15강

09~14강 요약

동양미래대학교 강환수 교수

본 강의 사용 및 참조 자료

- Perfect C, 3판, 강환수 외 2인 공저, 인피니티북스, 2021

11장	포인터 기초
16장	동적메모리와 전처리

목차

목차

09

포인터

포인터 변수 선언

- 포인터 변수, 간단히 포인터
- 변수 선언
  - 자료형과 포인터 변수 이름 사이에 연산자 *(asterisk)를 삽입
  - int *ptrint
    - ‘int 포인터 ptrint’라고 읽음
  - 포인터 변수 선언

    자료형 *변수이름 ;

```
int *ptrint;
short*ptrshort;
char * ptrchar;
double *ptrdouble;
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 466쪽

포인터에 주소 값 저장

- 어느 변수의 주소 값을 저장하려면
  - 반드시 그 변수의 자료유형과 동일한 포인터 변수에 저장

```
int data = 100;
int *ptrint;

ptrint = &data;
```

변수 data를 선언해 100을 저장

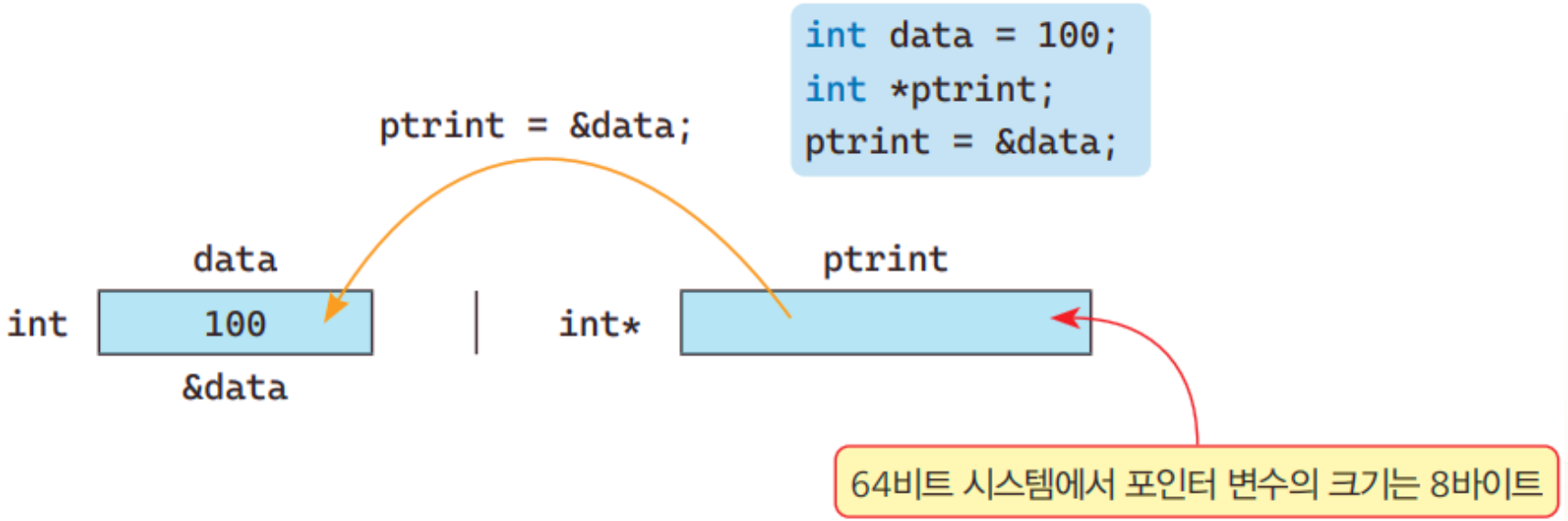
변수 ptrint는 정수 int형 변수의 주소를 저장하는 포인터 변수이며, 이 문장으로 포인터 변수 ptrint를 선언만 해 쓰레기 주소값이 저장

변수 ptrint에 data의 주소 값을 저장하는 문장

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 466쪽

포인터에 주소 값 저장

- '포인터 변수 ptrint는 변수 data를 가리킨다'
  - 또는 '참조(reference)한다'라고 표현
- 64비트 시스템인 윈도 10에서 포인터 변수
  - 가리키는 변수의 종류에 관계없이 크기가 모두 8바이트

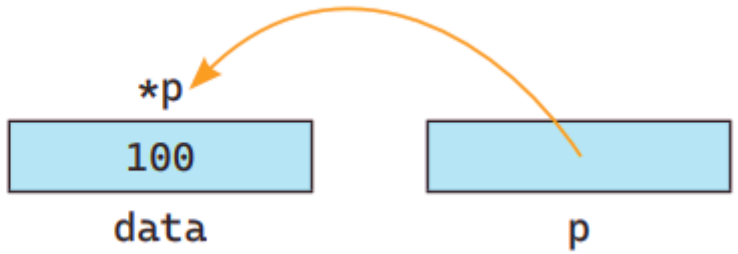


[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 467쪽

단항 연산자인 간접연산자 *

- 포인터 변수가 갖는 주소로 그 주소의 원래 변수를 참조하는 연산자와 방법
  - 간접연산자(indirection operator) *를 사용한 역참조
    - 전위연산자(피연산자 앞에 위치)로 피연산자는 포인터
    - *p는 피연산자인 p가 가리키는 변수 자체를 반환
  - 포인터 p는 data의 주소 값을 가지므로 *p는 data와 같음

```
int data = 100;
int *p = &data;
printf("간접참조 출력: %d \n", *p);
```



[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 472쪽

포인터 배열 개요와 선언

- 포인터 배열(array of pointer)
  - 주소값을 저장하는 포인터를 배열 원소로 하는 배열
  - 일반 배열 선언에서 변수이름 앞에 *를 붙이면 포인터 배열 변수 선언
- int *pa[3]
  - 배열크기가 3인 포인터 배열
  - pa [0]
    - 변수 a의 주소를 저장
  - pa [1] : 변수 b의 주소를 저장
  - pa [2] : 변수 c의 주소를 저장

포인터 배열 개요와 선언

- double *dary[5] = {NULL};
  - NULL 주소를 하나 지정, 나머지 모든 배열원소에 NULL 주소가 지정
  - 문장 float *ptr[3] = {&a, &b, &c};
    - 변수 주소를 하나, 하나 직접 지정하여 저장 가능
  - 포인터 배열 변수 선언

    자료형 *변수이름[배열크기] ;

```
int *pary[5];
char *ptr[4];
float a, b, c;
double *dary[5] = {NULL};
float *ptr[3] = {&a, &b, &c};
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 497쪽

10

문자와 문자열

문자

- char형 변수에 문자를 저장

```
char ch ='A';
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 509쪽

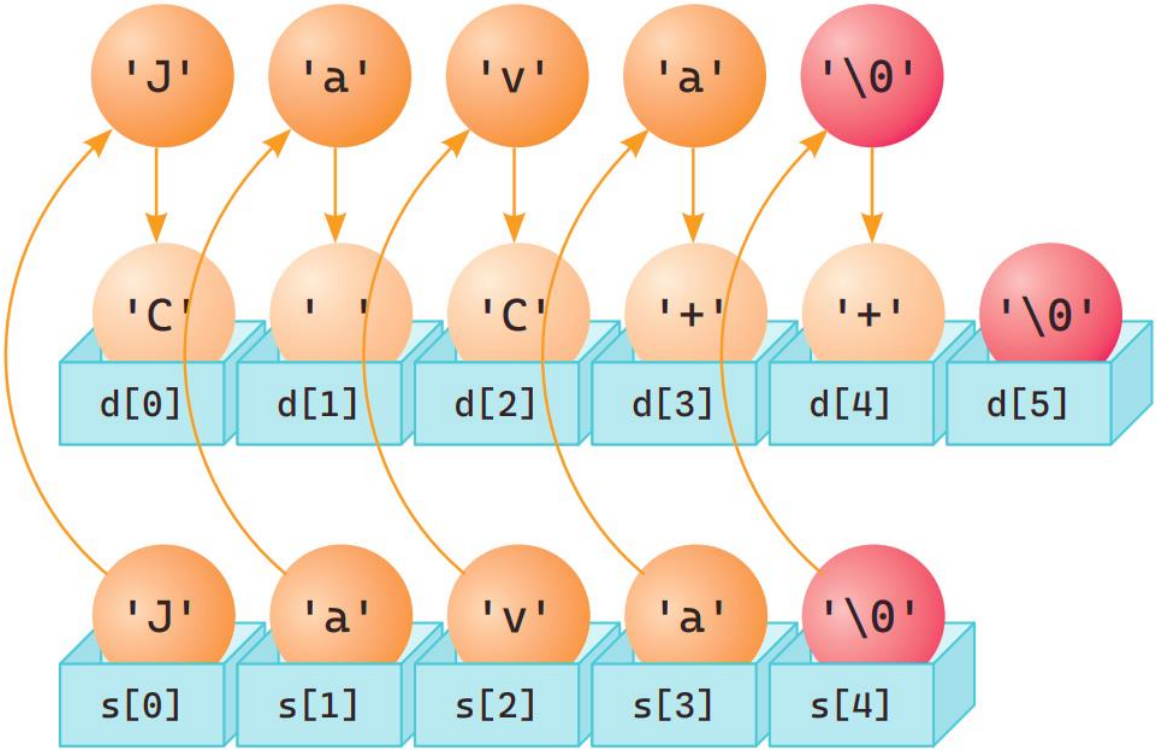
문자열

- 배열 csharp의 크기를 3으로 선언한 후 배열 csharp에 문자열 "C#"을 저장
  - 마지막 원소인 csharp[2]에 '₩0'을 저장

```
char ch ='A';

char csharp[3];
csharp[0] = 'C'; csharp[1] = '#'; csharp[2] = '\0';
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 509쪽

문자열 복사



결과는 d에도 "java"가 저장된다.

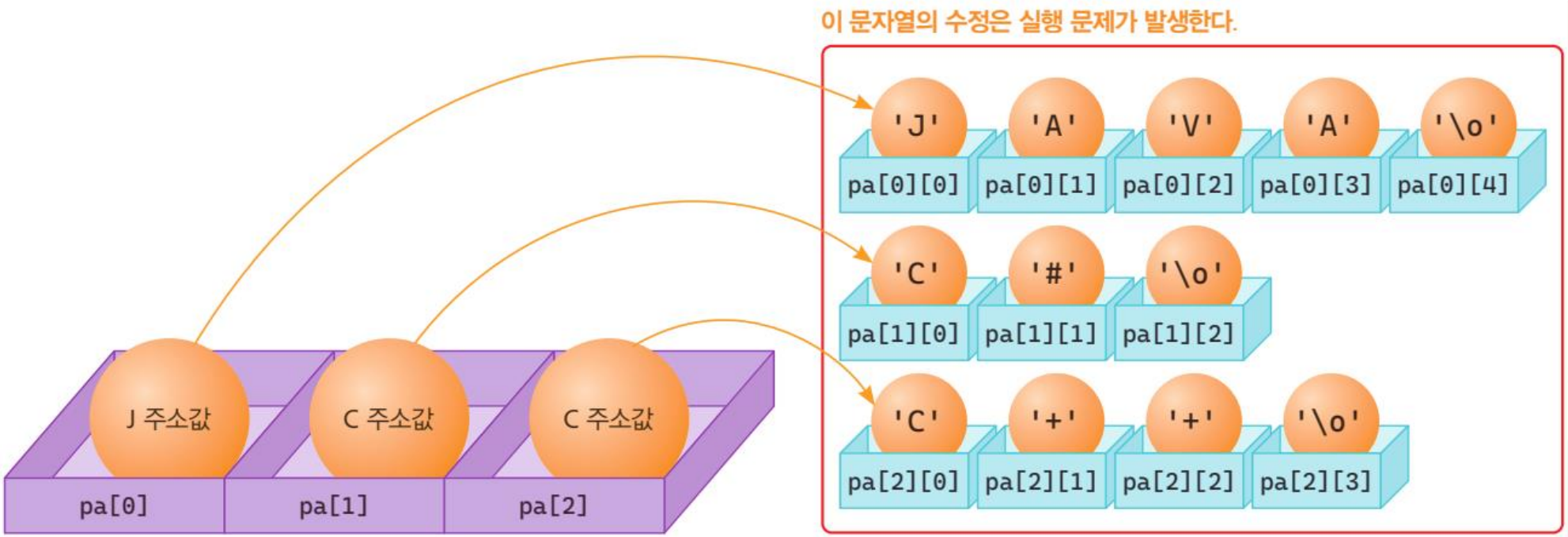
```
char d[] = "C C++";
char s[] = "Java";
strcpy(d, s);
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 526쪽

문자 포인터 배열을 이용, 여러 개의 문자열 처리

```
char *pa[] = {"JAVA", "C#", "C++"};
//각각의 3개 문자열 출력
printf("%s ", pa[0]); printf("%s ", pa[1]); printf("%s\n", pa[2]);
```

배열의 크기는 문자열 개수인 3을 지정하거나 빈 공백으로 한다.



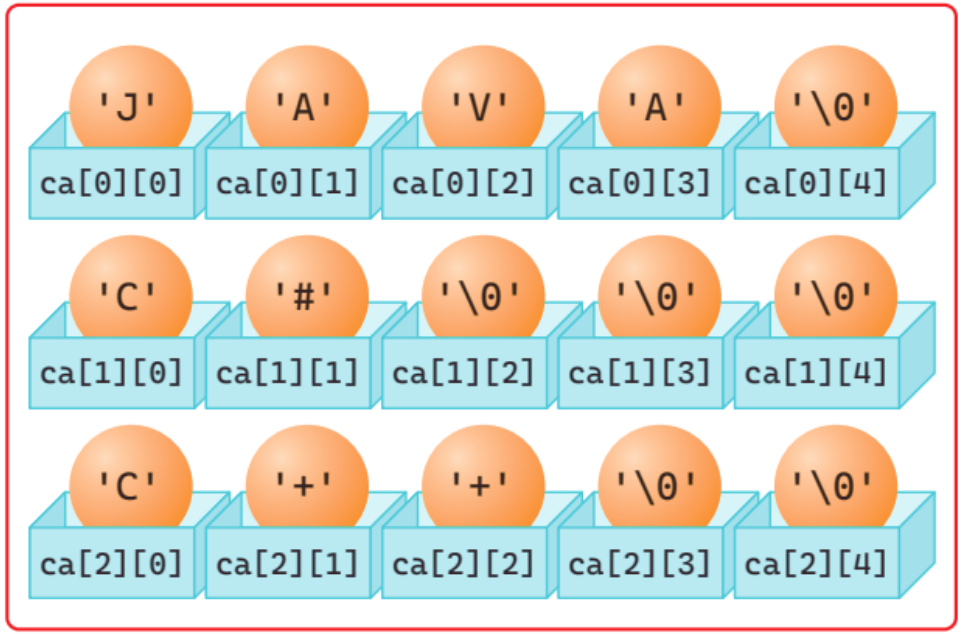
[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 536쪽

2차원 문자배열을 이용한 문자열 처리

```
char ca[][5] = {"JAVA", "C#", "C++"};
//각각의 3개 문자열 출력
printf("%s ", ca[0]); printf("%s ", ca[1]); printf("%s\n", ca[2]);
```

첫 번째(행) 크기는 문자열 개수를 지정하거나 빈 공백으로 두며, 두 번째(열) 크기는 문자열 중에서 가장 긴 문자열의 길이보다 1크게 지정한다.

이 문자열의 수정될 수 있다.



[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 537쪽

실습예제

Prj11 11cmdarg.c 명령행 인자 출력 난이도: ★★

```
01  #include <stdio.h>
02
03  int main(int argc, char* argv[])
04  {
05     int i = 0;
06
07     printf("실행 명령행 인자(command line arguments) >>\n");
08     printf("argc = %d\n", argc);
09     for (i = 0; i < argc; i++)
10        printf("argv[%d] = %s\n", i, argv[i]);
11
12     return 0;
13  }
```

argc(argument count)에 인자의 수가, argv(argument variables)에는 인자인 여러 개의 문자열의 포인터가 저장된 배열이 전달

```
실행 명령행 인자(command line arguments) >>
argc = 4
argv[0] = C:\Kang C\ch12\x64\Debug\Prj11.exe
argv[1] = Python
argv[2] = Go
argv[3] = Kotlin
```

비주얼 스튜디오에서 실행하면 전체 경로를 포함한 실행파일의 이름이 첫 번째 인자로 표시된다.

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 540쪽

11

구조체와 공용체

구조체 정의 방법

구조체 틀: 정의

```
struct lecture
{
    char name[20];  //강좌명
    int credit;     //학점
    int hour;       //시수
};
```

구조체 정의 없이는 자료형
struct lecture를 사용할 수 없다.

구조체를 자료형으로 사용

```
struct lecture datastructure;
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 553쪽

구조체 멤버의 이름은 모두 유일

- 멤버로는 다양한 자료형,
  다른 구조체 변수 및 구조체 포인터도 허용

문자열 입출력 함수: stdio.h

```
struct 구조체태그이름
{
    자료형 변수명1;
    자료형 변수명2;
    ...
};
```

구조체 구성요소(struct member)라 한다.
초기값을 설정할 수 없다.

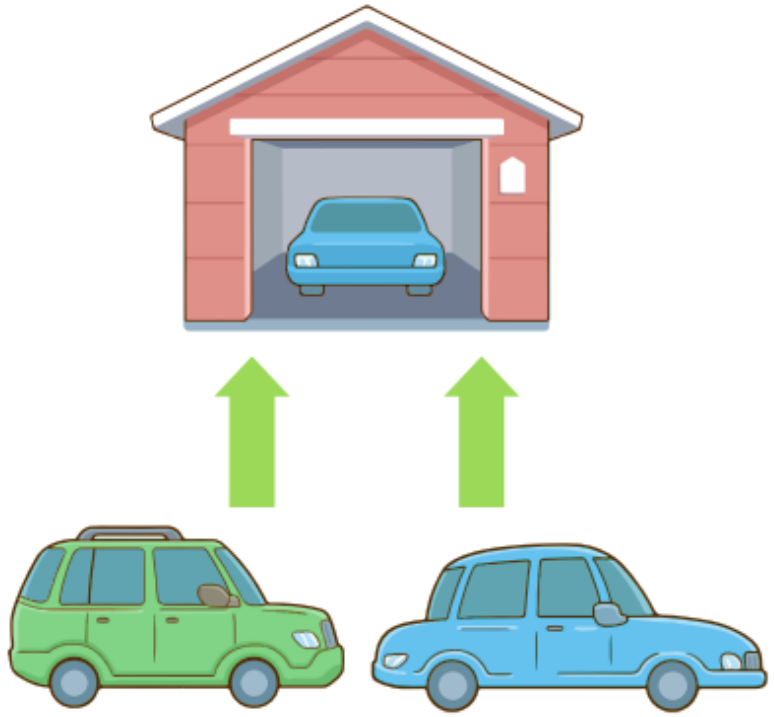
세미콜론은 반드시 필요하다.

```
struct lecture
{
    char name[20];    //강좌명
    int credit;       //학점
    int hour;         //시수
};
```

마지막 멤버 hour에도 반드시 ;이 필요하다.

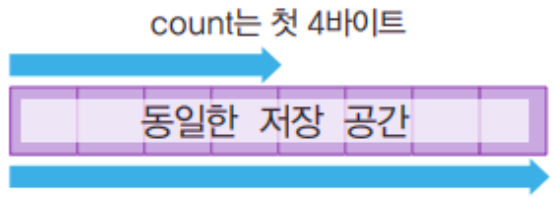
[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 554쪽

공용체 개념



공용체

```
union share
{
    int count;
    double value;
};

union share a;
```




```
a.count = 55;
```

```
a.value = 243.5;
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 566쪽

공용체 정의 및 변수 선언 구문

공용체 정의 및 변수 선언 구문

```
union 공용체태그이름
{
    자료형 멤버변수명1;
    자료형 멤버변수명2;
    ...
} [변수명1] [,변수명2];
```

공용체 구성요소인 멤버(struct member)이다.

세미콜론은 반드시 필요하다.

```
union data
{
    char ch;        //문자형
    int cnt;        //정수형
    double real;    //실수형
} data1;
```

```
union udata
{
    char name[4];   //char형 배열
    int n;          //정수형
    double val;     //실수형
};
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 567쪽

typedef 구문

- 이미 사용되는 자료 유형을
  다른 새로운 자료형 이름으로 재정의
- typedef int profit;
  - profit을 int와 같은 자료형으로 새롭게 정의하는 문장
  - 자료형 재정의 typedef 구문

```
typedef   기존자료유형이름   새로운자료형1, 새로운자료형2, ... ;
```

```
typedef int profit;
typedef unsigned int budget;
typedef unsigned int size_t;
typedef unsigned __int64 size_t;
```

여러 이름으로도 재정의할 수 있다.

새로운 이름 size_t도 자료형 unsigned int와 동일하게 이용 가능하다.

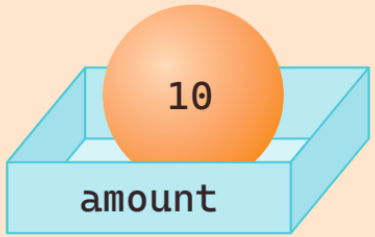
[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 573쪽

12

함수와 포인터 활용

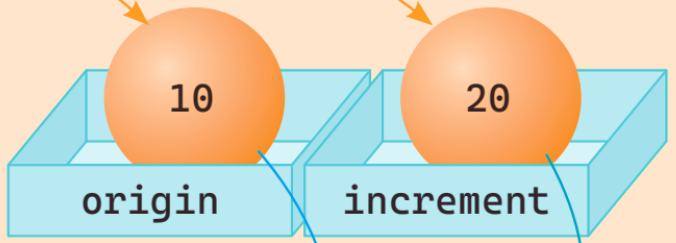
값에 의한 호출

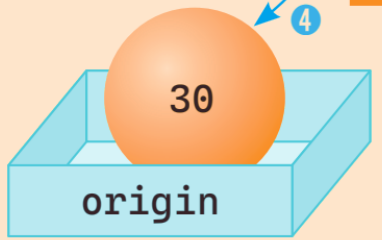
```
...
int main(void)
{
    int amount = 10;

    increase(amount, 20);

    printf("%d\n", amount);
...
}
```



변수 amount는 여전히 10

```
void increase(int origin, int increment)
{
    origin += increment;
}
```

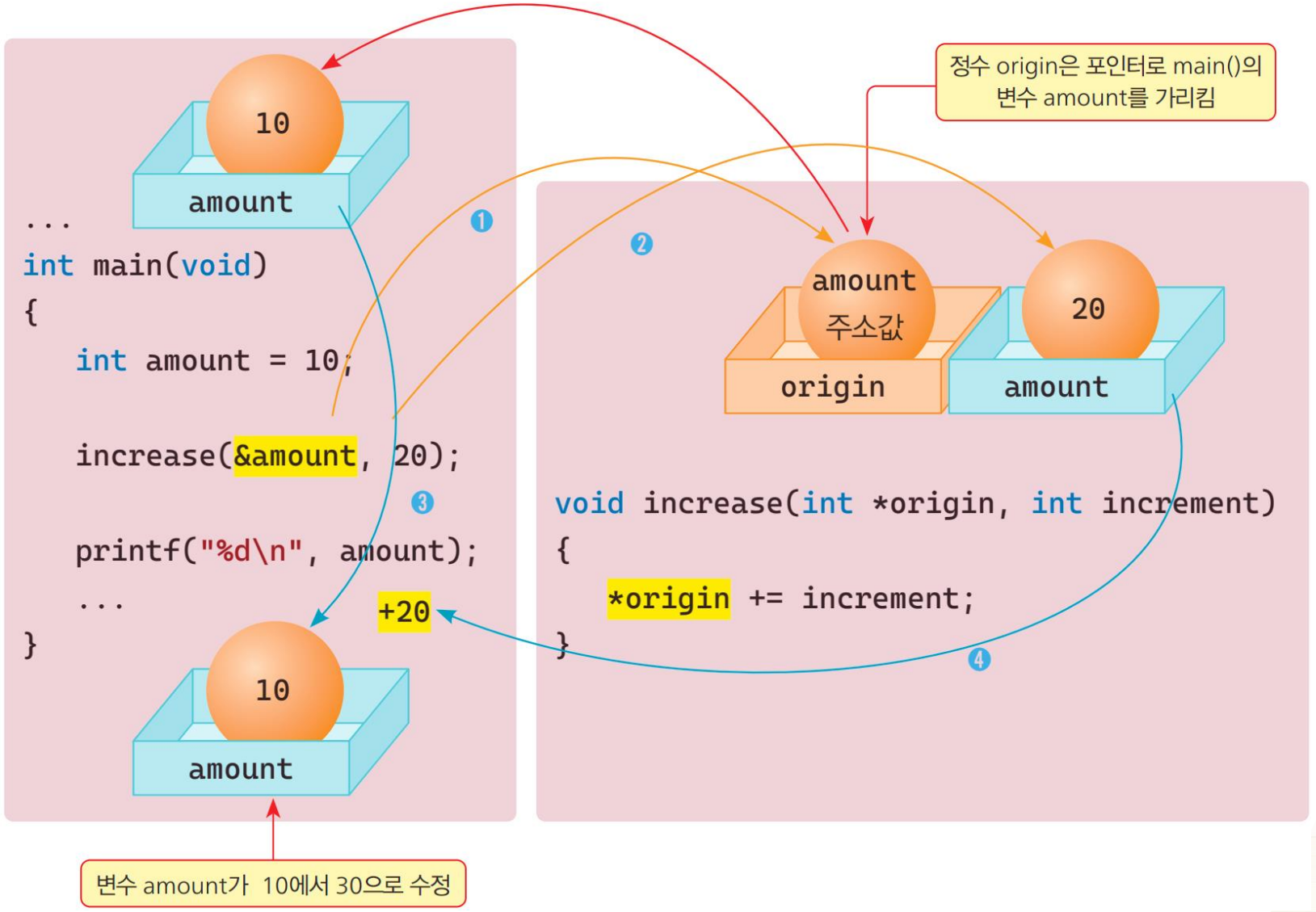




매개변수 origin이 30으로 변하나
main()의 변수 amount와는 무관

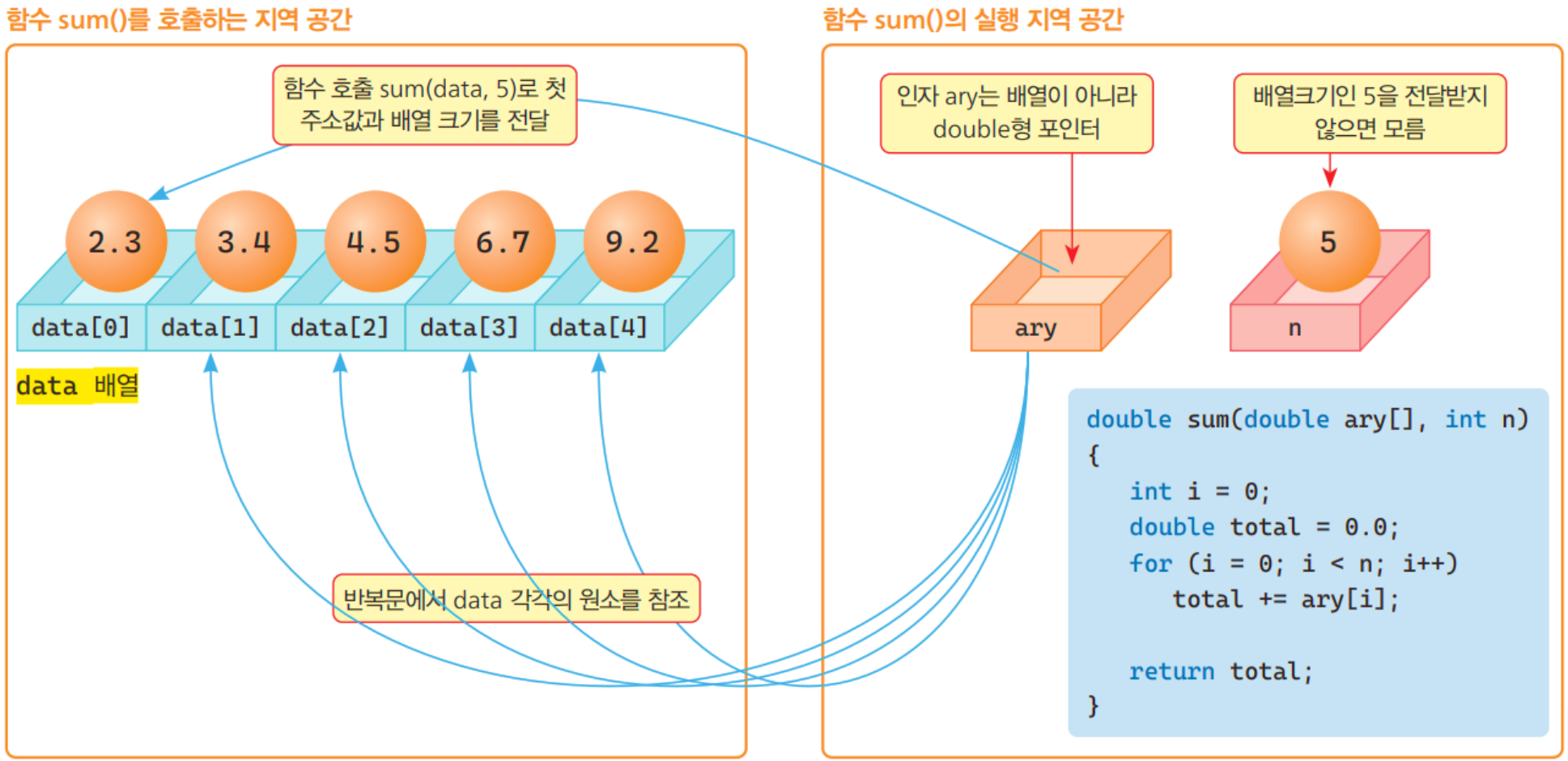
[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 598쪽

주소에 의한 호출(call by address)



[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 599쪽

배열 전달을 위한 함수호출



[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 602쪽

간접연산자 *를 사용한 배열원소의 참조방법

```
int i, sum = 0;
int point[] = {95, 88, 76, 54, 85, 33, 65, 78, 99, 82};
int *address = point;
int aryLength = sizeof (point) / sizeof (int);
```

가능

```
for (i=0; i<aryLength; i++)
  sum += *(point+i);
```

가능

```
for (i=0; i<aryLength; i++)
   sum += *(address++);
```

오류

```
for (i=0; i<aryLength; i++)
  sum += *(point++);
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 602쪽

13

파일 처리

FILE 자료형

- 헤더 파일 stdio.h에 정의되어 있는 구조체 유형
  - 함수 fopen( )의 반환값 유형 FILE *
    - 구조체 FILE의 포인터 유형
  - 함수 fopen( )은 인자가 파일이름과 파일열기 모드
    - 파일 스트림 연결에 성공
      - 파일 포인터를 반환
    - 실패하면
      - NULL을 반환

```
FILE* f; //파일 포인터
char* fname = "basic.txt"; //파일이름

if ((f = fopen(fname, "w")) == NULL)
{
   printf("파일이 열리지 않습니다.\n");
   exit(1);
};
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 650쪽

함수 fclose( )

- fopen( )으로 연결한 파일 스트림을 닫는 기능을 수행
- 파일 스트림을 연결한 후 파일 처리가 모두 끝났으면
  - 파일 포인터 f를 인자로 함수 fclose( )를 호출하여 반드시 파일을 닫도록
    - 내부적으로 파일 스트림 연결에 할당된 자원을 반납
    - 파일과 메모리 사이에 있던 버퍼의 내용을 모두 지우는 역할을 수행
- 성공하면 0을 실패하면 EOF을 반환

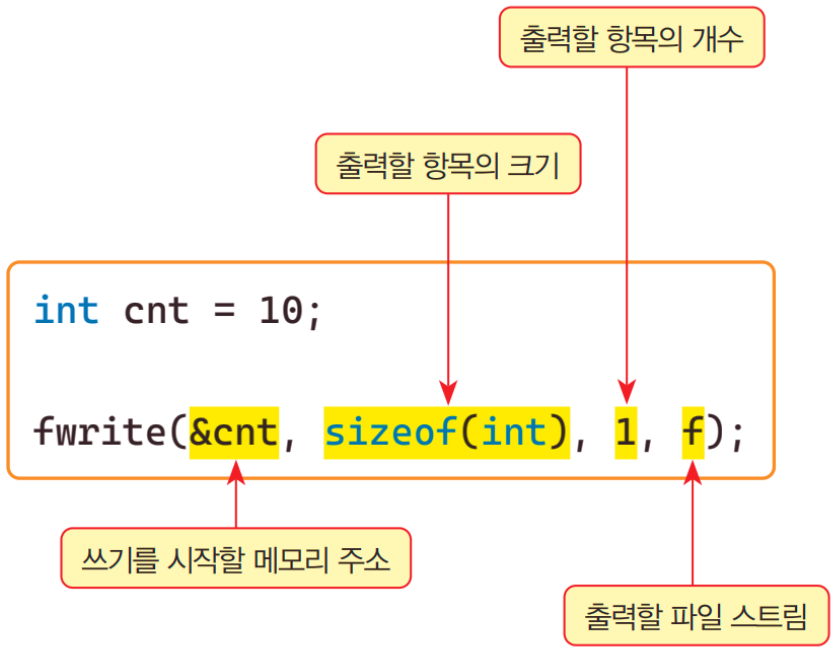
함수 fwrite( )와 fread( )

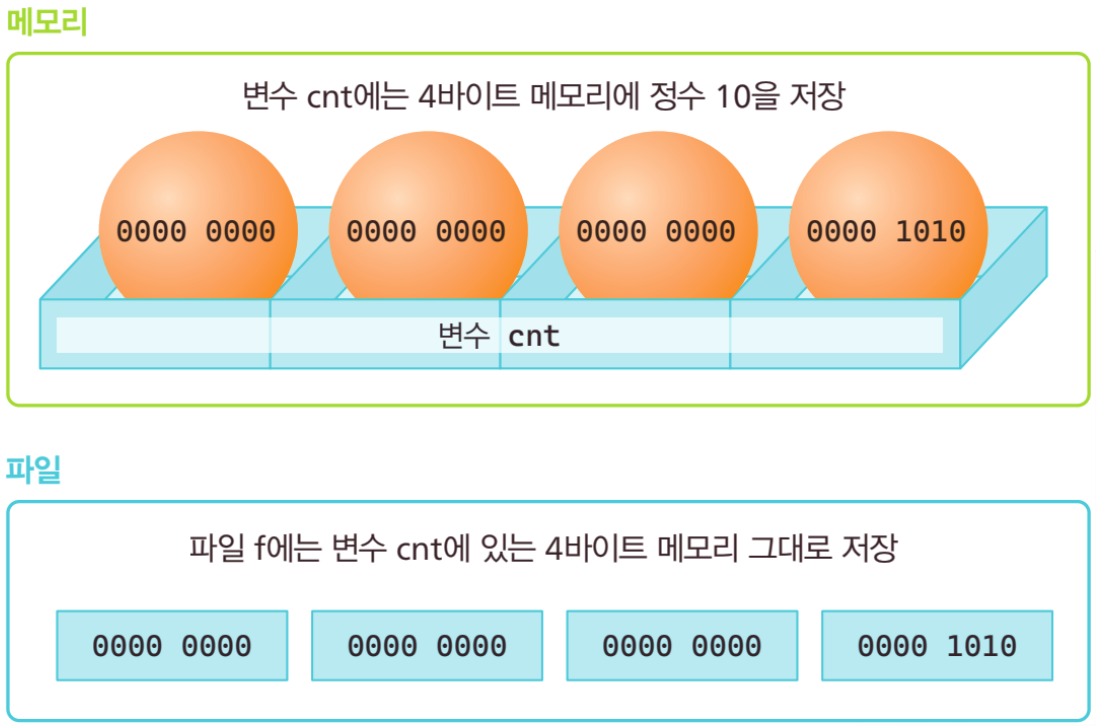
- 이진 모드로 블록 단위 입출력을 처리
- 이진파일(binary file)
  - C 언어의 자료형을 모두 유지하면서 바이트 단위로 저장되는 파일
- 헤더파일 stdio.h 필요

```
int cnt = 10;
fwrite(&cnt, sizeof(int), 1, f);
fread(&cnt, sizeof(int), 1, f);
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 668쪽

함수 fwrite( )에 의한 이진파일 출력





[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 668쪽

함수 fseek( ) 함수 원형

함수 fseek() 함수원형

```
int fseek(FILE * _File, long _Offset, int _Origin);
```

함수 fseek()는 파일 _File의 기준점 _Origin에서 _Offest만큼 파일 포인터를 이동하는 함수 , 성공하면 0을 반환하며 실패하면 0이 아닌 정수를 반환

```
fseek(f, 0L, SEEK_SET);
fseek(f, 100L, SEEK_CUR);
fseek(f, -100L, SEEK_END);
```

기호	값	의미
SEEK_SET	0	파일의 시작 위치
SEEK_CUR	1	파일의 현재 위치
SEEK_END	2	파일의 끝 위치

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 676쪽

함수 ftell( )과 rewind( )

- 함수 ftell( )
  - 인자인 파일의 파일 위치를 반환
- 함수 rewind( )
  - 파일 위치를 무조건 가장 앞으로 이동

함수	기능
int fseek(FILE *, long offset, int pos)	파일 위치를 세 기준점(pos)으로부터 오프셋(offset)만큼 이동
long ftell(FILE *)	파일의 현재 파일 위치를 반환
void rewind(FILE *)	파일의 현재 위치를 0 위치(파일의 시작점)로 이동

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 677쪽

14

동적 메모리

자기참조 구조체(self reference struct)

- 멤버 중의 하나가 자기 자신의 구조체 포인터 변수를 갖는 구조체
- 구조체 selfref
  - 멤버로 int 형 n과 struct selfref * 형 next로 구성
    - 즉 멤버 next의 자료형은 지금 정의하고 있는 구조체의 포인터 형
  - 구조체는 자기 자신 포인터를 멤버로 사용 가능
    - 자기 자신은 멤버로 사용 불가능

```
struct selfref {
    int n;
    struct selfref *next;
    //struct selfref one;    //컴파일 오류 발생
}
```

error C2079: 'one'은(는) 정의되지 않은 struct 'selfref'을(를) 사용합니다.

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 709쪽

연결 리스트(linked list)

자기 참조 구조체

- 동일 구조체의 표현을 여러 개 만들어 연결할 수 있는 기능

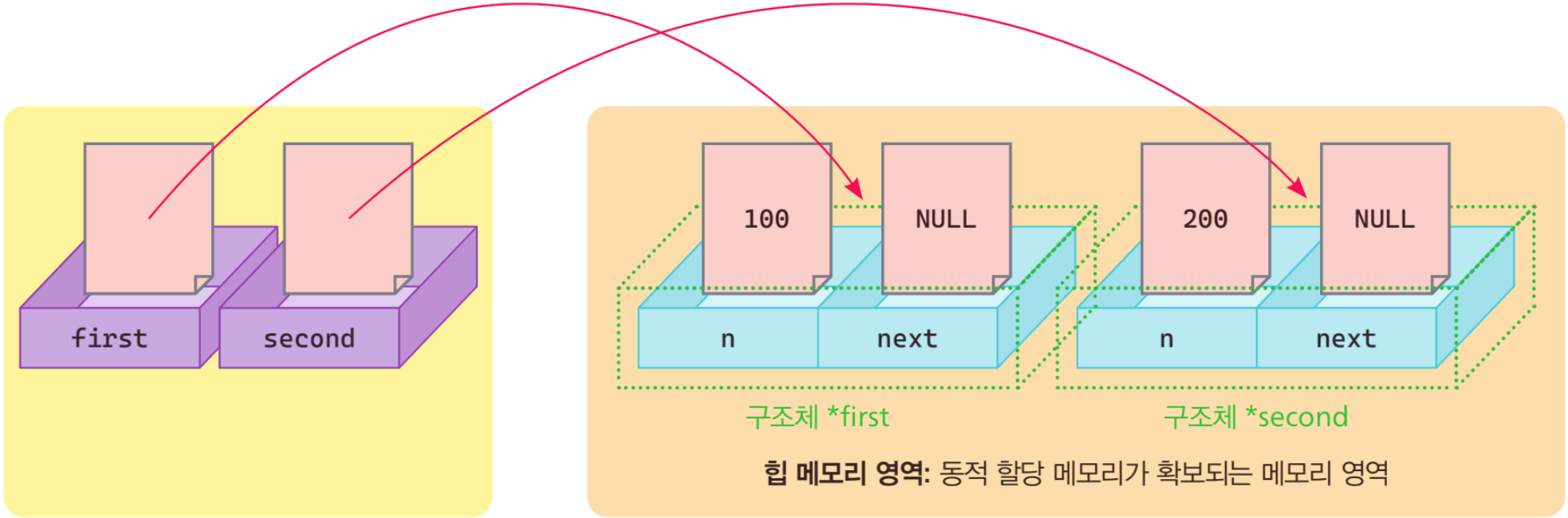
```
//❶ 우선 구조체 struct selfref를 하나의 자료형인 list 형으로 정의
typedef struct selfref list;

//❷ 두 구조체 포인터 변수 first와 second를 선언한 후,
//   함수 malloc()을 이용하여 구조체의 멤버를 저장할 수 있는 저장공간을 할당
list *first = NULL, *second = NULL;
first = (list *)malloc(sizeof(list));
second = (list *)malloc(sizeof(list));

//❸ 구조체 포인터 first와 second의 멤버 n에 각각 정수 100, 200을 저장하고,
//   멤버 next에는 각각 NULL을 저장
first->n = 100;
second->n = 200;
first->next = second->next = NULL;
```

[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 709쪽

구조체의 동적 할당

```
first = (list *)malloc(sizeof(list));
```

```
second = (list *)malloc(sizeof(list));
```

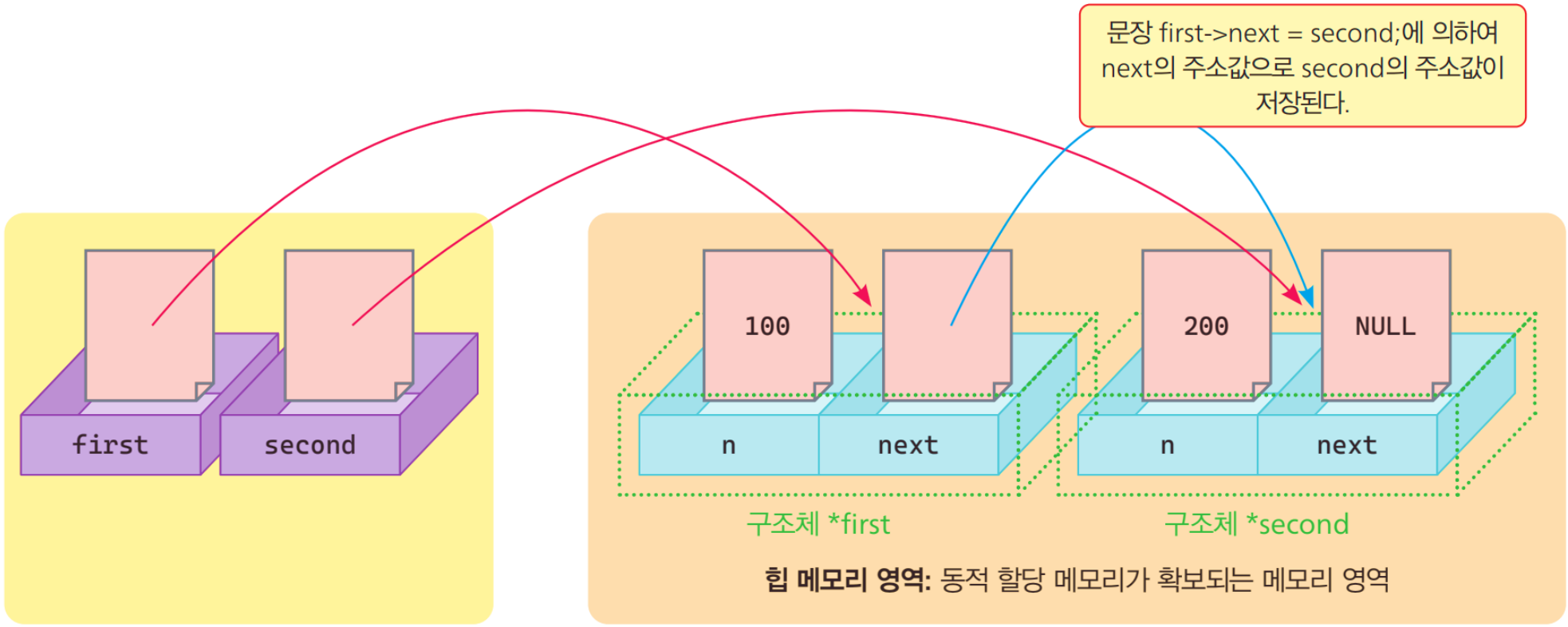



```
first->n = 100;
second->n = 200;
first->next = second->next = NULL;
```


[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 710쪽

구조체의 주소값을 저장

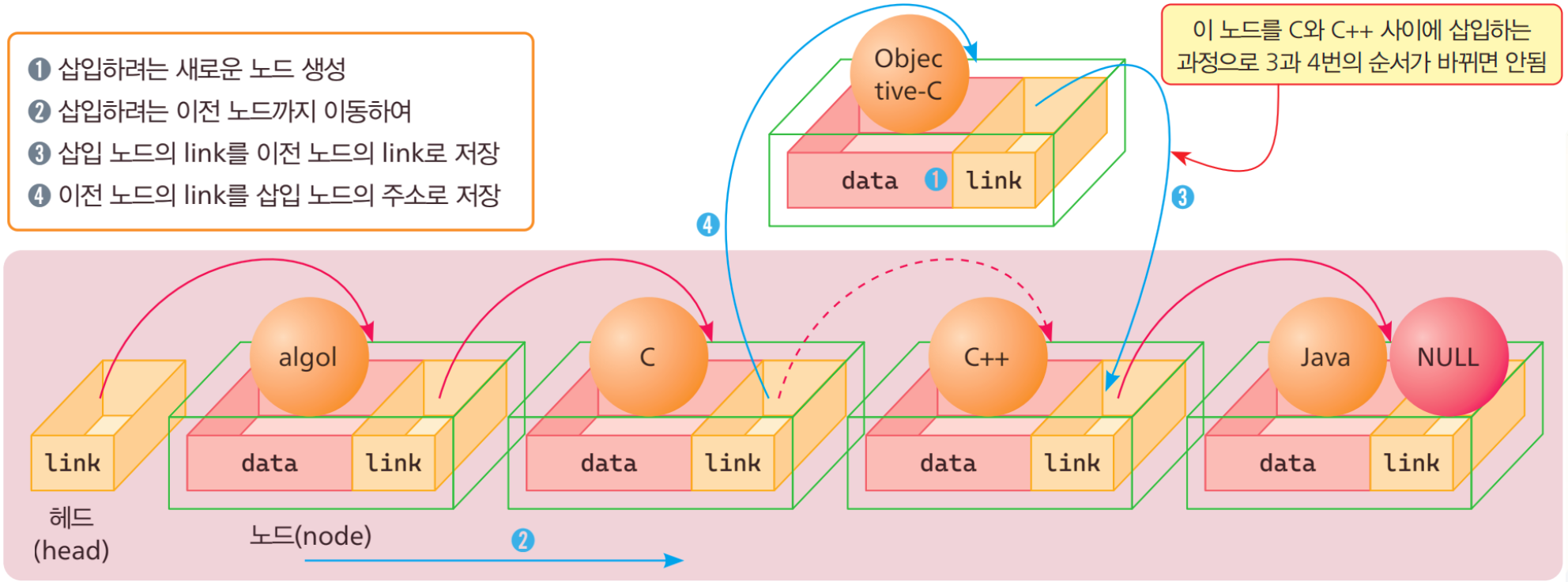
```
//❹ first 다음에 second를 연결
first->next = second;  // 구조체 *first가 다음 *second 구조체를 가리키도록 하는 문장
```



[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 710쪽

노드 삽입

❶ 삽입하려는 새로운 노드 생성
❷ 삽입하려는 이전 노드까지 이동하여
❸ 삽입 노드의 link를 이전 노드의 link로 저장
❹ 이전 노드의 link를 삽입 노드의 주소로 저장



[이미지] 강환수 외 2인 공저, Perfect C (3판), 인피니티북스, 2021, 719쪽

정 리 하 기

정리하기

- 포인터는 주소 값을 저장하는 변수이다.
- 간접연산자(indirection operator) *를 사용한
  *p는 피연산자인 p가 가리키는 변수 자체를 반환한다.
- 문자는 작은 따옴표, 문자열은 큰 따옴표를 사용해 각각 표현한다.
- 문자열의 마지막을 의미하는 NULL 문자 '₩0'가
  마지막에 저장되도록 한다.
- 여러 문자열은 문자 포인터 배열이나 2차원 문자 배열을 활용한다.
- 명령행 인자를 사용하기 위해서는
  두 개의 인자 argc와 argv를
  main(int argc, char * argv[])로 기술해 사용한다.

정리하기

- 구조체는 정수, 문자, 실수나 포인터 그리고
  이들의 배열 등을 묶어 하나의 자료형으로 이용하는 것이다.
- 공용체는 동일한 저장 장소에 공용으로 여러 자료형을 저장한다.
- 구조체와 공용체에서 접근연산자 .를 사용하여 멤버를 참조한다.
- 키워드 typedef로 이미 사용되는 자료 유형을
  다른 새로운 자료형 이름으로 재정의한다.
- 함수에서 매개변수로 값을 전달하는 방식은
  값의 전달(call by value) 방식과
  주소에 의한 전달(call by address) 방식으로 나뉜다.

정리하기

- 함수에서 배열을 매개변수로 사용하는 경우,
  배열의 크기도 알려줘야 한다.
- 함수 fopen( )으로 파일 스트림 열고 fclose( )로
  파일 스트림을 닫는다.
- 함수 fprintf( ) 등으로 텍스트 파일을 작성하고
  fwrite( )로 이진파일을 작성한다.
- 함수 malloc( )으로 힙(heap) 영역에 저장공간을 확보하며
  사용 후 함수 free( )로 저장공간을 해제한다.
- 자기참조 구초제는 멤버 중의 하나가
  자기 자신의 구조체 포인터 변수를 갖는 구조체이다.

학우여러분

감사합니다.