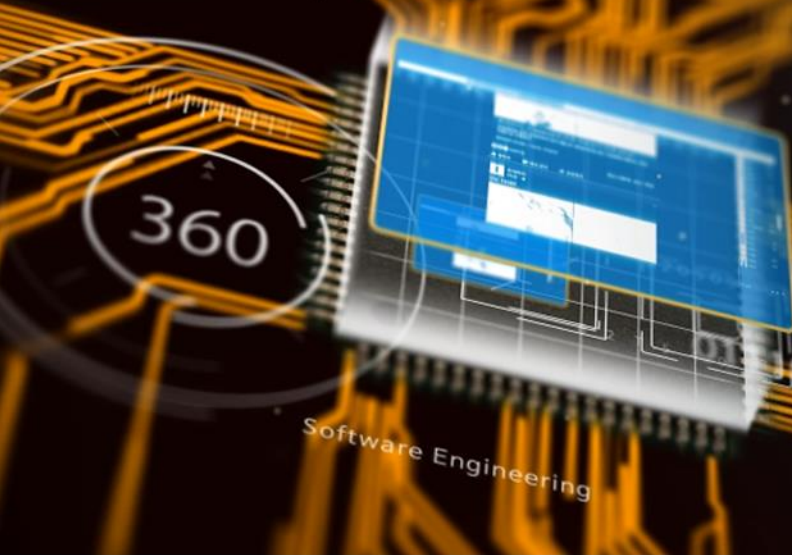


프로그래밍 언어 활용

1011101010001010101

part 1



포인터 기초



한국기술교육대학교
온라인평생교육원

학습내용


- 포인터 이해
- 포인터 이용

학습목표

- 포인터의 개념에 대해 설명할 수 있다.
- 포인터 변수를 이용하여 데이터 변수값을 참조할 수 있다.


포인터 이해

1 포인터 개념




포인터

특정 위치를 가리키는 주소 정보

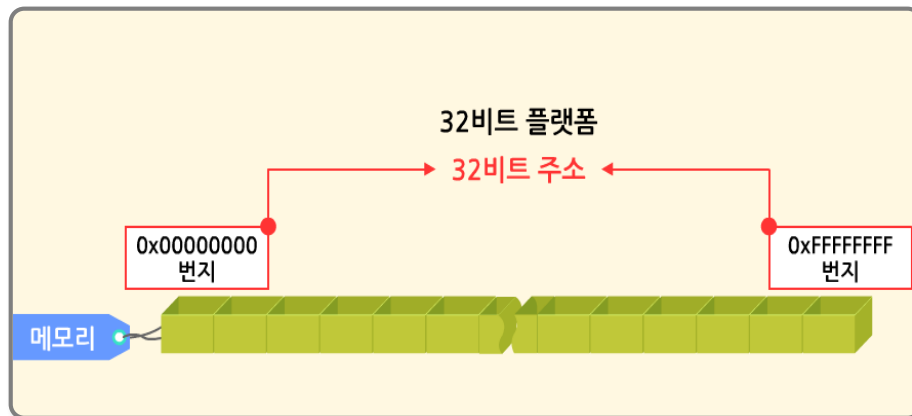


포인터 변수

포인터(주소)를 저장하는 변수



포인터 변수의 크기(주소의 크기)는 일반적으로 4바이트이지만 시스템에 따라 다름



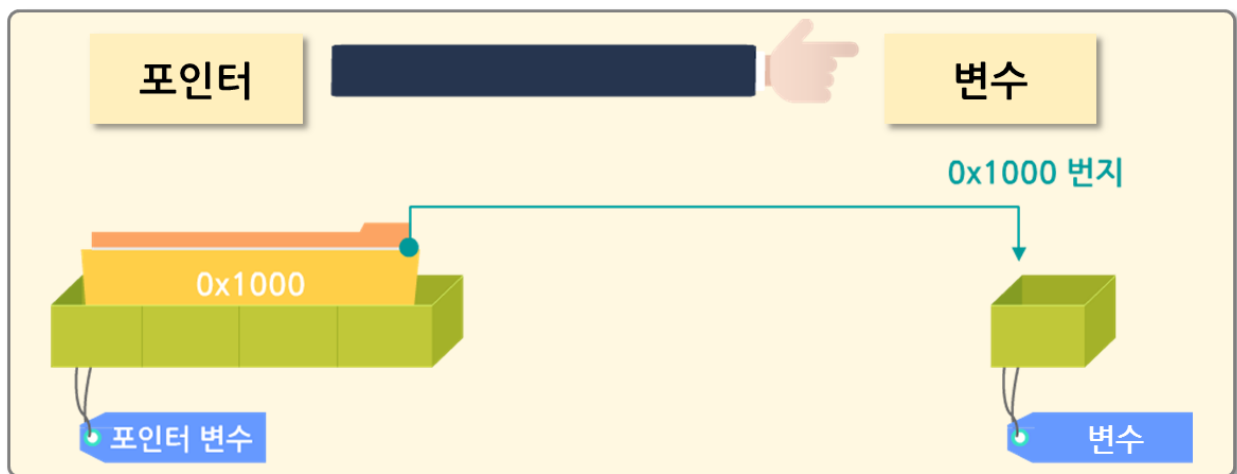
0x00	
0x01	
0x02	
int a; 0x03	a
0x04	
0x05	
0x06	
0x07	
0x08	
0x09	
0x0A	
0x0B	
0x0C	
0x0D	

포인터 이해

1 포인터 개념



포인터 변수의 역할은 다른 변수를 가리키는 변수



2 포인터 변수



포인터 변수의 데이터 타입은 포인팅하는 변수의 데이터 타입과 동일한 타입으로 선언

`char*`

char형 변수의
주소 저장

`int*`

int형 변수의
주소 저장

`double*`

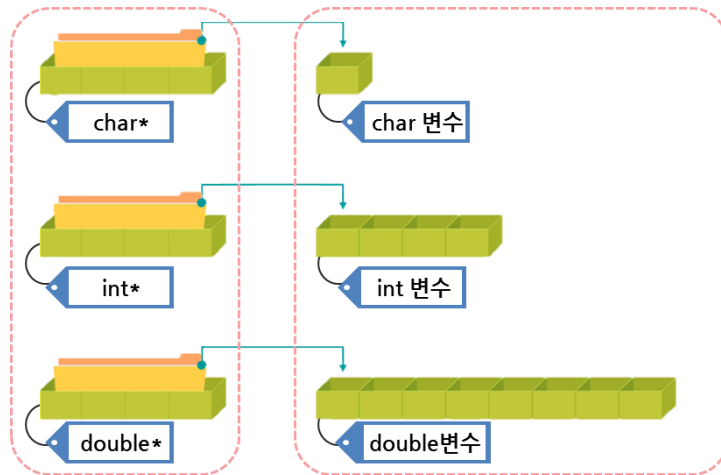
double형 변수의
주소 저장

포인터 이해

2 포인터 변수

데이터타입 * 포인터 변수명 ;

```
char* a;  
int *b;
```



포인터 변수의 크기는
모두 4바이트임

포인터 변수가 가리키는
변수의 크기는 서로 다름

포인터 이해

2 포인터 변수

sizeof() 연산자

int sizeof(int) : 매개변수에 기술한 공간의 크기를 **정수값으로 반환함**

```
int a;  
char b;  
double c;  
int *pa;  
char *pb;  
double *pc;
```

```
sizeof(a);  
sizeof(b);  
sizeof( c );  
sizeof(pa);  
sizeof(pb);  
sizeof(pc);
```

포인터 이해

3 포인터 초기화

&

주소 구하기 연산자

&데이터 변수

변수의 주소를 의미

*포인터 변수

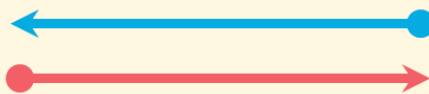
포인터 변수가 가리키는 주소의 값

int* 변수



x의 주소를 구함

&x



*p

p가 가리키는
변수에 접근함

int 변수



포인터 이해

3 포인터 초기화

```
int a=5;
```

```
int *pa=&a;
```

정수형 변수 a의 시작 주소

```
int *pa;  
pa = &a;
```

0x00	
0x01	
0x02	
0x03	
0x04	
0x05	
0x06	5
0x07	
0x08	
0x09	
0x0A	
0x0B	
0x0C	
0x0D	0x03

a

포인터 이용

1 포인터 다루기

```
#include <stdio.h>
int main()
{
    int a=9;
    int *pa = &a;

    printf("변수 a의 값은 %d 이고 a의 주소는 %p 이다.", a, &a);
    printf("포인터 변수 pa의 값은 %p 이고 pa가 가리키는 곳의 값은 %d 이다", pa,
*pa );
    return 0;
}
```

```
#include <stdio.h>
int main()
{
    int a=9;
    int *pa = &a;
    printf("변수 a의 값은 %d 이고 a의 주소는 %p 이다.", a, &a);
    printf("포인터 변수 pa의 값은 %p 이고 pa가 가리키는 곳의 값은 %d 이다", pa,
*pa );
    *pa = 12;
    printf( "a=%d", a);
    return 0;
}
```

포인터 이용

1 포인터 다루기

```
#include <stdio.h>
int main()
{
    int a,b=4;
    char ch='k' ; char *pch=&ch;
    int *pa = &a;  *pa=7;
    printf("a=%d  *pa = %d \n", a, *pa);
    printf("ch=%c  *pch = %c \n", ch, *pch);
    pa = &b;
    printf("b=%d \n", b);
    *pa = 15;
    printf("b=%d \n", b);
    return 0;
}
```

포인터 이용

2 이중 포인터

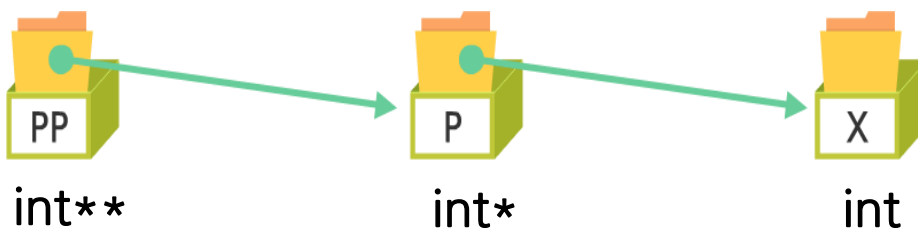


포인터 변수의 주소를 저장하는 포인터 변수

```
int x;
```

```
int *p = &x;
```

```
int **pp = &p;      pp는 이중 포인터
```



이중 포인터가 가리키는 포인터를 이용해서 변수에 접근하려면 **처럼 두 번 간접 참조를 해야 함

```
**pp = 10;
```

*pp는 p가 되므로, 다시 p가 가리키는 x에 접근하려면 *가 하나 더 필요함

포인터 이용

3 포인터 오류

1 포인터도 변수이므로 반드시 초기화해야 함

2 포인터 변수를 초기화하지 않고 사용하면 실행 에러가 발생함

`int *p;` ●———— p는 초기화되지 않았으므로 쓰레기 값을 가짐
`*p = 10;` ●———— 실행 에러가 발생함

3 널 포인터: 포인터가 다른 변수를 가리키지 않을 때는 NULL(0)로 초기화함

`int *p = NULL;` ●———— p를 널 포인터로 초기화함
 p는 아직 다른 변수를 가리키지 않음

4 포인터를 안전하게 사용하려면 우선 포인터가 널 포인터인지를 검사

`if(p != NULL)` ●———— p가 널 포인터가 아닌지 확인 후에 사용
`*p = 10;`

`if(p)` ●———— p가 널 포인터가 아닌지 확인
`*p = 10;`

포인터 이용

3 포인터 오류

5 포인터 변수의 데이터형이 반드시 포인터 변수가 가리키는 변수의 데이터형과 일치해야 함

```
short a;
```

```
int *p = &a; ●———— 컴파일 경고가 발생
```

```
*p = 10; ●———— 컴파일 경고를 무시하고 실행하면, 실행에러가 발생
```

학습정리

1. 포인터 이해

- 포인터 : 다른 변수의 주소를 저장하는 변수
- 포인터의 선언 : 데이터형*변수명이 필요함
예) `int * p;`
- 포인터의 사용 : 변수의 주소를 구할 때는 주소 구하기 연산자 `&`를 이용하고, 포인터가 가리키는 변수에 접근할 때는 간접 참조 연산자 `*`를 이용함
예) `int *p = &x;`
`*p = 10;`

2. 포인터 이용

- 포인터 사용 시 주의사항
 - 포인터 변수는 포인터가 가리키는 변수의 데이터형과 일치하도록 선언해야 함
 - 잘못된 포인터를 사용하는 것은 위험하므로, 포인터가 가리키는 변수가 없을 때는 NULL을 저장함