

# C 기초 프로그래밍 교육

---

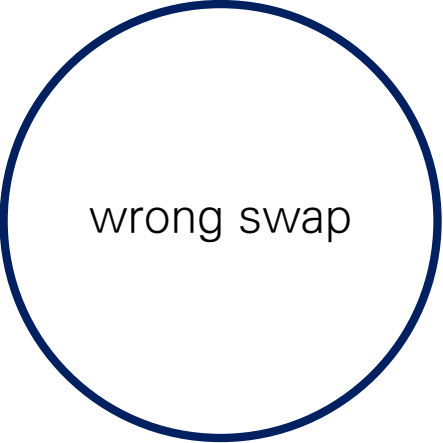
Week 5. pointer application

교육 구성

# INDEX

- 
1. 과제 풀이
  2. 리뷰
  3. 배열과 포인터
  4. 함수 포인터
-

## 1. 과제 풀이



wrong swap

# 과제 1 - wrong swap

```
void wrong_swap(int x, int y)
{
    int temp = x;
    x = y;
    y = temp;
}

int main()
{
    int a = 3;
    int b = 4;

    printf("before swap %d %d\n", a, b);

    wrong_swap(a, b);
    printf("after swap %d %d\n", a, b);
}
```

```
minibeef@argos-edu:~/cedu/week4/hw$ ./wrong_swap
before swap 3 4
after swap 3 4
```

두 값의 위치를 바꾸는 함수

과제 풀이

# 과제 1 - wrong swap

```
minibeef@argos-edu:~/cedu/week4/hw$ ./wrong_swap  
before swap 3 4  
after swap 4 3
```

포인터를 사용하여 위와 같이 출력되도록 만들기

# 과제 1 - wrong swap

```
void wrong_swap(int* x, int* y)
{
    int temp = *x;
    *x = *y;
    *y = temp;
}

int main()
{
    int a = 3;
    int b = 4;

    printf("before swap %d %d\n", a, b);


    wrong_swap(&a, &b);
    printf("after swap %d %d\n", a, b);
}
```

포인터를 사용하여 “원본을 수정”

```
minibeef@argos-edu:~/cedu/week4/hw$ ./wrong_swap
before swap 3 4
after swap 4 3
```

〈결과〉

## 2. 리뷰



지금까지  
배운 것

리뷰

# 지금까지 배운 것

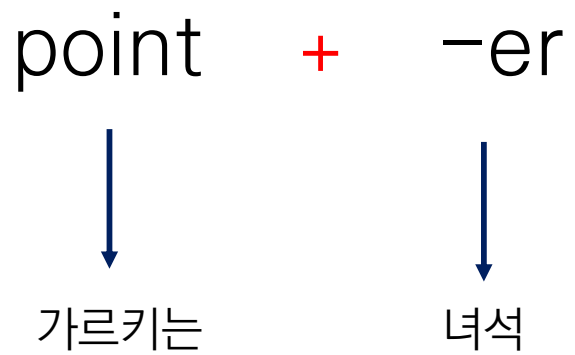




리뷰

# 포인터 다시 보기

point + -er



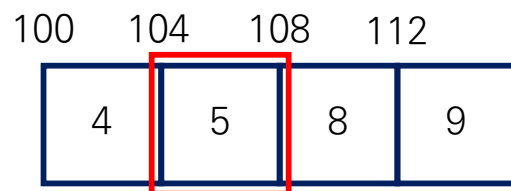
The diagram illustrates the word 'pointer' by breaking it down into its components. 'point' is followed by a red plus sign and '-er'. Below 'point' is a blue arrow pointing down to the Korean text '가르키는' (teaching). Below '-er' is a blue arrow pointing down to the Korean text '녀석' (家伙, referring to a person or thing).

가르키는      녀석

뭘 가르키느냐? -> 변수의 주소

# 포인터 다시 보기

“값”을 저장하는게 아닌 “주소”를 저장하는 변수 : 포인터 변수



포인터 변수

리뷰

# 포인터 다시 보기

선언

```
int a = 3;  
int* pa = &a;
```

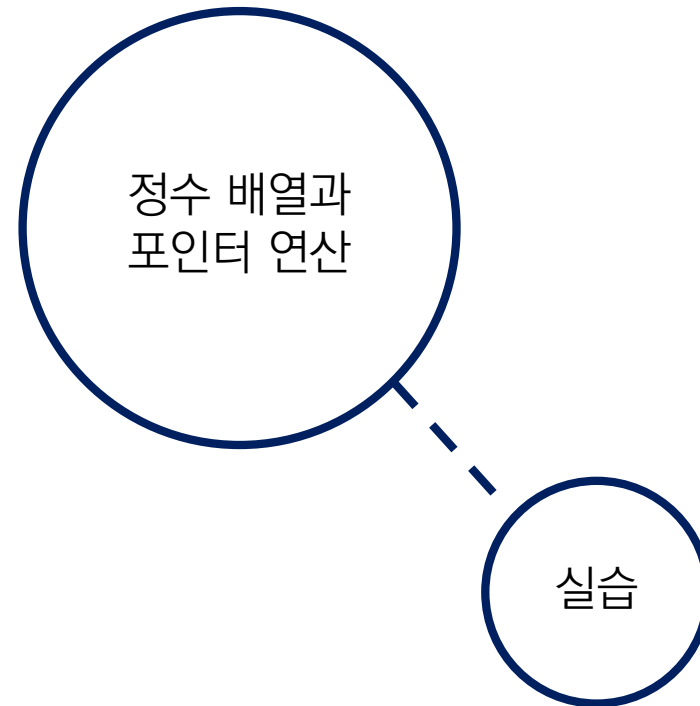
사용

```
printf("%d", *pa);
```

주소 얻어내기

&변수이름

### 3. 배열과 포인터



# 정수 배열과 포인터 연산

```
#include <stdio.h>

int main()
{
    int arr[3];
    arr[0] = 1;
    arr[1] = 2;
    arr[2] = 3;

    printf("%d %d %d\n", arr[0], arr[1], arr[2]);
}
```

```
#include <stdio.h>

int main()
{
    int arr[3];
    int* parr = &arr;
    *(parr + 0) = 1;
    *(parr + 1) = 2;
    *(parr + 2) = 3;

    printf("%d %d %d\n", *(parr + 0), *(parr + 1), *(parr + 2));
}
```

두 코드의 차이가 뭘까?

# 정수 배열과 포인터 연산

```
minibee@argos-edu:~/cedu/week5$ ./ex1  
1 2 3  
minibee@argos-edu:~/cedu/week5$ ./ex2  
1 2 3
```

결과는 같다..!

# 정수 배열과 포인터 연산

```
#include <stdio.h>

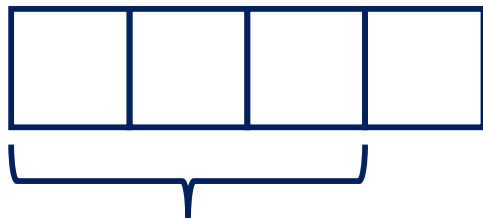
int main()
{
    int arr[3];
    arr[0] = 1;
    arr[1] = 2;
    arr[2] = 3;

    printf("%d %d %d\n", arr[0], arr[1], arr[2]);
}
```

```
#include <stdio.h>

int main()
{
    int arr[3];
    int* parr = &arr;
    *(parr + 0) = 1;
    *(parr + 1) = 2;
    *(parr + 2) = 3;

    printf("%d %d %d\n", *(parr + 0), *(parr + 1), *(parr + 2));
}
```



arr

# 정수 배열과 포인터 연산

```
#include <stdio.h>

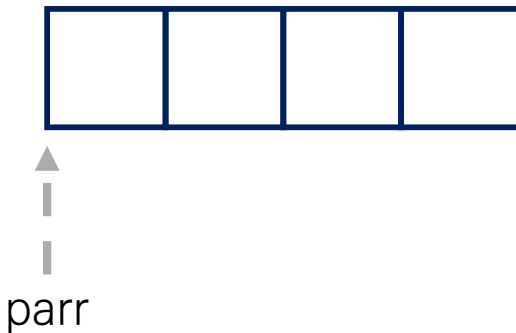
int main()
{
    int arr[3];
    arr[0] = 1;
    arr[1] = 2;
    arr[2] = 3;

    printf("%d %d %d\n", arr[0], arr[1], arr[2]);
}
```

```
#include <stdio.h>

int main()
{
    int arr[3];
    int* parr = &arr;
    *(parr + 0) = 1;
    *(parr + 1) = 2;
    *(parr + 2) = 3;

    printf("%d %d %d\n", *(parr + 0), *(parr + 1), *(parr + 2));
}
```





# 정수 배열과 포인터 연산

```
#include <stdio.h>

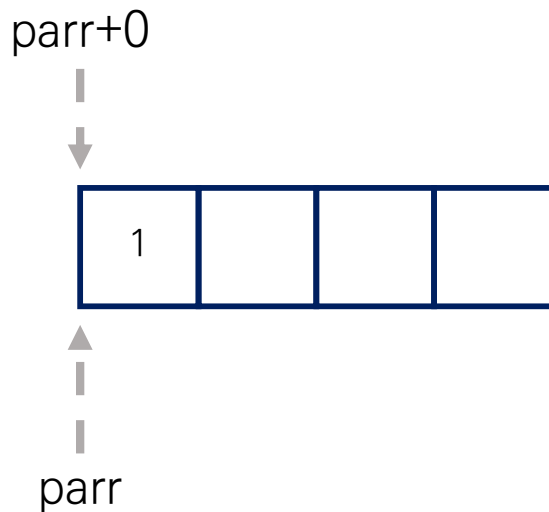
int main()
{
    int arr[3];
    arr[0] = 1;
    arr[1] = 2;
    arr[2] = 3;

    printf("%d %d %d\n", arr[0], arr[1], arr[2]);
}
```

```
#include <stdio.h>

int main()
{
    int arr[3];
    int* parr = &arr;
    *(parr + 0) = 1;
    *(parr + 1) = 2;
    *(parr + 2) = 3;

    printf("%d %d %d\n", *(parr + 0), *(parr + 1), *(parr + 2));
}
```



# 정수 배열과 포인터 연산

```
#include <stdio.h>

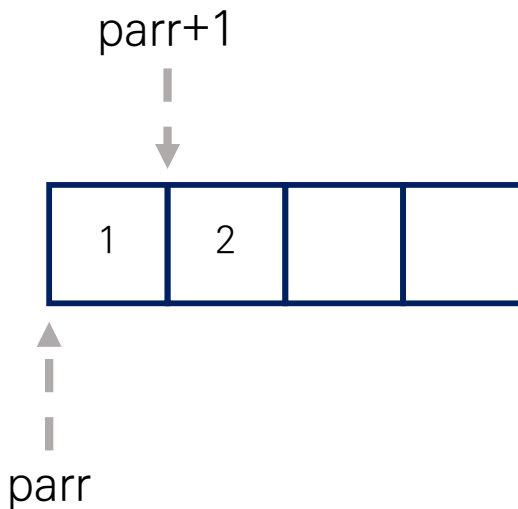
int main()
{
    int arr[3];
    arr[0] = 1;
    arr[1] = 2;
    arr[2] = 3;

    printf("%d %d %d\n", arr[0], arr[1], arr[2]);
}
```

```
#include <stdio.h>

int main()
{
    int arr[3];
    int* parr = &arr;
    *(parr + 0) = 1;
    *(parr + 1) = 2;
    *(parr + 2) = 3;

    printf("%d %d %d\n", *(parr + 0), *(parr + 1), *(parr + 2));
}
```



# 정수 배열과 포인터 연산

```
#include <stdio.h>

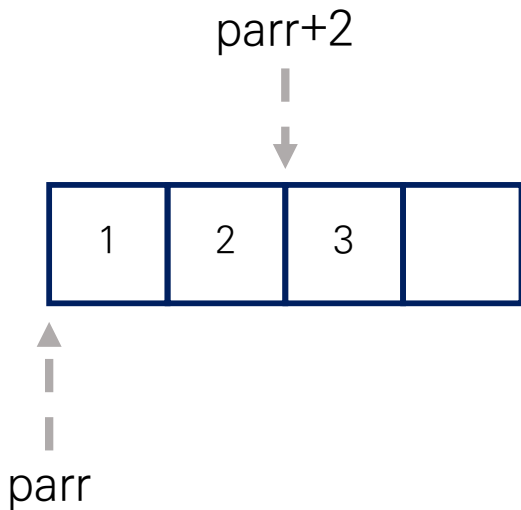
int main()
{
    int arr[3];
    arr[0] = 1;
    arr[1] = 2;
    arr[2] = 3;

    printf("%d %d %d\n", arr[0], arr[1], arr[2]);
}
```

```
#include <stdio.h>

int main()
{
    int arr[3];
    int* parr = &arr;
    *(parr + 0) = 1;
    *(parr + 1) = 2;
    *(parr + 2) = 3;

    printf("%d %d %d\n", *(parr + 0), *(parr + 1), *(parr + 2));
}
```



# 정수 배열과 포인터 연산

즉,

$$\text{arr}[n] == *(\text{parr} + n)$$

어떻게 이게 가능한가? → 포인터 연산

# 정수 배열과 포인터 연산

```
#include <stdio.h>

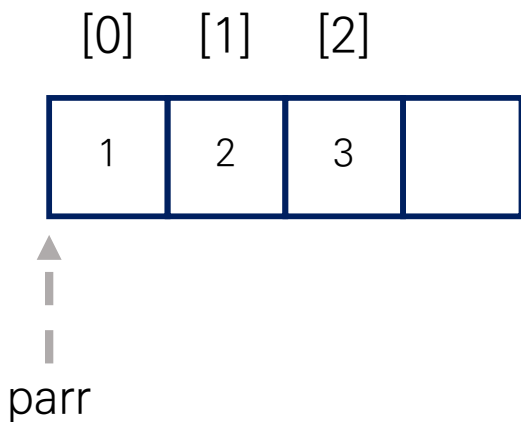
int main()
{
    int arr[3];
    arr[0] = 1;
    arr[1] = 2;
    arr[2] = 3;

    printf("%d %d %d\n", arr[0], arr[1], arr[2]);
}
```

```
#include <stdio.h>

int main()
{
    int arr[3];
    int* parr = &arr;
    *(parr + 0) = 1;
    *(parr + 1) = 2;
    *(parr + 2) = 3;

    printf("%d %d %d\n", *(parr + 0), *(parr + 1), *(parr + 2));
}
```



# 정수 배열과 포인터 연산

```
minibee@argos-edu:~/cedu/week5$ gcc -o ex2 ex2.c
ex2.c: In function 'main':
ex2.c:6:14: warning: initialization from incompatible pointer type [-Wincompatible-pointer-types]
  int* parr = &arr;
               ^
```

배열의 주소를 넣고 싶었는데 Warning이 뜬다.

# 정수 배열과 포인터 연산

```
minibee@argos-edu:~/cedu/week5$ gcc -o ex2 ex2.c
ex2.c: In function 'main':
ex2.c:6:14: warning: initialization from incompatible pointer type [-Wincompatible-pointer-types]
  int* parr = &arr;
               ^
```

배열의 주소를 넣고 싶었는데 Warning이 뜬다.



배열은 이름 그 자체가 주소를 대변한다.

# 정수 배열과 포인터 연산

```
#include <stdio.h>
int main()
{
    int arr[3];
    int* parr = arr;
    *(parr + 0) = 1;
    *(parr + 1) = 2;
    *(parr + 2) = 3;

    printf("%d %d %d\n", *(parr + 0), *(parr + 1), *(parr + 2));
}
```



# 포인터만 사용하기

```
minibee@argos-edu:~/cedu/week5$ ./prac1
3 4 5
합계는 12입니다.
```

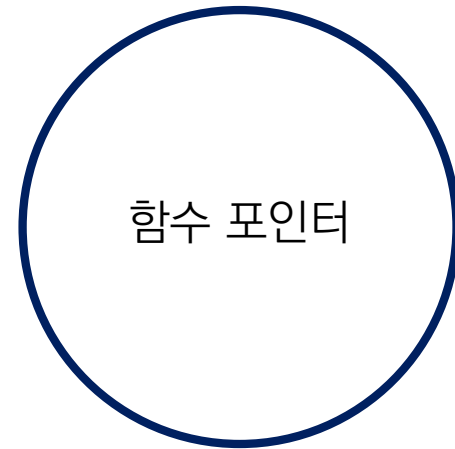
```
#include <stdio.h>

int main()
{
    int result = 0;
    int input[3];
    int* pointer = input;
    scanf("%d %d %d",                 );

    printf("합계는 %d입니다.\n", result);
}
```

## 4. 함수 포인터



| 함수에도 포인터가 있다

# 함수 포인터

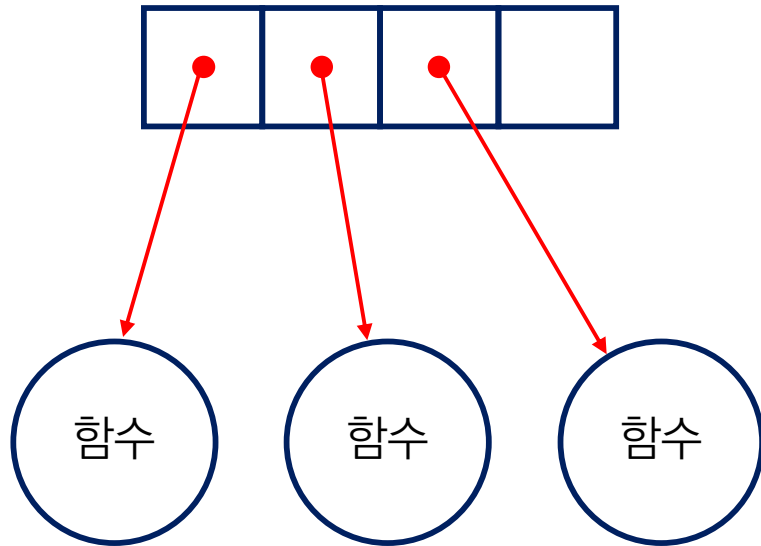
## 함수 포인터?

함수의 주소를 가르키는 변수

함수에도 포인터가 있다

# 함수 포인터

함수의 배열



함수를 인자로 갖는 함수

```
int function(정수, 함수, 함수);
```

함수에도 포인터가 있다

# 함수 포인터

함수에도 주소가 있을까?

```
#include <stdio.h>

int sum(int a, int b)
{
    return a + b;
}

int main()
{
    printf("sum address : %p\n", sum);
}
```

```
minibeef@argos-edu:~/cedu/week5$ ./ex3
sum address : 0x55d62953f64a
```

함수에도 포인터가 있다

# 함수 포인터

함수 포인터 만들기

반환값자료형 (\*함수포인터이름)(매개변수자료형1, 매개변수자료형2);

```
#include <stdio.h>

int exam1() { return 0; }
void exam2(int a, float b) { return; }

int main() {
    int (*ptr1)();
    ptr1 = exam1;

    void (*ptr2)(int, float);
    ptr2 = exam2;
}
```

함수에도 포인터가 있다

# 함수 포인터

함수 포인터 사용하기

함수 포인터를 마치 실제 함수인 것처럼 사용

```
#include <stdio.h>
int sum(int a, int b)
{
    return a + b;
}

int main()
{
    int (*ptr)(int, int) = sum;
    printf("%d\n", ptr(3, 4));
}
```

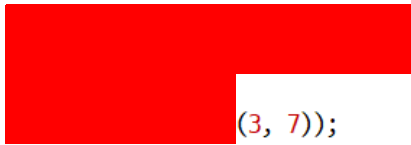
```
minibeef@argos-edu:~/cedu/week5$ ./ex5
7
```

# 함수 포인터만 사용하기

```
#include <stdio.h>

void print(int str) {
    printf("%d\n", str);
}

int sum(int a, int b)
{
    return a + b;
}

int main()
{
     (3, 7));
}
```

```
minibeef@cargos-edu:~/cedu/week5$ ./prac2
10
```



끝