



Chapter 08. 구조체

목차

1. 구조체의 이해
2. 구조체 포인터와 배열
3. 공용체와 열거형
4. typedef

학습목표

- struct를 사용해서 새로운 자료형을 정의한다.
- 구조체를 다양하게 사용한다.
- 매개변수가 구조체인 함수를 정의한다.
- 구조체 포인터를 이용해서 구조체를 간접 참조한다.
- 구조체로 배열을 선언하고 사용한다.
- 공용체를 정의하고 사용하는 방법을 학습한다.
- 열거체를 정의하고 사용하는 방법을 학습한다.
- typedef를 정의하는 방법을 학습한다.

01 구조체의 이해

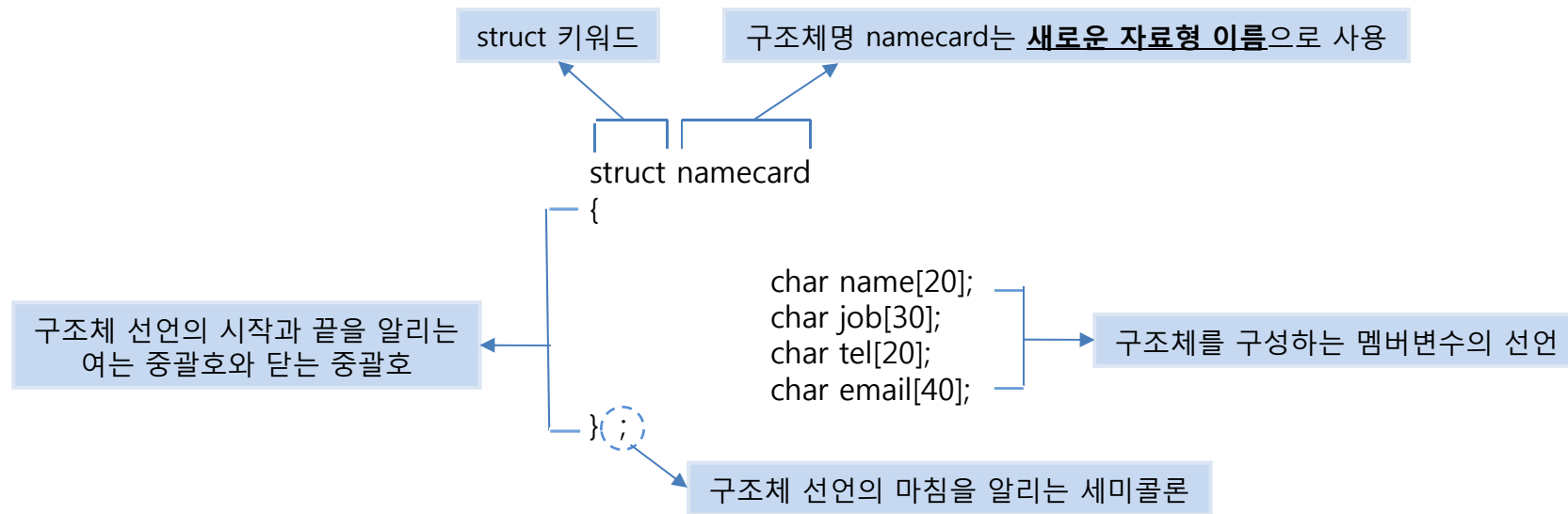
■ 구조체 (새로운 자료형) 선언

```
struct 구조체명{  
    자료형 멤버변수명;  
    자료형 멤버변수명;  
    ...  
    ...  
};
```

구조체 선언 기본 형식

- ① 구조체명 : struct라는 예약어로 서로 다른 구조체 여러 개를 선언할 경우, 이들을 구분하려고 지정한다. 구조체 선언 후에 구조체 변수를 선언할 때 이 구조체명을 사용한다.
- ② 구조체 멤버변수 선언 : 구조체 선언문은 멤버 여러 개로 구성되므로 중괄호로 묶어서 그 내부에 멤버변수를 선언한다. 구조체 멤버변수는 일반 변수나 배열을 선언하는 방법과 동일하다.

01 구조체의 이해



[그림 8-1] 명함 관리 프로그램의 구조체 예

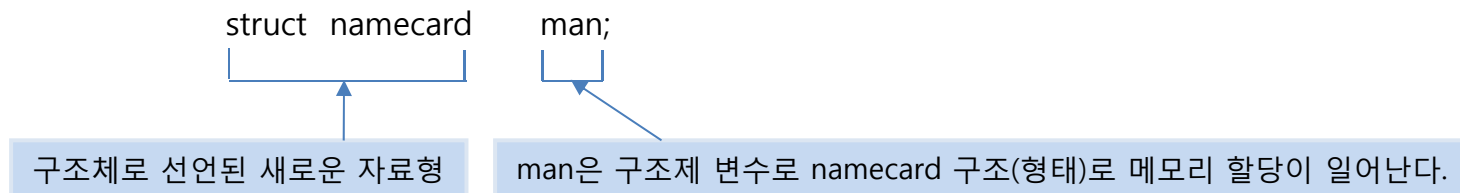
- 구조체 선언은 새로운 자료형을 만들어 내는 것을 의미하는데, 자료형은 변수를 선언하는 용도로 사용될 때 진정한 의미가 있다.

구조체 변수 선언 기본 형식

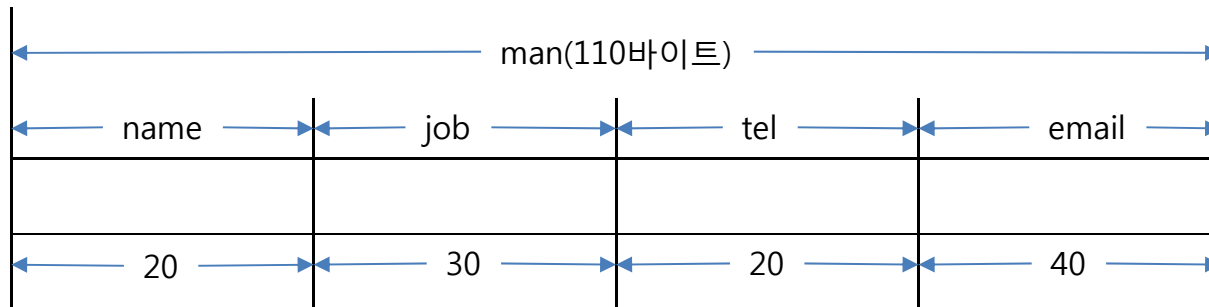
```
struct 구조체명 구조체변수명1, 구조체변수명2, ..., 구조체변수명n;
```

01 구조체의 이해

- 구조체 선언문(정의)은 일반적으로 파일의 머리 부분에 기술한다. 구조체 선언이 되어 있지 않은 상태에서 구조체 변수를 선언하면 "구조체 namecard가 선언되어 있지 않다."는 컴파일 에러가 발생한다.



- 구조체 변수 `man`이 메모리에 어떻게 할당되는지 그림으로 표현해 보자.



01 구조체의 이해

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

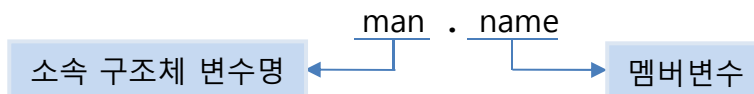
```
namecard man; // ----- ❷
```

- 구조체 변수를 선언하는 방법은 ❶과 같이 예약어 struct를 붙이기도 하지만 ❷와 같이 예약어 struct를 생략하고 구조체명만으로 구조체 변수를 선언하는 것을 허용한다.
- 선언된 구조체 변수를 사용해 보자.

```
구조체_변수명.멤버변수;
```

구조체 변수 기본 형식

- 멤버들의 집합체인 구조체는 멤버변수 단위로 값을 저장하는데, 멤버변수는 특정 구조체에 소속된 구성원이어서 단독적으로 사용하지 못하므로 다음 예처럼 . 연산자 앞에 반드시 구조체 변수명을 지정해야 한다.



예제 8-1. 구조체를 정의하고 사용하기(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 구조체의 이해

■ 구조체 변수의 초기화

- 배열처럼 값 4개를 콤마(,)로 구분해서 나열하고 전체를 중괄호로 묶는다.

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

- 일반 변수에서 한꺼번에 변수 2개 이상을 선언하면서 초기값을 줄 수 있듯이, 구조체 변수도 다음과 변수에 여러 초기값을 구분해서 나열하고 중괄호로 묶을 수 있다.

- 일반 변수 초기값 설정 예

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

- 구조체 변수 초기값 설정 예

```
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"};
```

예제 8-2. 구조체 초기화 하기(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 with the title bar "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 구조체의 이해

■ 구조체 단위로 값 대입하기

- 구조체 변수에 저장된 값을 다른 구조체 변수에 저장하고자 할 때 다음과 같이 표현할 수 있다. 대입 연산자로 동일한 구조체로 선언된 변수에 대해서 구조체 단위로 값을 복사할 수 있다. 만약 이것이 불가능하다면 일일이 멤버변수, 즉 필드 단위로 값을 복사해야 했을 것이다.

```
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);
```

예제 8-3. 구조체 단위로 값 복사하기(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<<"WtWt 이름 Wt 직업 WtWt 연락처 Wt 이메일 ";
17     cout<<"\nWtWt =====";
18     cout<<"\n<구조체 변수 x>";
19     cout<<"Wt"<<x.name <<"Wt"<<x.job <<"Wt"<<x.tel <<"Wt"<<x.email;
20     cout<<"\n<구조체 변수 y>";
21     cout<<"Wt"<<y.name<<"Wt"<<y.job<<"Wt"<<y.tel<<"Wt"<<y.email<<"\n";
22 }

```



01 구조체의 이해

■ 함수의 매개변수로 사용하는 구조체

- 구조체는 함수의 매개변수가 될 수 있다.

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

```
structPrn(x);           // 구조체 변수 x를 매개변수로 함수에 전달  
structPrn(y);  
structPrn(z);
```

- structPrn 함수의 형식 매개변수 temp는 호출 시 넘겨준 실 매개변수의 값을 복사하여 저장한다.
형식 매개변수 temp는 함수 호출 시 x를 넘겨주면 x값을, y를 넘겨주면 y값을, z를 넘겨주면 z값을 복사해서 멤버변수 단위로 접근해서 출력한다.

```
void structPrn(namecard temp) // 함수의 정의  
{  
    cout<<"\n"<<temp.name<<"\t"<<temp.job<<"\t"<<temp.tel<<"\t"<<temp.email;  
}
```

예제 8-4. 값에 의한 전달 방식으로 구조체를 매개변수로 사용하는 함수(08_04.cpp)

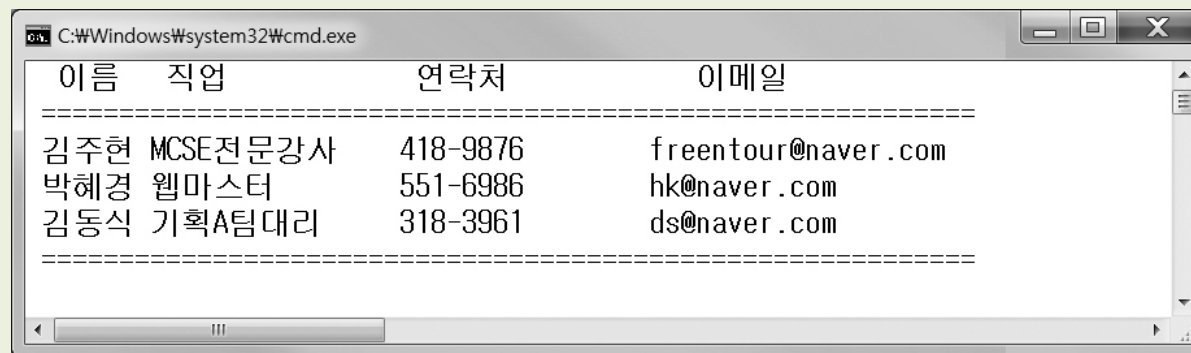
```
#include <iostream>
using namespace std;

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

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

예제 8-4. 값에 의한 전달 방식으로 구조체를 매개변수로 사용하는 함수(08_04.cpp)

```
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 구조체의 이해

■ 구조체를 반환값으로 갖는 함수

- 키보드에서 입력 받은 값을 구조체 변수에 저장하려고 다음과 같이 프로그램을 작성했다.

```
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;
```

- 구조체 값을 입력하는 부분을 함수로 작성해 보자.

01 구조체의 이해

- 구조체 값을 입력하는 부분을 함수로 구현할 경우에는 호출된 함수가 호출한 함수로 입력 받은 값을 반환해야 한다. 다음은 키보드에서 입력한 값을 구조체에 저장하기 위한 함수이다.

```
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;
}
```

예제 8-5. 구조체를 반환하는 함수 작성하기(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
}
```

예제 8-5. 구조체를 반환하는 함수 작성하기(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 구조체 포인터와 배열

■ 구조체 변수와 포인터의 관계

- 구조체 포인터는 자료형으로 구조체를 기술하면 된다.

포인터 변수 선언 기본 형식

```
구조체_자료형 * 포인터_변수명;
```

- 구조체 포인터도 구조체 변수를 선언할 때처럼 구조체 선언문이 먼저 나와야 한다.

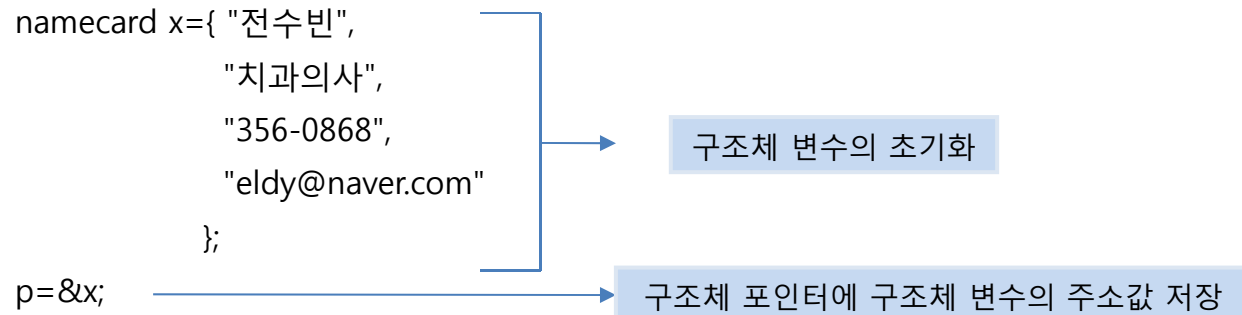
```
struct namecard{  
    char name[20];  
    char job[30];  
    char tel[20];  
    char email[40];  
};  
namecard *p;
```

구조체 선언

구조체 포인터 선언

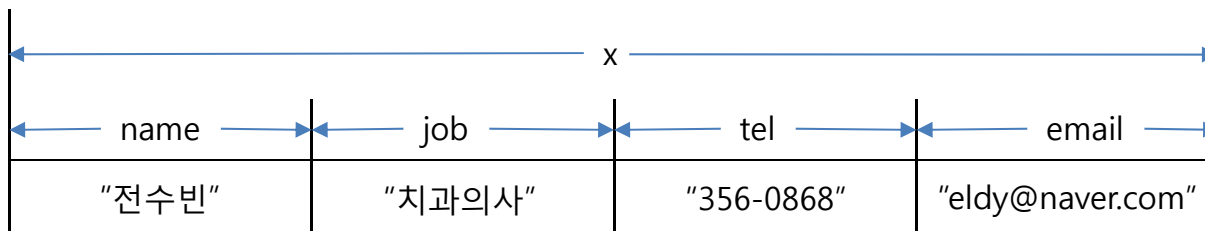
- 포인터 변수는 대입 연산자를 이용해서 특정 주소를 저장하고 있어야 하며, 이때 저장할 주소는 namecard 구조체 변수의 주소가 되어야 한다.

02 구조체 포인터와 배열



■ 구조체 변수와 포인터의 메모리 할당

- `cout<<"sizeof(x) = > " << sizeof(x); // 110 byte`
`cout<<"sizeof(p) = > " << sizeof(p); // 4 byte`
- 구조체 변수 x의 메모리 크기를 출력해보면 110바이트다. 이는 실질적인 값을 저장하는 기억공간이기 때문이다. 반면 포인터 변수 p는 4바이트인데, 이는 구조체 변수 x를 가리키기 위한 주소값만을 저장하기 때문이다.



02 구조체 포인터와 배열

- 구조체 포인터 변수는 * 연산자로 해당 위치의 구조체 전체를 얻을 수 있다.
다음과 같이 표현하면 . 연산자가 우선순위가 * 보다 높으므로 p에 대한 멤버변수 name을 먼저 찾으려 하므로 에러가 발생한다.

```
*p.name; // 에러 발생 *(p.name)
```

- 따라서 포인터 변수 **p가 가리키는 구조체를 먼저 얻어야 하므로** *p를 먼저 연산할 수 있도록 괄호 연산자를 사용해서 다음과 같이 표현해야 한다.

```
(*p).name; // p->name와 같은 의미
```

[표 8-1] 구조체와 멤버참조 연산자

종류	사용법
. 연산자	구조체 변수명.멤버
-> 연산자	구조체 포인터 변수명 -> 멤버

예제 8-6. 구조체 변수와 포인터의 관계 알아보기(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 << "구조체 : " << temp.name << " " << temp.job << " " << temp.tel << " " << temp.email;
}
```

예제 8-6. 구조체 변수와 포인터의 관계 알아보기(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 구조체 포인터와 배열

■ 구조체 포인터의 용도

- 구조체 포인터를 매개변수로 하는 '주소에 의한 전달 방식(call by address)'의 함수

```
#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;
}
```

예제 8-7. 구조체 포인터를 매개변수로 사용하는 함수 작성하기(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 구조체 포인터와 배열

■ 구조체 배열

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

- x, y, z와 같이 동일한 형태의 구조체 변수들을 배열로 선언해서 사용할 수 있다.

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

- 구조체 배열로 할당된 기억공간을 그림으로 그려보면 다음과 같다.

	← 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"

예제 8-9. 구조체 배열 사용하기(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 열거형

■ 열거형

- 열거형은 예약어 enum을 사용한다. 열거형은 일정한 패턴이 있는 상수들을 하나의 집합으로 선언한다. 열거형 변수는 집합 안에 포함되는 특정 상수값을 가지게 된다.

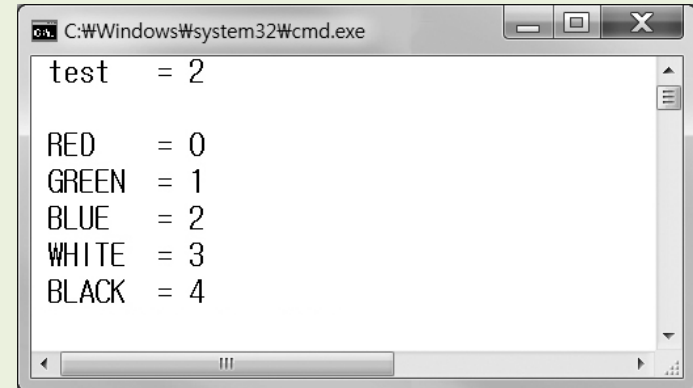
```
enum COLOR { RED, GREEN, BLUE, WHITE, BLACK };
```

- 열거형 COLOR은 새로운 자료형이다. 열거형 상수의 값은 따로 지정하지 않는 경우 첫 원소가 0이고 값이 차례로 1씩 증가한다.
- 열거형 COLOR은 템플릿만 정의한 것이기 때문에 메모리는 할당되지 않는다. 메모리를 할당하려면 다음처럼 열거형 변수를 선언해야 한다.

```
enum COLOR test;
```

예제 8-12. 열거형 사용하기(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

- typedef는 이미 사용하고 있는 자료형의 이름을 새롭게 지정할 때 사용하는 예약어로 자료형에 대해서만 적용할 수 있는 명령어다.

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

- ①에서 typedef 문을 사용해서 PTR를 int *로 정의해 두었기에 ②와 같이 변수를 선언하면 ptr1, ptr2, ptr03 모두 포인터 변수로 인식한다.

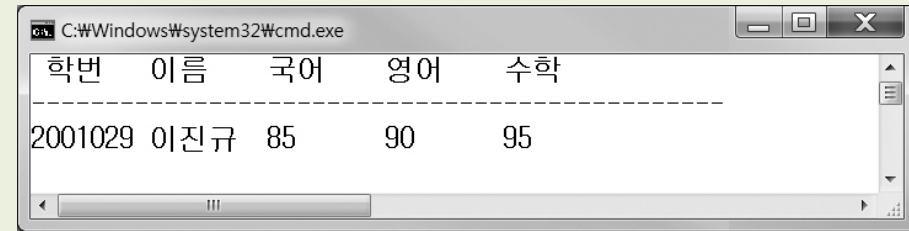
- typedef를 사용하면 구조체 변수를 선언할 때 struct란 예약어를 사용하지 않을 수 있다.

```
typedef struct sungjuck SJ;
```

```
SJ s; // struct란 예약어를 사용하지 않고 태그명으로 구조체 변수 선언을 할 수 있다.
```

예제 8-14. typedef 사용하기(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 }
```



학번	이름	국어	영어	수학
2001029	이진규	85	90	95

Homework

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