



Chapter 08. Structure

목차

1. Understanding the Structure
2. Structure pointers and arrays
3. Unions and enumerations
4. typedef

학습목표

- Use struct defines a new data type.
- It uses a variety of structures.
- The parameters define the structure of the function.
- Using the structure pointer to a struct indirect reference.
- It declares an array of structures to use.
- Learn how to define and use a union.
- Learn how to define and use an enumeration member.
- Learn how to define a typedef.

01 Understanding the Structure

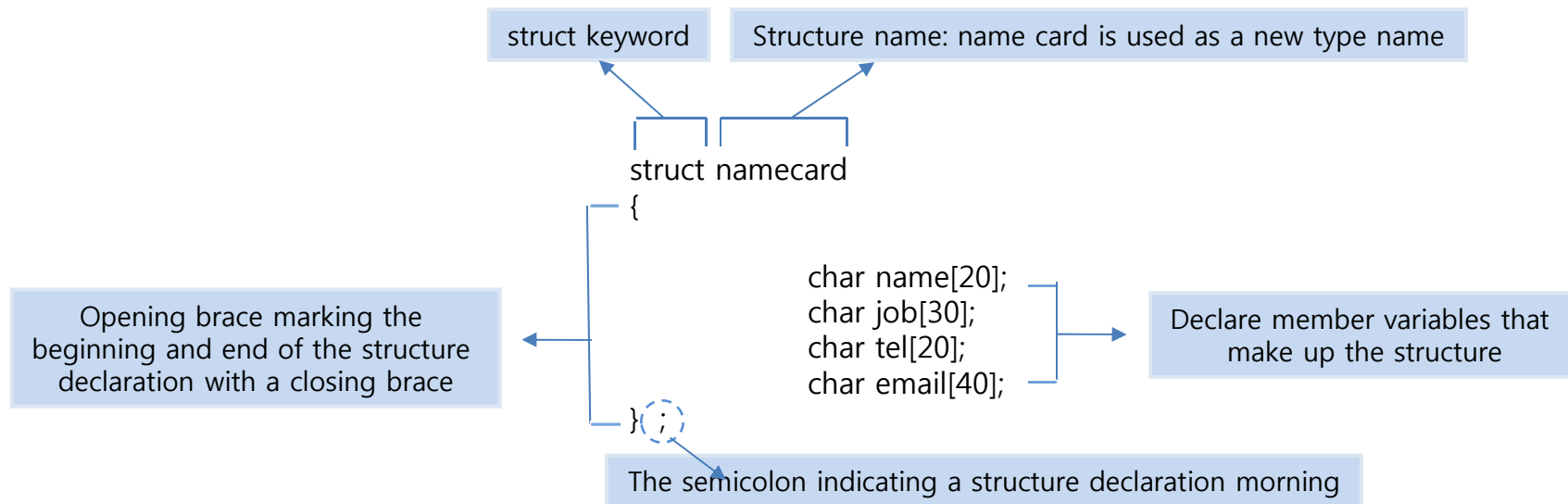
■ Declaration of structure (new types)

```
struct Structure Name{  
    Variable data type members;  
    Variable data type members  
    ;  
    ...  
    ...  
};
```

구조체 선언 기본 형식

- ① Structure name: If you declare several different structure to that reserved words struct, specifying try to separate them.. Use this name when you declare a structure variable structure after the structure declaration.
- ② The structure member variable declaration: struct declaration declares a member variable inside enclosed in braces so configured into multiple members. The structure member variables are the same as the way to declare a variable or a regular array.

01 Understanding the Structure



[Picture 8-1] Card management program structure, for example,

- Struct declaration to mean creating a new data type, data type is the true meaning when used for the purpose of declaring a variable.

struct Name one structure variable structure, people, structure variable name 2, ..., a structure variable name n;

구조체 변수 선언 기본 형식

01 Understanding the Structure

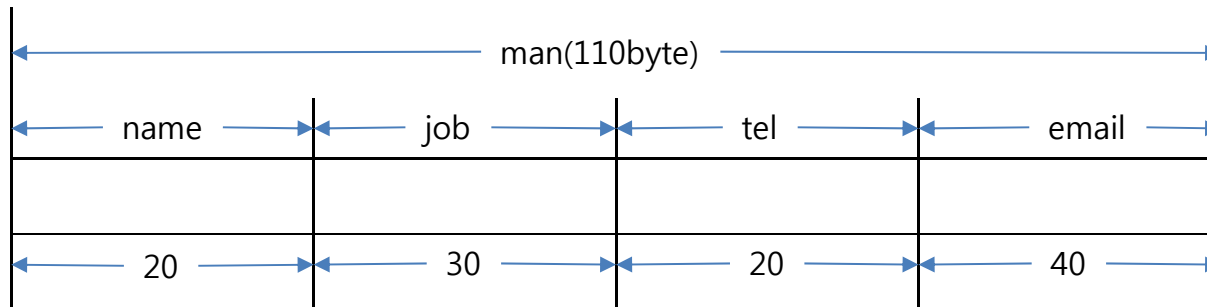
- The technique is typically the head of the file structure declaration (definition). If you declare a variable in the state structure is not the structure declaration "namecard structure that has not been declared" a compile-time error occurs.

struct namecard
man;

The new data types are declared in a structure

This man is caused by a memory allocation name card structure (form) with variable structure

- Let structure variables man expresses a picture of how to allocate memory.



01 Understanding the Structure

```
struct namecard man; // ----- ❶
```

```
namecard man;      // ----- ❷
```

- The way to declare a structure variable is a reserved word struct paste, but also as ❶ ❷ and omit the reserved word struct, as permitted to declare a structure variable structure name only.
- Let's use the variable declaration structures.

```
_ Variable structure member variables;
```

구조체 변수 기본 형식

- Members of the assembly structure member variables to store values in units member variable is because they do not subsequently used as the sole members belong to a specific structure, as in the following example:. You must specify the structure variable name before the operator.

```
man . name
```

Sector structure variable name

Member variable

Example 8-1. To define and use a structure (08_01.cpp)

```
#include <iostream>
using namespace std;

struct namecard {    // 구조체 정의
    char name[20];
    char job[30];
    char tel[20];
    char email[40];
};

void main()
{
    // struct 구조체 변수 man 선언
    //"struct"는 생략가능
    struct namecard man;
```

```
    cout << "이름을 입력하세요: ";
    cin >> man.name;
    cout << "직업을 입력하세요: ";
    cin >> man.job;
    cout << "연락처를 입력하세요: ";
    cin >> man.tel;
    cout << "이메일을 입력하세요: ";
    cin >> man.email;

    cout << "\n\n<입력된 데이터 출력>";
    cout << "\n이름 : " << man.name;
    cout << "\n직업 : " << man.job;
    cout << "\n연락처: " << man.tel;
    cout << "\n이메일: " << man.email;
    cout << "\nsizeof(namecard): " << sizeof(namecard) << "\n";
}
```


01 Understanding the Structure

■ Initialization of a structure variable

- 4 lists the value as an array separated by a comma (,) and binds the whole in braces.

```
namecard x={ "김주현", "MCSE전문강사", "418-9876", "freentour@naver.com"};
```

- While at the same time declaring two or more variables in the General variables, as can the initial value, the structure variables are listed separated a number of initial values for the following variables it can be grouped in braces.

- **General Variable Default Setting Example**

```
int a=10, b=20, c=30;
```

- **Structure Variable Default Setting Example**

```
namecard x={ "김주현", "MCSE전문강사", "418-9876", "freentour@naver.com"},  
          y={ "박혜경", "웹마스터", "551-6986", "hk@naver.com"},  
          z={ "김동식", "기획A팀대리", "318-3961", "ds@naver.com"};
```

Example 8-2. To initialize structure (08_02.cpp)

```

01 #include <iostream>
02 using namespace std;
03 struct namecard{
04     char name[20];
05     char job[30];
06     char tel[20];
07     char email[40];
08 };
09 void main()
10 {
11     namecard x={ "김주현", "MCSE전문강사", "418-9876", "freentour@naver.com"};
12     namecard y={ "박혜경", "웹마스터", "551-6986", "hk@naver.com"};
13     namecard z={ "김동식", "기획A팀대리", "318-3961", "ds@naver.com"};
14
15     cout<<" 이름 Wt 직업 WtWt 연락처 Wt 이메일 ";
16     cout<<"\n =====";
17     cout<<"\n "<<x.name <<"Wt"<< x.job <<"Wt"<< x.tel <<"Wt"<< x.email;
18     cout<<"\n "<<y.name <<"Wt"<< y.job <<"Wt"<< y.tel <<"Wt"<< y.email;
19     cout<<"\n "<<z.name <<"Wt"<< z.job <<"Wt"<< z.tel <<"Wt"<< z.email;
20     cout<<"\n =====\n";
21 }

```



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The window displays a table of employee information with four columns: 이름 (Name), 직업 (Job), 연락처 (Contact), and 이메일 (Email). The table is separated by lines of equals signs. The data rows are as follows:

이름	직업	연락처	이메일
김주현	MCSE전문강사	418-9876	freetour@naver.com
박혜경	웹마스터	551-6986	hk@naver.com
김동식	기획A팀대리	318-3961	ds@naver.com

01 Understanding the Structure

■ An assignment to construct a unit value

- When you want to store the value stored in the variable structure in other structure variables can be expressed as follows You can copy the values to the structure unit for a variable declared with the same structure as the assignment operator. If this is not possible if individually member variable, that would need to copy the value of a field-by-field basis.

```
namecard x={ "김주현", "MCSE전문강사", "418-9876", "freentour@naver.com"};
```

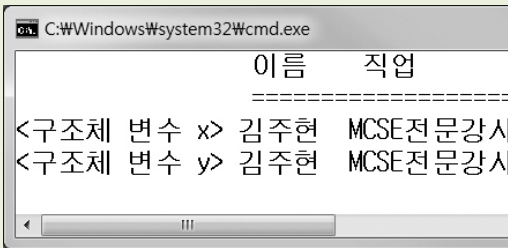
```
namecard y;
```

```
y = x;
```

```
=  
strcpy( y.name , x.name);  
strcpy( y.job , x.job );  
strcpy( y.tel , x.tel );  
strcpy( y.email , x.email);
```

Example 8-3. To copy the value of the structure unit (08_03.cpp)

```
01 #include <iostream>
02 using namespace std;
03 struct namecard{
04     char name[20];
05     char job[30];
06     char tel[20];
07     char email[40];
08 };
09 void main()
10 {
11     namecard x={ "김주현", "MCSE전문강사", "418-9876", "freentour@naver.com"};
12     namecard y;
13
14     y = x;
15
16     cout<<"\t\t 이름 \t\t 직업 \t\t\t 연락처 \t\t 이메일 ";
17     cout<<"\n\t\t\t =====";
18     cout<<"\n<구조체 변수 x>";
19     cout<<"\t"<<x.name <<"\t"<<x.job <<"\t"<<x.tel <<"\t"<<x.email;
20     cout<<"\n<구조체 변수 y>";
21     cout<<"\t"<<y.name<<"\t"<<y.job<<"\t"<<y.tel<<"\t"<<y.email<<"\n";
22 }
```



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The output of the program is displayed as follows:

```
이름      직업
=====
<구조체 변수 x> 김주현 MCSE전문강사
<구조체 변수 y> 김주현 MCSE전문강사
```



The screenshot shows a Windows command prompt window with the title bar "C:\Windows\system32\cmd.exe". The command prompt displays a table of employee information. The table has four columns: "이름" (Name), "직업" (Job), "연락처" (Contact), and "이메일" (Email). The table is separated by a dashed line. The data rows are as follows:

	이름	직업	연락처	이메일
<구조체 변수 x>	김주현	MCSE전문강사	418-9876	freentour@naver.com
<구조체 변수 y>	김주현	MCSE전문강사	418-9876	freentour@naver.com

01 Understanding the Structure

■ Structure used as a parameter of the function

- Structure may be a parameter in the function.

```
struct namecard x, y, z;
```

```
structPrn(x);           // The structure variable x is passed to the function as a parameter
structPrn(y);
structPrn(z);
```

- StructPrn formal parameters of function variables temp and stores handed over a copy of the value of the parameter in the call room. Type parameters to copy the temp is the value of y and z values of x the value of x by passing the function call to pass the y, z to pass the output unit to access the member variables.

```
void structPrn(namecard temp) // The definition of a function
{
    cout<<"\n"<<temp.name<<"\t"<<temp.job<<"\t"<<temp.tel<<"\t"<<temp.email;
}
```

Example 8-4. Function (08_04.cpp) to use the structure as a transmission method according to the value as a parameter

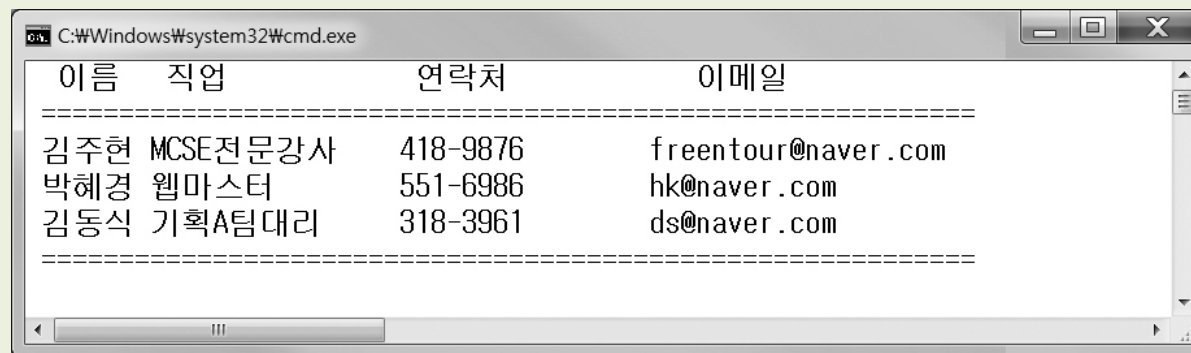
```
#include <iostream>
using namespace std;

struct namecard{
    char name[20];
    char job[30];
    char tel[20];
    char email[40];
};

void structPrn(namecard temp)
{
    cout << "Wn 구조체 : " << temp.name << "Wt" << temp.job << "Wt" << temp.tel << "Wt" << temp.email;
}
```

Example 8-4. Function (08_04.cpp) to use the structure as a transmission method according to the value as a parameter

```
void main() {  
    namecard x = { "김주현", "MCSE전문강사", "418-9876", "freentour@naver.com" };  
    namecard y = { "박혜경", "웹마스터", "551-1234", "hk@naver.com" };  
    namecard z = { "김동식", "기획팀대리", "318-4321", "ds@naver.com" };  
  
    cout << "₩t   이름 ₩t ₩t 직업 ₩t₩t 연락처 ₩t ₩t 이메일 ";  
    cout << "₩n=====";  
    structPrn(x);    // 구조체 매개변수 x  
    structPrn(y);    // 구조체 매개변수 y  
    structPrn(z);    // 구조체 매개변수 z  
    cout << "₩n=====₩n";  
}
```



```
C:\Windows\system32\cmd.exe  
이름   직업   연락처   이메일  
=====
```

김주현	MCSE전문강사	418-9876	freentour@naver.com
박혜경	웹마스터	551-6986	hk@naver.com
김동식	기획팀대리	318-3961	ds@naver.com

```
=====
```

01 Understanding the Structure

■ Function having the return value to the structure

- Try to save the value received from the keyboard to create a program structure variables were as follows:

```
11  namecard man;  
12  
13  cout<<" 이름을 입력하세요=>";  
14  cin>>man.name;  
15  cout<<" 직업을 입력하세요=>";  
16  cin>>man.job;  
17  cout<<" 연락처를 입력하세요=>";  
18  cin>>man.tel;  
19  cout<<" 이메일을 입력하세요=>";  
20  cin>>man.email;
```

- Let's create a section to enter the structure as a function value.

01 Understanding the Structure

- When implementing the structure part of the input value to a function, the function is called to return the received value to the calling function. Here is a function for storing the values entered from the keyboard on the structure.

▪

```
void main()
```

```
{
```

```
    namecard a;
```

```
    a = structInput();
```

```
}
```

```
namecard structInput()
```

```
{
```

```
    namecard temp;
```

```
    cout<<" 이름을 입력하세요=>";
```

```
    cin>>temp.name;
```

```
    cout<<" 직업을 입력하세요=>";
```

```
    cin>>temp.job;
```

```
    cout<<" 연락처를 입력하세요=>";
```

```
    cin>>temp.tel;
```

```
    cout<<" 이메일을 입력하세요=>";
```

```
    cin>>temp.email;
```

```
    return temp;
```

```
}
```

Example 8-5. Create a function that returns a structure (08_05.cpp)

```
#include <iostream>
using namespace std;
struct namecard {
    char name[20];
    char job[30];
    char tel[20];
    char email[40];
};
void structPrn(namecard temp) {
    cout << "\n 구조체 : " << temp.name << "\t" << temp.job << "\t" << temp.tel << "\t" << temp.email;
}
namecard structInput() {    // 구조체 반환값을 갖는 함수
    namecard temp;
    cout << " 이름을 입력하세요 :";
    cin >> temp.name;
    cout << " 직업을 입력하세요 :";
    cin >> temp.job;
    cout << " 연락처를 입력하세요 :";
    cin >> temp.tel;
    cout << " 이메일을 입력하세요 :";
    cin >> temp.email;
    cout << "\n";
    return temp;           // 구조체 반환
}
```

Example 8-5. Create a function that returns a structure (08_05.cpp)

```
void main()
{
    namecard x, y, z;

    x = structInput(); // 구조체 반환값을 갖는 함수 호출
    y = structInput();
    z = structInput();

    cout << "Wt   이름 Wt 직업 WtWt 연락처 Wt 이메일 ";
    cout << "Wn===== ";

    structPrn(x);
    structPrn(y);
    structPrn(z);

    cout << "Wn===== Wn";
}
```



C:\Windows\system32\cmd.exe

이름을 입력하세요=>김주현
직업을 입력하세요=>MCSE전문강사
연락처를 입력하세요=>418-9876
이메일을 입력하세요=>freentour@naver.com

이름을 입력하세요=>박혜경
직업을 입력하세요=>웹마스터
연락처를 입력하세요=>551-6986
이메일을 입력하세요=>hk@naver.com

이름을 입력하세요=>김동식
직업을 입력하세요=>기획A팀대리
연락처를 입력하세요=>318-3961
이메일을 입력하세요=>ds@naver.com

이름	직업	연락처	이메일
김주현	MCSE전문강사	418-9876	freentour@naver.com
박혜경	웹마스터	551-6986	hk@naver.com
김동식	기획A팀대리	318-3961	ds@naver.com

02 Structure pointers and arrays

■ Relationship of structure variables and pointers

- When a structure pointer is described as a structure type.

```
_ _ Structure data type * pointer variable name;
```

포인터 변수 선언 기본 형식

- This structure should come first declaration as to declare structure variables in a structure pointer.

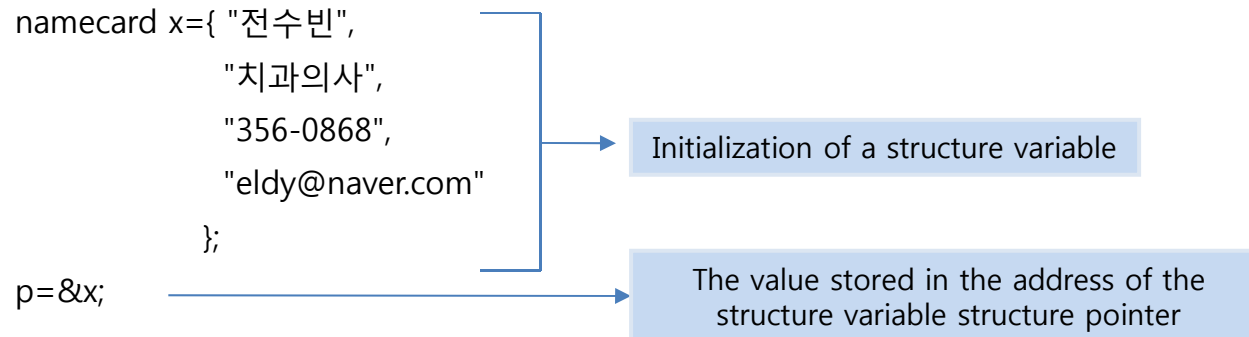
```
struct namecard{  
    char name[20];  
    char job[30];  
    char tel[20];  
    char email[40];  
};  
namecard *p;
```

Declare a Structure

Struct pointer declaration

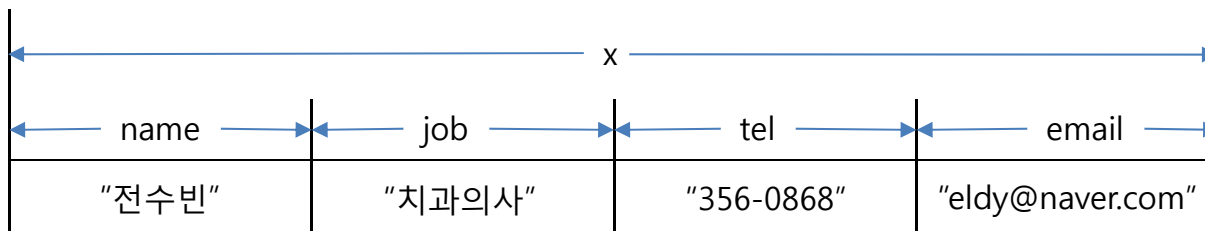
- Pointer variable and must be stored to a particular address, using the assignment operator, where the address of the store address is to be namecard structure variables.

02 Structure pointers and arrays



■ The allocation of structural variables and pointers memory

- `cout<<"sizeof(x) = > " << sizeof(x); // 110 byte`
`cout<<"sizeof(p) = > " << sizeof(p); // 4 byte`
- output is 110 bytes memory size of the structure variable x. because storage space to store the actual value. On the other hand, 4 bytes of a pointer variable p inde, since only stores the address value to point to a structure variable x.



02 Structure pointers and arrays

- Structure pointer variable may obtain the entire structure of the position to the operator *.
- When expressed as follows. Since operator precedence is higher than *, so to find the member variable name for the p. First, an error occurs.

```
*p.name; // 에러 발생 *(p.name)
```

- Therefore, the pointer variable p points to obtain the structure first, so using the parentheses operator to the operation * p it must first expressed as follows.

(*p).name; // p->name Same meaning as

[Table 8-1] Structure and member reference operator

Kinds	How to use
.Operator	Struct variable name. Members
-> Operator	Structure pointer variable name -> Members

Example 8-6. Learn the relationship of structure variables and pointers (08_06.cpp)

```
#include <iostream>
using namespace std;

struct namecard{
    char name[20];
    char job[30];
    char tel[20];
    char email[40];
};

void structPrn(namecard temp)
{
    cout << "₩n 구조체 : " << temp.name << "₩t" << temp.job << "₩t" << temp.tel << "₩t" << temp.email;
}
```

Example 8-6. Learn the relationship of structure variables and pointers (08_06.cpp)

```
void main() {
    namecard x = { "전수빈", "치과의사", "356-0868", "eldy@naver.com" };
    namecard y = { "전원지", "디자이너", "346-0876", "onejee@naver.com" };

    namecard *p;    // 구조체 포인터

    p = &x;
    cout << "Wt      이름 Wt 직업 WtWt 연락처 Wt 이메일 ";
    cout << "Wn
=====";
    cout << "Wn 구조체포인터: " << (*p).name << "Wt" << (*p).job << "Wt" << (*p).tel << "Wt" << (*p).email;

    p = &y;
    cout << "WnWnWt      이름 Wt 직업 WtWt 연락처 Wt 이메일 ";
    cout << "Wn
=====";
    cout << "Wn 구조체포인터: " << p->name << "Wt" << p->job << "Wt" << p->tel << "Wt" << p->email;
    cout << "WnWn";
}
```


02 Structure pointers and arrays

■ Use of structure pointers

- Function pointer to a struct as a parameter, "Address delivered by the way (call by address) "

```
#include <iostream>
using namespace std;

struct namecard{
    char name[20];
    char job[30];
    char tel[20];
    char email[40];
};

void structPrn(namecard temp)
{
    cout << "원 구조체 : " << temp.name << "원" << temp.job << "원" << temp.tel << "원" << temp.email;
}
```

Example 8-7. Creating a function using a structure pointer as a parameter (08_07.cpp)

```
// 구조체 포인터를 매개변수로 전달하는 경우
// Call by address

void structInput(namecard *pTemp)
{
    cout << " 이름을 입력하세요  :";
    cin >> pTemp->name;
    cout << " 직업을 입력하세요  :";
    cin >> pTemp->job;
    cout << " 연락처를 입력하세요 :";
    cin >> pTemp->tel;
    cout << " 이메일을 입력하세요 :";
    cin >> pTemp->email;
    cout << "\n";
}
```

```
void main()
{
    namecard x, y, z;

    structInput(&x);    // 구조체 반환값을 갖는 함수 호출
    structInput(&y);
    structInput(&z);

    cout << "Wt   이름 Wt 직업 WtWt 연락처 Wt 이메일 ";
    cout << "Wn ======";

    structPrn(x);    // 구조체 매개변수 x
    structPrn(y);    // 구조체 매개변수 y
    structPrn(z);    // 구조체 매개변수 z

    cout << "Wn ======Wn";
}
```

02 Structure pointers and arrays

■ Array of structures

```
namecard x;  
namecard y;  
namecard z;
```

- x, a structure variable of the same type as in the y, z may be used in an array declaration..

```
namecard x[3] = { { "김주현", "MCSE전문강사", "418-9876", "freentour@naver.com" },  
                  { "박혜경", "웹마스터", "551-6986", "hk@naver.com" },  
                  { "김동식", "기획A팀대리", "318-3961", "ds@naver.com" }  
                };
```

- When drawing a memory space allocated to the array structure of Figure as follows.

	← name →	← job →	← tel →	← email →
x[0]	"김주현"	"MCSE전문강사"	"418-9876"	"freentour@naver.com"
x[1]	"박혜경"	"웹마스터"	"551-6986"	"hk@naver.com m"
x[2]	"김동식"	"기획A팀대리"	"318-3961"	"ds@naver.com m"

Example 8-9. Using an array of structures (08_09.cpp)

```
01 #include <iostream>
02 using namespace std;
03 struct namecard{
04     char name[20];
05     char job[30];
06     char tel[20];
07     char email[40];
08 };
09 void main()
10 {
11     namecard x[3]={ {"김주현", "MCSE전문강사", "418-9876", "freentour@naver.com"},
12                     {"박혜경", "웹마스터", "551-6986", "hk@naver.com"},
13                     {"김동식", "기획A팀대리", "318-3961", "ds@naver.com"}
14 };
15     cout<<" 이름 Wt 직업 WtWt 연락처 Wt 이메일 Wn";
16     cout<<"=====Wn";
17     for(int i = 0 ; i < 3; i++)
18         cout<<x[i].name<<"Wt"<<x[i].job<<"Wt"<<x[i].tel<<"Wt"<<x[i].email<<"Wn ";
19     cout<<"=====Wn";
20 }
```



이름	직업	연락처	이메일
김주현	MCSE전문강사	418-9876	freentour@naver.com
박혜경	웹마스터	551-6986	hk@naver.com
김동식	기획A팀대리	318-3961	ds@naver.com

03 Enumeration

■ Enumeration

- The enumeration uses the reserved word enum. The enumeration is declared as a set of constants that have a certain pattern. Enum variable will have a certain constant value included in the set.

```
enum COLOR { RED, GREEN, BLUE, WHITE, BLACK };
```

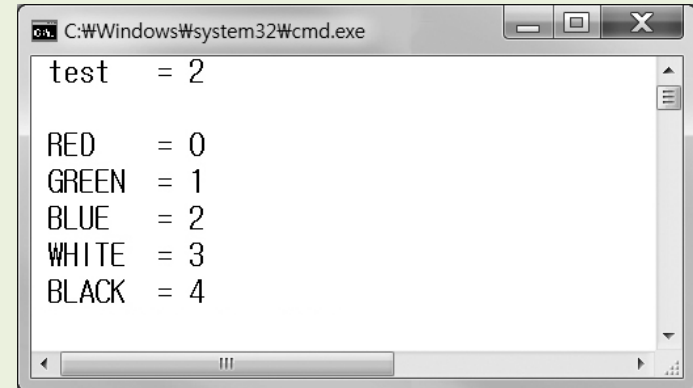
- **COLOR is a new enumeration type.** The value of the enumeration constant is increased if you do not specify the value of the first element is zero and then one by one.

- Since enumeration COLOR is only defined template memory is not allocated. To allocate memory should declare an enum variable like this:

```
enum COLOR test;
```

Example 8-12. Using enumeration (08_12.cpp)

```
01 #include <iostream>
02 using namespace std;
03
04 enum COLOR { RED, GREEN, BLUE, WHITE, BLACK };
05 void main()
06 {
07     enum COLOR test;
08
09     test = BLUE;
10     cout<<" test  = "<<test <<"\n\n"; // test는 정수값 2로 정의되어 있다.
11
12     cout<<" RED    = "<< RED  <<"\n";
13     cout<<" GREEN  = "<< GREEN<<"\n";
14     cout<<" BLUE   = "<< BLUE  <<"\n";
15     cout<<" WHITE  = "<< WHITE <<"\n";
16     cout<<" BLACK  = "<< BLACK<<"\n";
17 }
```



```
C:\Windows\system32\cmd.exe
test  = 2

RED    = 0
GREEN  = 1
BLUE   = 2
WHITE  = 3
BLACK  = 4
```

04 typedef

- typedef is a command that can be applied only for the data type as a reserved word that you use to specify the name of the new data types that are already in use.

① typedef int * PTR ② PTR ptr1, ptr2, ptr03;

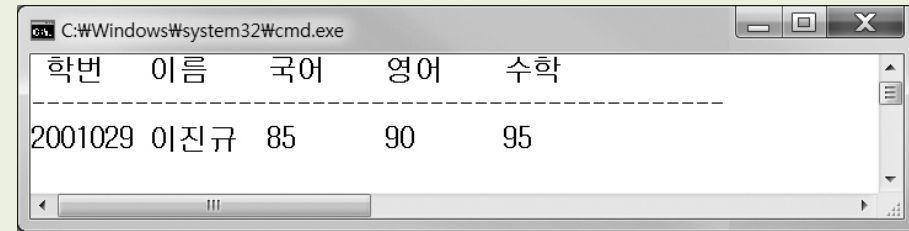
- ① Placed in groups by defining the PTR using a typedef statement to int * ② If you declare a variable, as recognized by ptr1, ptr2, ptr03 both the pointer.
- The typedef may not use a reserved word struct is to declare structure variables.

```
typedef struct sungjuck SJ;
```

SJ s; // Without using a struct is a reserved word can be declared as a structure variable tag names.

Example 8-14. Using typedef (08_14.cpp)

```
01 #include <iostream>
02 using namespace std;
03 // 성적 관리를 위한 구조체(템플릿) 정의
04 struct sungjuck {
05     char no[8]; char name[16]; // 학번과 이름
06     int kor, eng, mat, tot;    // 국어, 영어 수학 점수, 총점
07     double ave;              // 평균
08     char level;              // 학점
09     int grade;               // 등수
10 };
11
12 typedef struct sungjuck SJ;    // 구조체 자료형 sungjuck의 이름을 SJ로 재 정의
13
14 void main()
15 {
16     SJ s={"2001029", "이진규", 85, 90, 95};
17
18     cout<<" 학번 Wt이름 Wt국어 Wt영어 Wt수학 Wn";
19     cout<<"----- Wn";
20     cout<<s.no<<"Wt"<<s.name<<"Wt"<<s.kor<<"Wt"<<s.eng<<"Wt"<<s.mat<<"Wn";
21 }
```



Homework

- Chapter 8 Exercise: 1, 2, 3, 4, 5, 7, 10