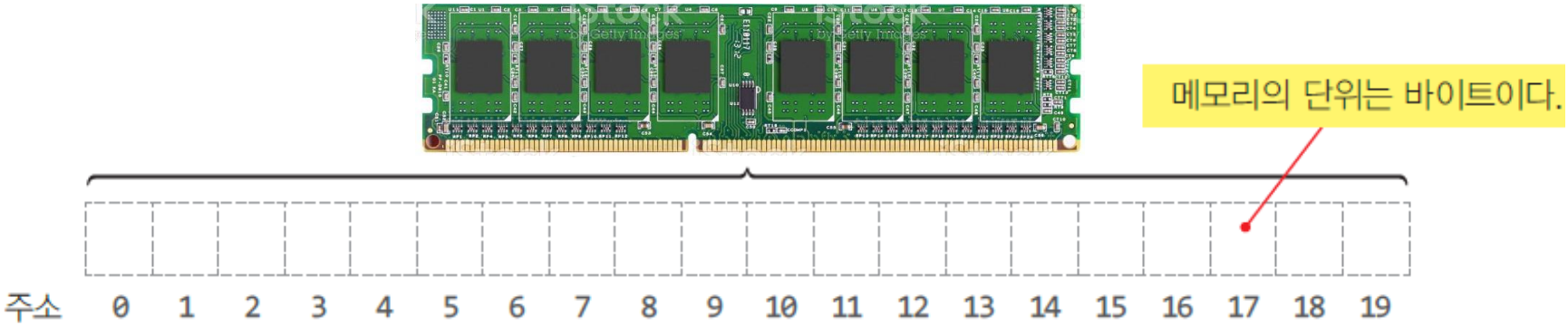
12장 포인터의 이해

차례

1. 포인터란 무엇인가?
2. 포인터와 관련 있는 연산자: &연산자, *연산자

메모리의 구조

- 메모리는 프로그램이나 데이터를 저장하는 장치
- 메모리는 1바이트(82페이지 참고) 단위로 고유한 주소가 붙어있음
  - 주소 : 메모리 공간의 위치를 나타내는 숫자
  - 첫번째 바이트의 주소는 0번지이고 바이트마다 1씩 증가
- CPU는 메모리의 주소를 이용하여 원하는 공간에 데이터를 저장

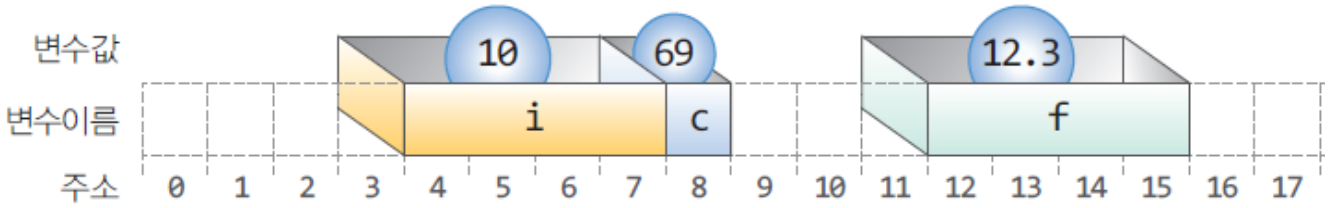


20바이트 메모리 -> 0~19까지 주소가 존재
16GB 메모리 -> $16x2^{30}$ Byte($16x2^{30}x8$ bit) -> 0~ ($16x2^{30}$-1)

변수와 메모리

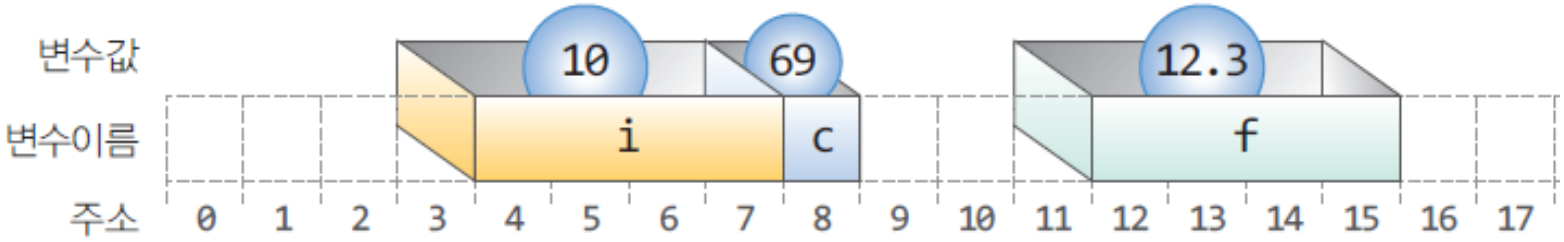
- 프로그램에서 선언한 모든 변수는 메모리에 저장됨
- 변수의 자료형에 따라 차지하는 메모리 공간의 크기가 다름
- char: 1바이트, int: 4바이트, float: 4바이트, double: 8바이트

```
int main(void)
{
        int i = 10;
        char c = 69;
        float f = 12.3;
}
```



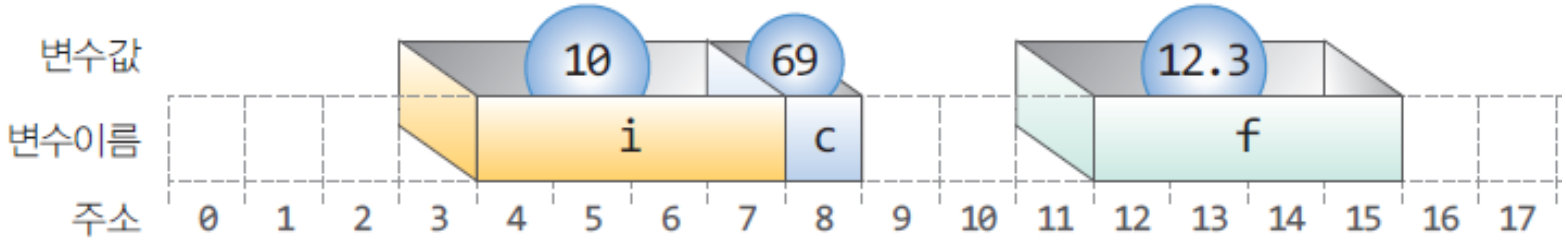
변수의 주소

- 변수가 차지하고 있는 공간의 시작 번지(가장 작은 주소)
  - 변수 i의 주소 : 4~7번지-> 4번지
  - 변수 c의 주소 : 8번지
  - 변수 f의 주소 : 12~15번지 -> 12번지
- 변수가 저장되는 주소는 컴퓨터에 의해 자동으로 결정됨 -> 프로그래머가 원하는 주소에 마음대로 저장할 수 없음



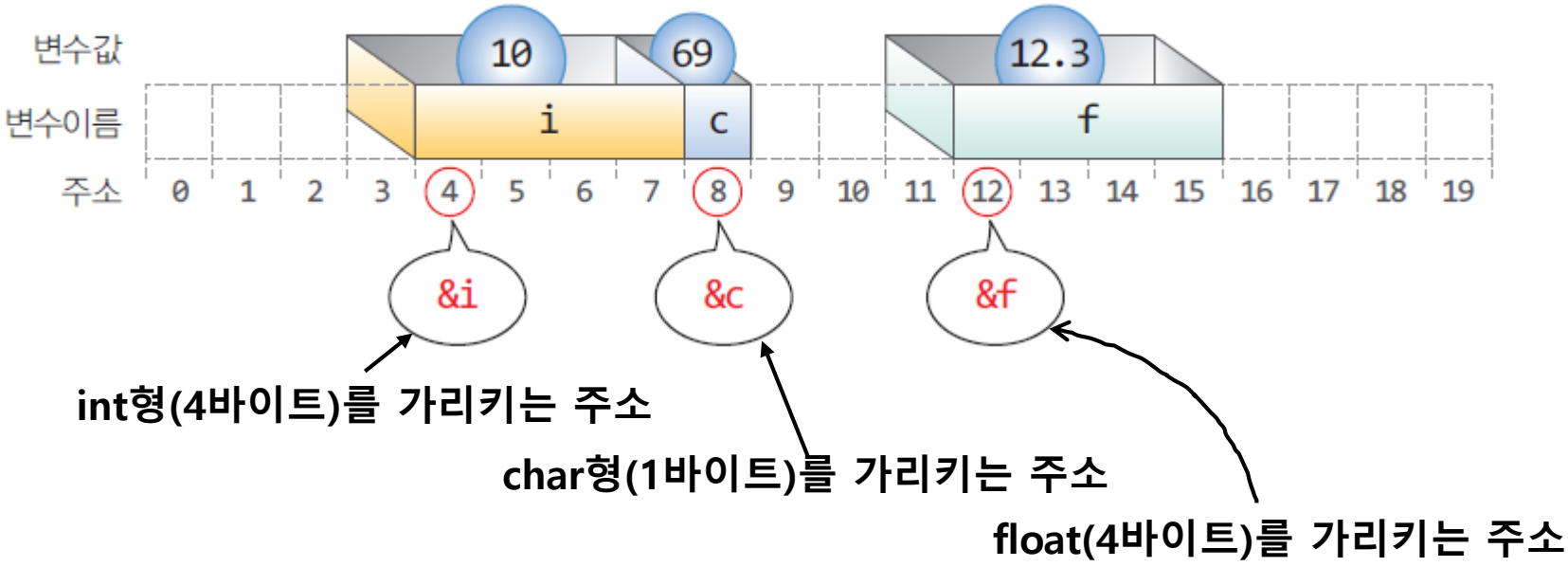
변수의 주소가 가리키는 자료형

- 변수의 주소는 가리키는 자료형이 모두 다름
  - 변수 i의 주소 4번지 -> int형(4바이트)를 가리키는 주소
  - 변수 c의 주소 8번지 -> char형(1바이트)을 가리키는 주소
  - 변수 f의 주소 12번지 -> float(4바이트)을 가리키는 주소



변수의 주소를 구하는 방법 : 주소연산자 &

- 주소연산자 &(앰퍼샌드) : 변수의 주소를 계산하는 연산자
- 변수 a의 주소 : **&a** -> 변수 a가 저장된 메모리의 시작 주소
- 변수의 주소는 가리키는 자료형이 정해져 있음



주소의 자료형

- 주소의 자료형 -> **(주소가 가리키는 변수의 자료형)***

변수선언	주소	자료형	설명
char c;	&c	char*	char를 가리키는 주소형
int i;	&i	int*	int를 가리키는 주소형
float f;	&f	float*	float를 가리키는 주소형
double d;	&d	double*	double을 가리키는 주소형

예제: 변수의 주소를 출력

- 주소를 출력할 때 형식지정자 : %u(부호없는 정수), %p(16진수)
- 2가지 형식 지정자로 모두 실행해볼 것

```
#include<stdio.h>
int main(void)
{
    int i = 10;
    char c = 69;
    float f = 12.3;
    printf("i의 주소: %u₩n", &i);      // 변수 i의 주소 출력
    printf("c의 주소: %u₩n", &c);      // 변수 c의 주소 출력
    printf("f의 주소: %u₩n", &f);      // 변수 f의 주소 출력
    return 0;
}
```

```
i의 주소: 1245024
c의 주소: 1245015
f의 주소: 1245000
```

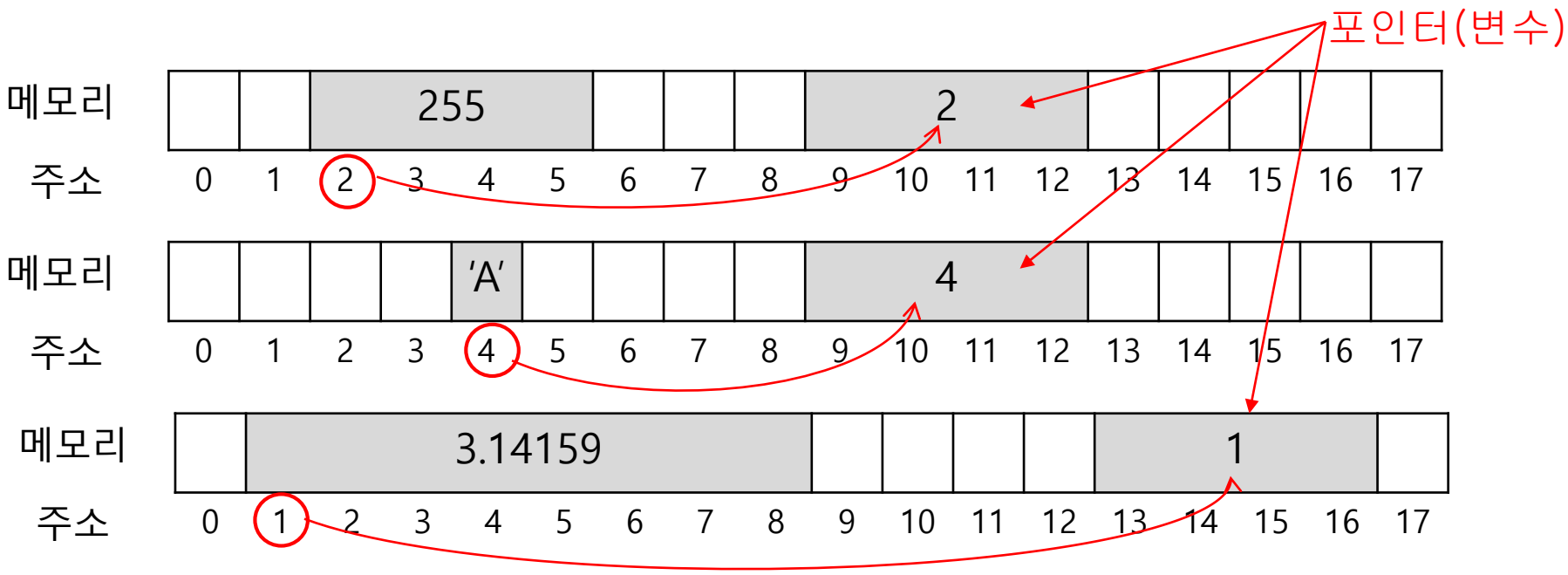
주소를 저장하는 변수

- 주소는 문자, 정수, 실수와는 다른 새로운 자료형 -> 기존 변수에 저장이 불가능하므로 주소를 저장할 수 있는 새로운 변수 필요

자료형	예	변수
정수형	10, -5	int a;
실수형	3.14, -5.2	double a;
문자형	'a', 'C'	char ch;
주소형	100번지, 200번지	?

포인터란?

- 포인터(pointer): 주소를 저장하기 위한 변수(주소형 변수)
- 일반 변수는 데이터 자체(정수, 실수, 문자)를 저장하는 변수
- 포인터는 데이터가 저장된 변수의 주소를 저장하는 변수



포인터의 선언


```
자료형*  변수명;
자료형 * 변수명;
자료형  *변수명;
```

- 자료형 : 포인터에 저장할 주소가 가리키는 자료형
- * : 포인터 변수임을 의미(곱셈기호 아님)
- 자료형* : 포인터의 (자료)형
- 변수명 : 포인터 변수의 이름

```
int* pi ;          // int 형을 가리키는 주소를 저장하는 포인터
char* pc;          // char 형을 가리키는 주소를 저장하는 포인터
float* pf;         // float 형을 가리키는 주소를 저장하는 포인터
double* pd;        // double 형을 가리키는 주소를 저장하는 포인터
```

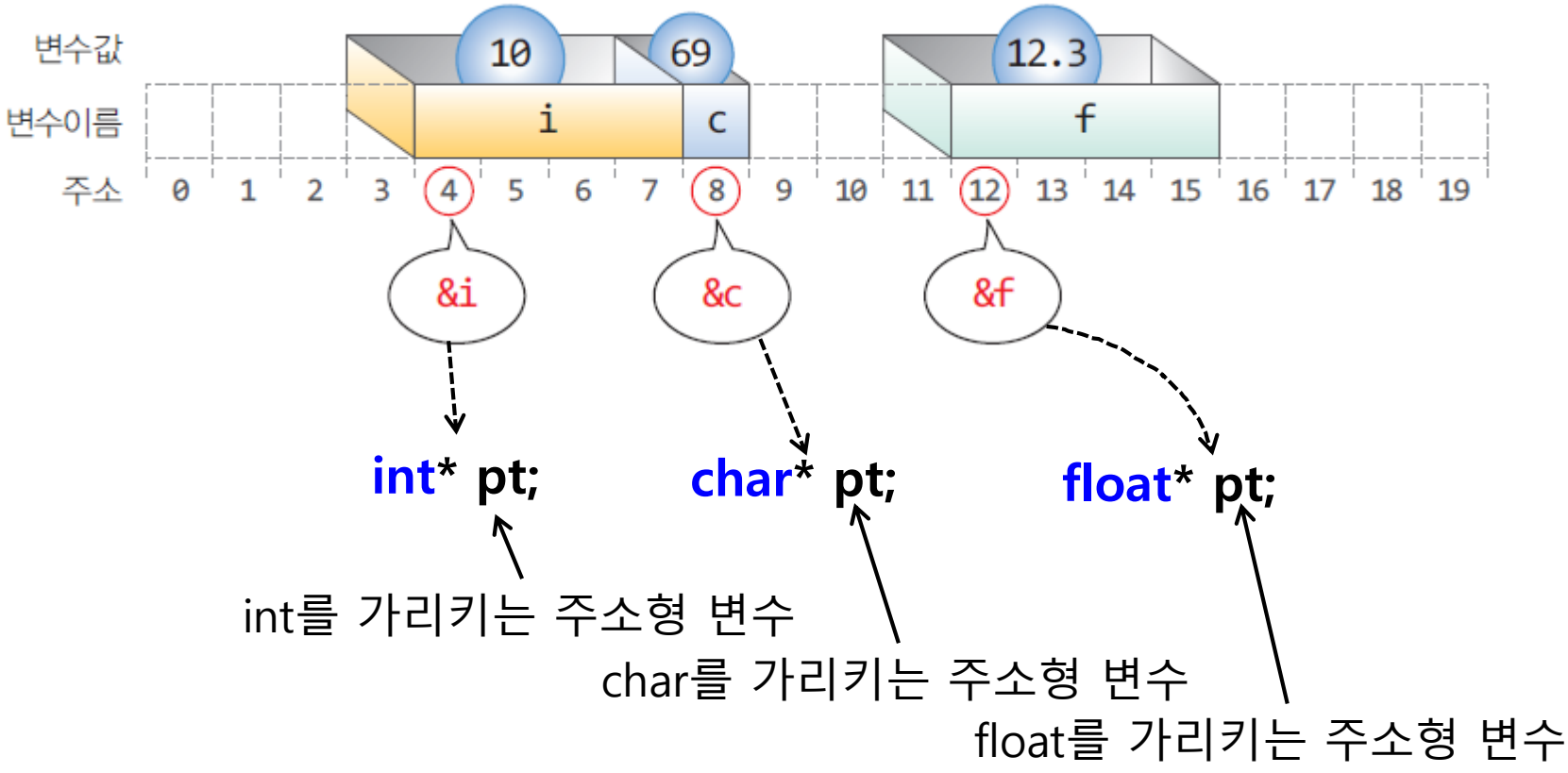
포인터의 선언

- 포인터의 (자료)형 : 포인터에 저장될 주소의 자료형(자료형*)

```
int* pi ;        -> pi의 자료형 int*, pi는 int*형 포인터
char* pc;        -> pc의 자료형 char*, pc는 char*형 포인터
float* pf;       -> pf의 자료형 float*, pf는 float*형 포인터
double* pd;      -> pd의 자료형 double*, pd는 double*형 포인터
```

포인터의 선언

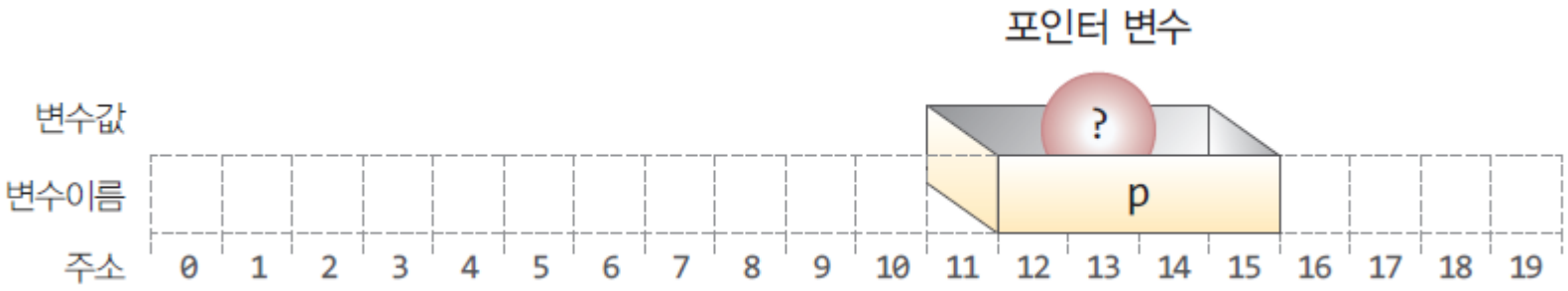
- 포인터 선언시 자료형의 의미 : 주소가 가리키는 공간에 저장된 데이터의 자료형을 의미



포인터의 선언

- 포인터도 변수이므로 선언하면 4 or 8바이트 메모리 공간에 할당되고 초기값이 없으면 쓰레기값을 가짐
  - 32비트시스템(x86)-> 주소값 크기 4바이트->포인터 변수의 크기 4바이트
  - 64비트시스템(x64)-> 주소값 크기 8바이트->포인터 변수의 크기 8바이트

```
int* pi ;          //포인터변수 pi는 4 or 8바이트에 할당
char* pc;          //포인터변수 pc는 4 or 8바이트에 할당
double* pd;        //포인터변수 pd는 4 or 8바이트에 할당
```



예제 : 포인터의 메모리 크기


```
#include <stdio.h>
int main(void)
{
        int* pi=NULL ;
        char* pc=NULL;
        double* pd=NULL;
        printf("%d₩n", pi);              // 0
        printf("%d₩n", pc);              // 0
        printf("%d₩n", pd);              // 0
        printf("%d₩n", sizeof(pi));      // ?
        printf("%d₩n", sizeof(pc));      // ?
        printf("%d₩n", sizeof(pd));      // ?
}
```

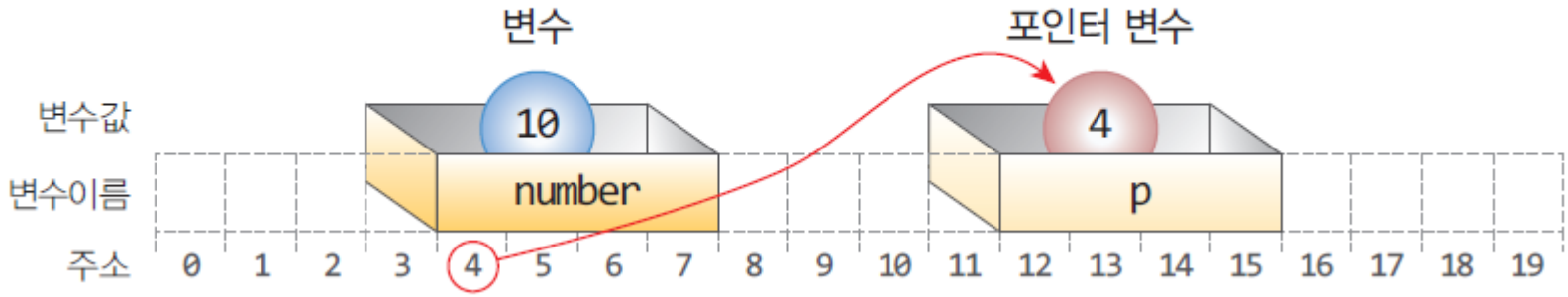
64비트시스템(x64)

32비트시스템(x86)

포인터에 주소 저장

- 주소연산자를 이용하여 변수의 주소를 구한 후 포인터에 대입

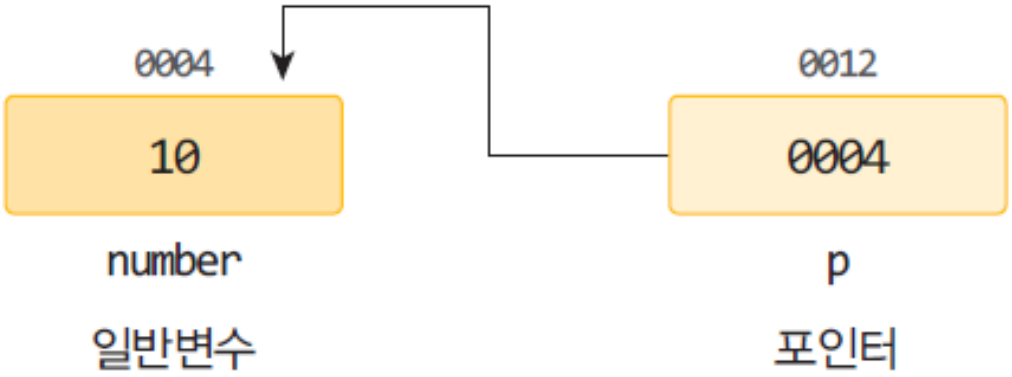
```
int number = 10;          // 정수형 변수 number 선언
int* p;                   // 포인터 변수 p 선언
p = &number;              // 변수 number의 주소를 포인터 p로 대입
```



포인터에 주소 저장

- "**포인터 p가 변수 number를 가리킨다**" 라고 표현함

```
int number = 10;          // 정수형 변수 number 선언
int* p;                   // 포인터 변수 p 선언
p = &number;              // 변수 number의 주소를 포인터 p로 대입
```



포인터에 주소 저장


```
char c = 'A';                // 문자형 변수 c
int i = 36;                  // 실수형 변수 f
double d = 3.141592;         // 실수형 변수 d

char* pc = &c;               // char를 가리키는 포인터 pc
int* pi = &i;                // float를 가리키는 포인터 pf
double* pd = &d;             // double를 가리키는 포인터 pd

pi = &c;                     // 컴파일 오류
pi = &d;                     // 컴파일 오류
```

실습과제 1

- 다음에서 pt가 가리키는 데이터의 자료형은 무엇인가?
- 다음에서 pt에 저장될 주소의 자료형은 무엇인가?
- 다음에서 pt의 자료형은 무엇인가?
- 다음 처럼 포인터를 선언할 때 double의 의미를 설명하라.
- 포인터 변수 pt가 메모리에 할당될 때 메모리 크기는?

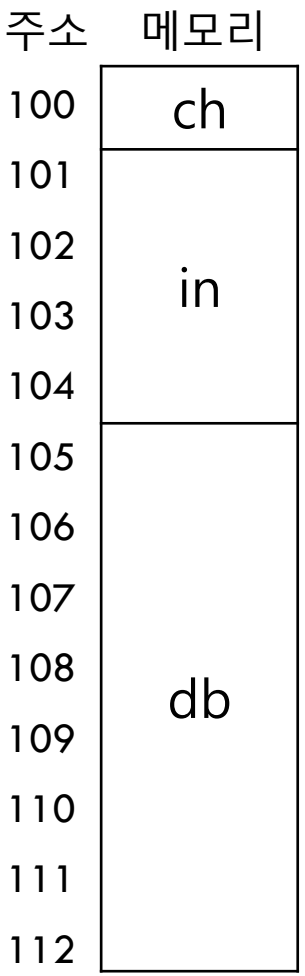
```
double * pt;
```

실습과제2

- 아래의 변수가 우측 그림처럼 메모리가 할당될 때 다음 표의 빈칸을 채우시오.

```
char ch = 'A';
int in = 10;
double db = 3.4;
```

수식	결과값	결과값의 자료형
&ch		
&in		
&db		



실습과제3

- 다음 코드처럼 변수 a, b, c를 선언하고 각 변수의 주소를 주소연산자(&)를 이용하여 화면에 다음 처럼 출력하라.

```
char a = 'A';
int b = 36;
double c = 3.141592;
```

char형 변수 a의 주소 : xxxxxx
int형 변수 b의 주소 : xxxxxx
double형 변수 c의 주소 : xxxxxx

과제제출방법

- 소스코드, 라인단위의 주석, 실행결과를 포함하는 pdf파일을 작성한 후 eclass 과제 게시판에 업로드, **반드시 하나의 pdf파일로 업로드할 것**
- 기한 : 과제 게시판에 마감시간 참조
- 실행결과를 캡쳐할 때 글자를 알아보기 쉽게 확대해서 캡쳐할 것.
- 소스코드의 첫 부분은 아래처럼 제목,날짜,작성자(학번,이름)를 작성할 것

```
// ********************************************
//    제   목 : 정수 4개의 평균을 구하는 프로그램
//    날   짜 : 2023년 9월10일
//    작성자 : 15010101 홍길동
// ********************************************

// 소스코드 작성
```