printf("Enter the name: ");
        scanf("%s", list[i].name);
        printf("Enter the height (floating point): ");
        scanf("%lf", &list[i].height);
    }

    for(i = 0; i < SIZE; i++)
        printf("SID: %d, Name: %s, Height: %f\n", list[i].number, list[i].name, list[i].height);

    return 0;
}
```

```
Enter the student ID : 20070001
Enter the name : hong
Enter the height (floating point): 180.2
Enter the student ID : 20070002
Enter the name : kim
Enter the height (floating point): 178.3
Enter the student ID : 20070003
Enter the name : lee
Enter the height (floating point): 176.3
SID: 20070001, Name: hong, Height: 180.200000
SID : 20070002, Name: kim, Height : 178.300000
SID : 20070003, Name: lee, Height : 176.300000
```

Structure and function

- If the input data type is the structure
 - ✓ Pass the structure copy
 - ✓ The larger size of the structure will take much time and memory

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

Return the structure

➤ Call by value

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

    printf("age:");
    scanf("%d", &s.age);
    printf("Name:");
    scanf("%s", s.name);
    printf("Height:");
    scanf("%f", &s.height);

    return s;
}
```

Return the copy of
the structure s

return s;

EX #4

```

#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("Vector Sum is (%f, %f).\n", sum.x, sum.y);

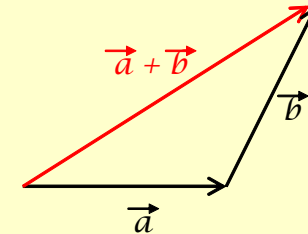
    return 0;
}

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;
}

```



Vector Sum is (7.000000, 9.000000).

저작권 안내 / Copyright Notice

본 사이트에서 수업 자료로 이용되는 저작물은 저작권법 제25조 수업목적저작물 이용 보상금제도의 의거, 한국복제전송저작권협회와 약정을 체결하고 적법하게 이용하고 있습니다. **약정범위를 초과하는 사용은 저작권법에 저촉될 수 있으므로 수업자료의 대중 공개-공유 및 수업 목적외의 사용을 금지합니다. (일부 공개로 제공되는 유튜브 강의를 불법으로 공유하거나 배포하는 행위를 금함)**

The teaching materials provided on this site are copyrighted and signed legally by the Korea Reproduction and Transmission Rights Association. Using these materials beyond the scope of the agreement is a violation against copyright laws. **Hence, one is not allowed to open or share the materials in public. Using them outside of the class is strictly prohibited.**

2020. 3. 11.

인하대학교·한국복제전송저작권협회

Inha University · Korea Reproduction and Transmission Rights Association