

# 시스템 프로그래밍을 위한 C언어

## Function 포인터의 활용 및 필요성

현대자동차 입문교육  
박대진 교수

# Pointer to Function

<Pointer to Function>

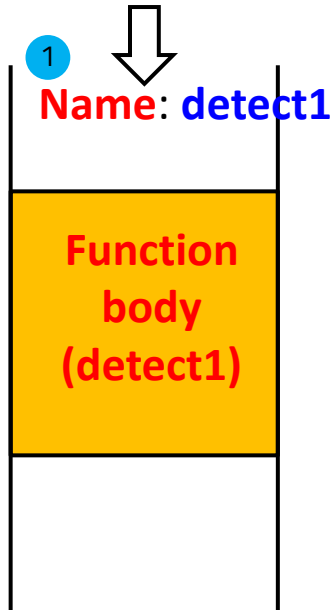
```
int (*detect)(int, char);
```

<Function to Body>

```
int detect1(int a, char b) {  
    // do something  
}
```

3 **detect** = detect1;  
detect(1,3);

2 **Address:** 0x88779D  
**detect(1, 3);**



*Function is also allocated in memory*

Function name is  
itself start address  
to function body.  
**detect1** is 0x88779D

# Replacing Function Start Address

<Pointer to Function>

```
int (*detect)(int, char);
```

```
detect = detect1;
```

```
detect(1,2);
```

```
detect = detect2;
```

```
detect(3,4);
```



detect is not pointing  
to function **detect2**  
at 0x8877CD

<Function to Body>

```
int detect1(int a, char b) {  
    // do something  
}
```

```
int detect2(int a, char b) {  
    // do something  
}
```

Address: 0x88779D

Function  
(detect1)

Address: 0x8877CD

Function  
(detect2)

# Reconfiguring Functions using Function Pointer Redirection

```
void detect_v1(int a, char b) {  
    // do something  
    printf("detect_v1() is activated\n");  
}  
  
void detect_v2(int a, char b) {  
    // do something  
    printf("detect_v2() is activated\n");  
}  
  
void (*detect)(int, char) ;
```

```
void main() {  
    int k=10;  
    char c=12;  
    int cond=1;  
    if(cond==1)  
        detect = detect_v2;  
    else  
        detect = detect_v1;  
    detect(10, 12);  
}
```