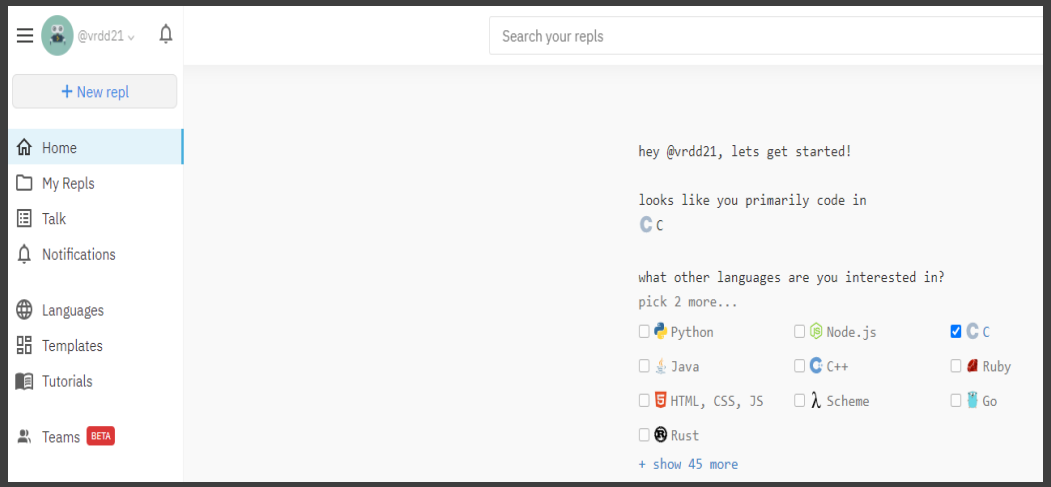
프로그래밍 연습

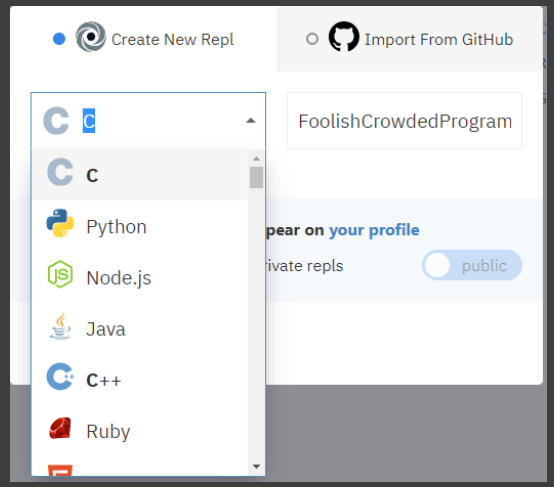
실습 week 9

실습환경

Repl.it

- 위 사이트에 접속하여 구글 계정등으로 로그인을 합니다.
- 처음 이용하는 경우 + New repl을 누르시고 오른쪽 이미지와 같이 C를 선택하세요
- 이미 만든 것은 My Repls를 통해 확인할 수 있습니다.





과제 풀이

실습

실습 1

2 개의 수를 입력받아 입력된 수가 같을 경우 " 같은 숫자입니다 ." 를 출력하고 종료하고 , 다를 경우 덧셈과 뺄셈 결과를 출력하시오 .

```
#include <stdio.h>

int get_checkEqual_add_subtract(int x, int y,
                int *ret_add, int *ret_subtract);

int main()
{
        int a=0,b=0,c=0;

        printf("2 개의 숫자를 입력하세요 : ");
        scanf("%d%d", &a, &b);

        c = get_checkEqual_add_subtract(a,b,&a,&b);

        if(c)
                printf(" 같은 숫자입니다 .₩n");
        else {
                printf(" 덧셈 : %d₩n", a);
                printf(" 뺄셈 : %d₩n", b);
        }

        return 0;
}
```

```
int get_checkEqual_add_subtract(int x, int y,
                int *ret_add, int *ret_subtract)
{
        if(x == y)
                return 1;
        else {
                *ret_add = x+y;
                *ret_subtract = x-y;
                return 0;
        }
}
```

[실행 결과]

```
~/week8$ ./ex8-1
2 개의 숫자를 입력하세요 : 3 5
덧셈 : 8
뺄셈 : -2
~/week8$ ./ex8-1
2 개의 숫자를 입력하세요 : 3 3
같은 숫자입니다 .
```

실습 1

➢ 코드분석

```
int a=0,b=0,c=0;
```

- 입력 및 결과 저장을 위한 변수 a, b 를 선언하고 ,
  변수 a, b 의 동일 여부 비교 결과 저장을 위한 변수 c 를 선언한다 .

```
c = get_checkEqual_add_subtract(a,b,&a,&b);
```

- 함수에 넘기는 총 4 개의 인자 중 처음 2 개는 call-by-value,
  나머지는 call-by-reference 로 넘긴다 .

```
int get_checkEqual_add_subtract(int x, int y, int *ret_add, int *ret_subtract)
```

- get_checkEqual_add_subtract() 함수 내 변수 x, y, ret_add, ret_subtract 에는
  아래의 값이 할당된다 .

  변수 x <- main() 함수 내 변수 a 의 값
  변수 y <- main() 함수 내 변수 b 의 값
  변수 ret_add <- main() 함수 내 변수 a 의 주소
  변수 ret_subtract <- main() 함수 내 변수 b 의 주소

실습 1

➢ 코드분석

```
if(x == y)
        return 1;
else {
        *ret_add = x+y;
        *ret_subtract = x-y;
         return 0;
}
```

- 인자로 넘어온 입력 값 2 개를 비교하여 같으면 1 을 return
- 다르면 더해서 ret_add, 빼서 ret_subtract 에 할당하고 0 을 return 한다 .

```
c = get_checkEqual_add_subtract(a,b,&a,&b);

if(c)
        printf(" 같은 숫자입니다 .₩n");
else {
        printf(" 덧셈 : %d₩n", a);
        printf(" 뺄셈 : %d₩n", b);
}
```

- main() 함수의 변수 a 와 get_checkEqual…() 함수의 변수 ret_add 의 주소가 같기 때문에 a 를 출력해 보면 get_checkEqual…() 함수 내에서 변경한 값이 저장되어 있다 . B 도 마찬가지이다 .

실습 2

배열과 랜덤 함수를 이용하여 1 부터 45 까지 6 개의 로또 번호를 생성하는 프로그램을 작성하라 .

1. 6개의 로또 번호에는 중복된 값이 있어서는 안된다 .
2. 실행할 때마다 새로운 번호를 생성해야 한다 .
3. 오름차순으로 값이 정렬되어야 한다 .
    - Chapter 6 수업 자료 중 Selection sort 를 이용하여 코드를 구성하라 .

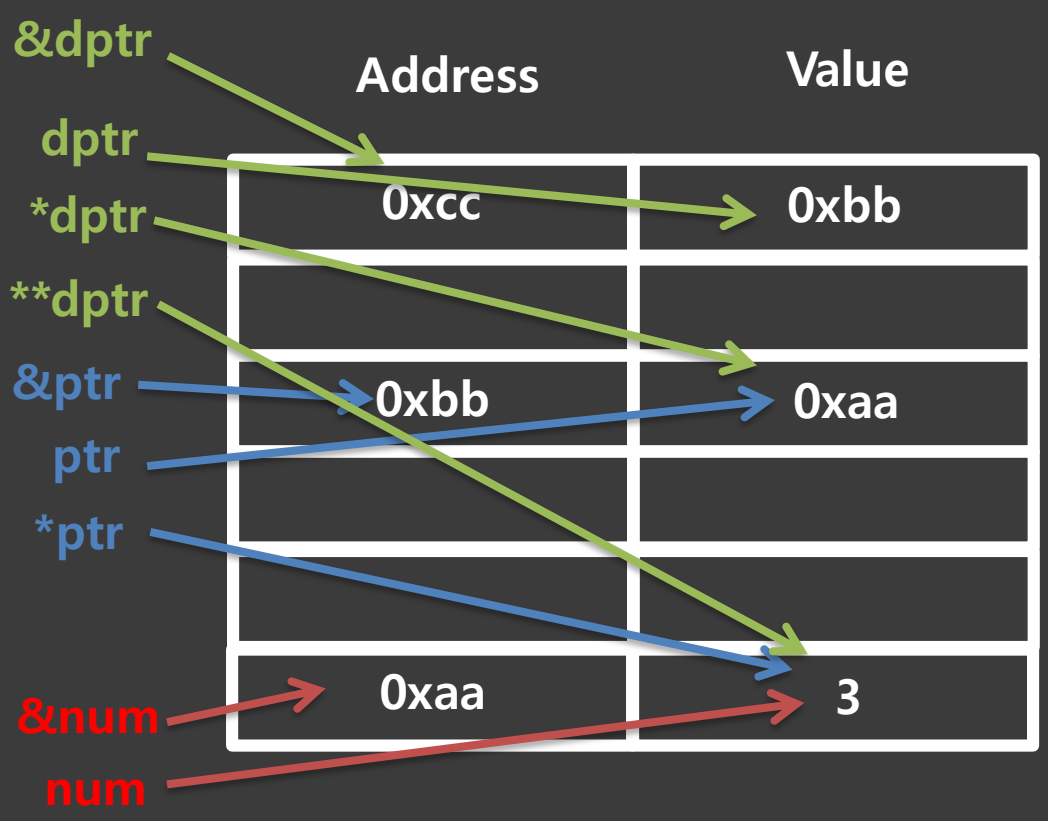
```
~/week8$ ./ex8-2
1 3 13 29 41 42
1 3 13 13 41 42 // 중복 숫자
1 42 29 3 41 13 // 非 오름차순-정렬
```

< 실행화면 >

실습 3

포인터의 개념 잡기
싱글 포인터든 더블 포인터든 * 가 안 붙을 경우 자기자신의 Value를 반환한다.
* 가 하나 붙을 경우 자신이 가리키는 변수의 value를 반환한다.
* 가 두 개 붙을 경우 자신이 가리키는 변수가 가리키는 변수의 value를 반환한다.
위의 패턴이 반복되는 구조이다.

```c
#include <stdio.h>

int main(void)
{
  int num = 3;
  int * ptr = &num;
  int ** dptr = &ptr;

  printf("========== address ==========");
  printf("&num: %p\n", &num);
  printf("&ptr: %p\n", &ptr);
  printf("&dptr: %p\n", &dptr);

  printf("pointers who point value of num\n");
  printf("%d %d %d\n", num, *ptr, **dptr);

  printf("pointers who point value of ptr\n");
  printf("%p %p\n", ptr, *dptr);

  printf("pointer who points value of dptr\n");
  printf("%p\n", dptr);


  return 0;
}
```



실습 3

포인터의 개념 잡기

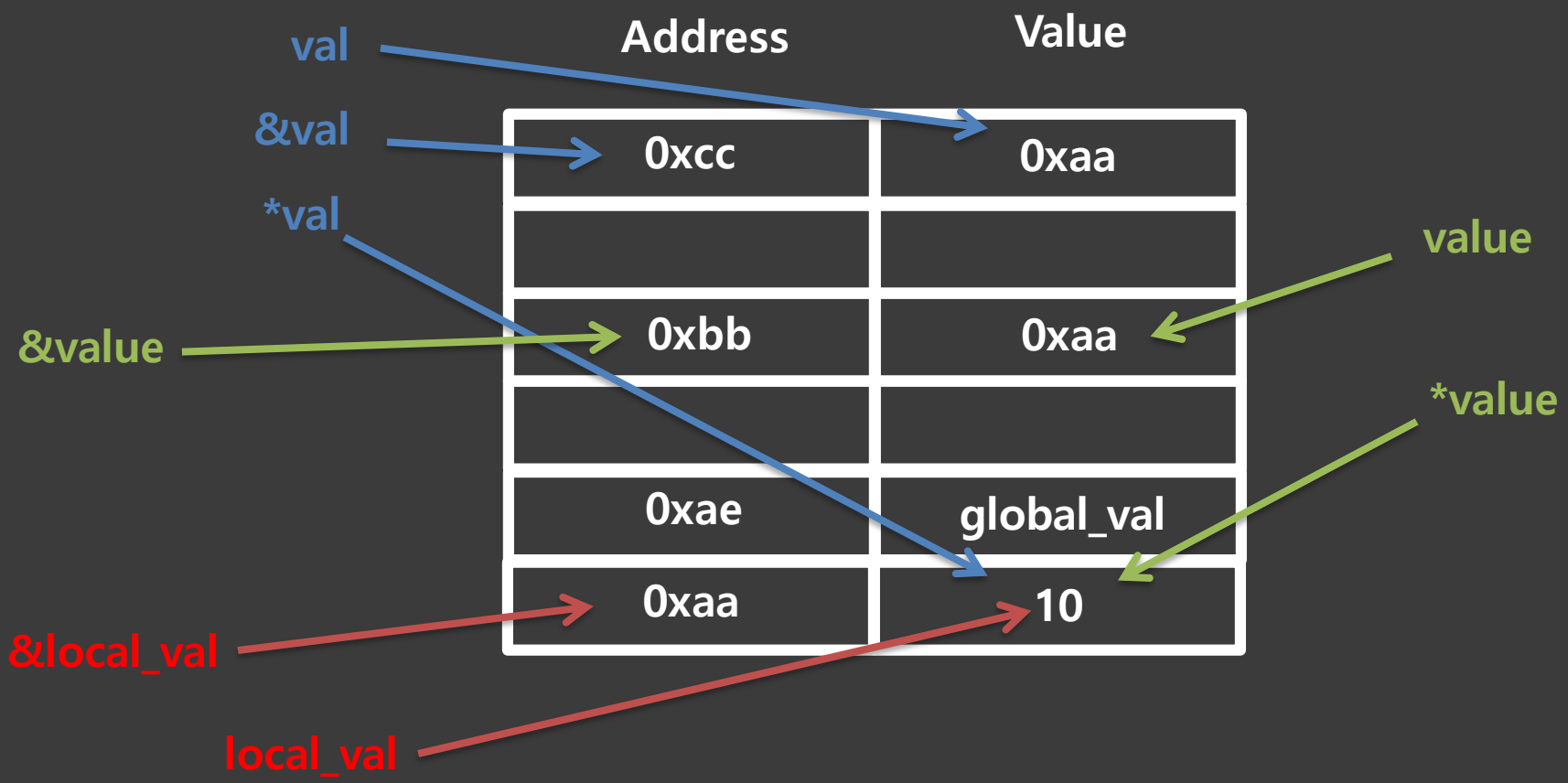
왜 포인터의 포인터를 써야 하는가?

```c
#include <stdio.h>

int global_val = 30;
int *value;
int *refer;

void call_by_value(int *val)
{
    printf("val: %p\n", val);
    printf("*val: %d\n", *val);
    val = &global_val;
    printf("val: %p\n", val);
    printf("*val: %d\n", *val);
    printf("value: %p\n\n\n", value);
}

void call_by_refer(int **ref)
{
    *ref = &global_val;
}

int main()
{
    int local_val = 10;
    value = &local_val;
    refer = &local_val;

    printf("value: %p\n", value);
    printf("local val: %p\n", &local_val);
    printf("global value: %p\n\n\n", &global_val);

    printf("before : *value=%d, *refer=%d\n", *value, *refer);
    call_by_value(value);
    printf("After call by value : *value=%d, *refer=%d\n", *value, *refer);
    call_by_refer(&refer);
    printf("After call by refer  : *value=%d, *refer=%d\n", *value, *refer);


    return 0;
}
```

실습 3

포인터의 개념 잡기

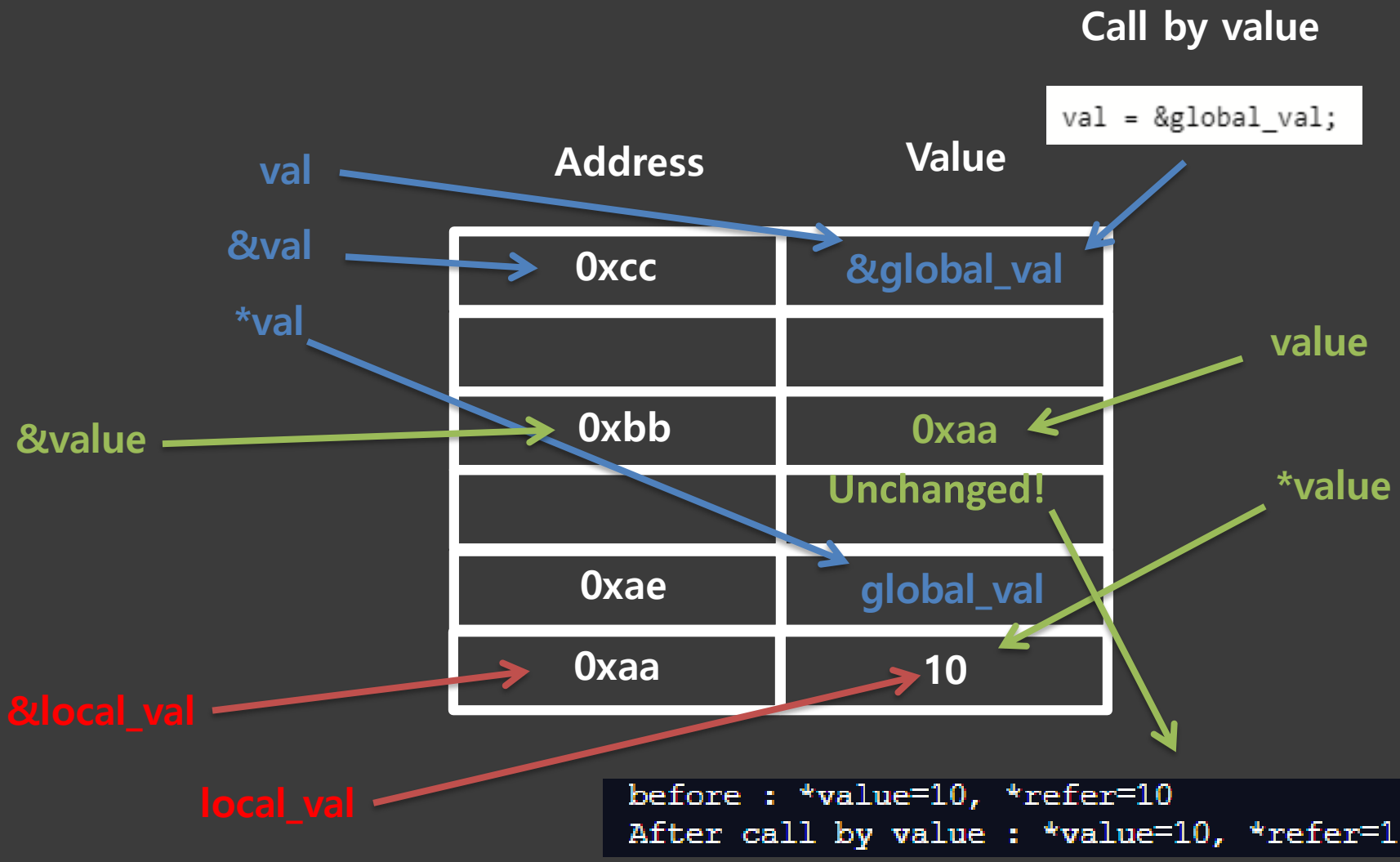
왜 포인터의 포인터를 써야 하는가?



실습 3

포인터의 개념 잡기

왜 포인터의 포인터를 써야 하는가?



실습 3

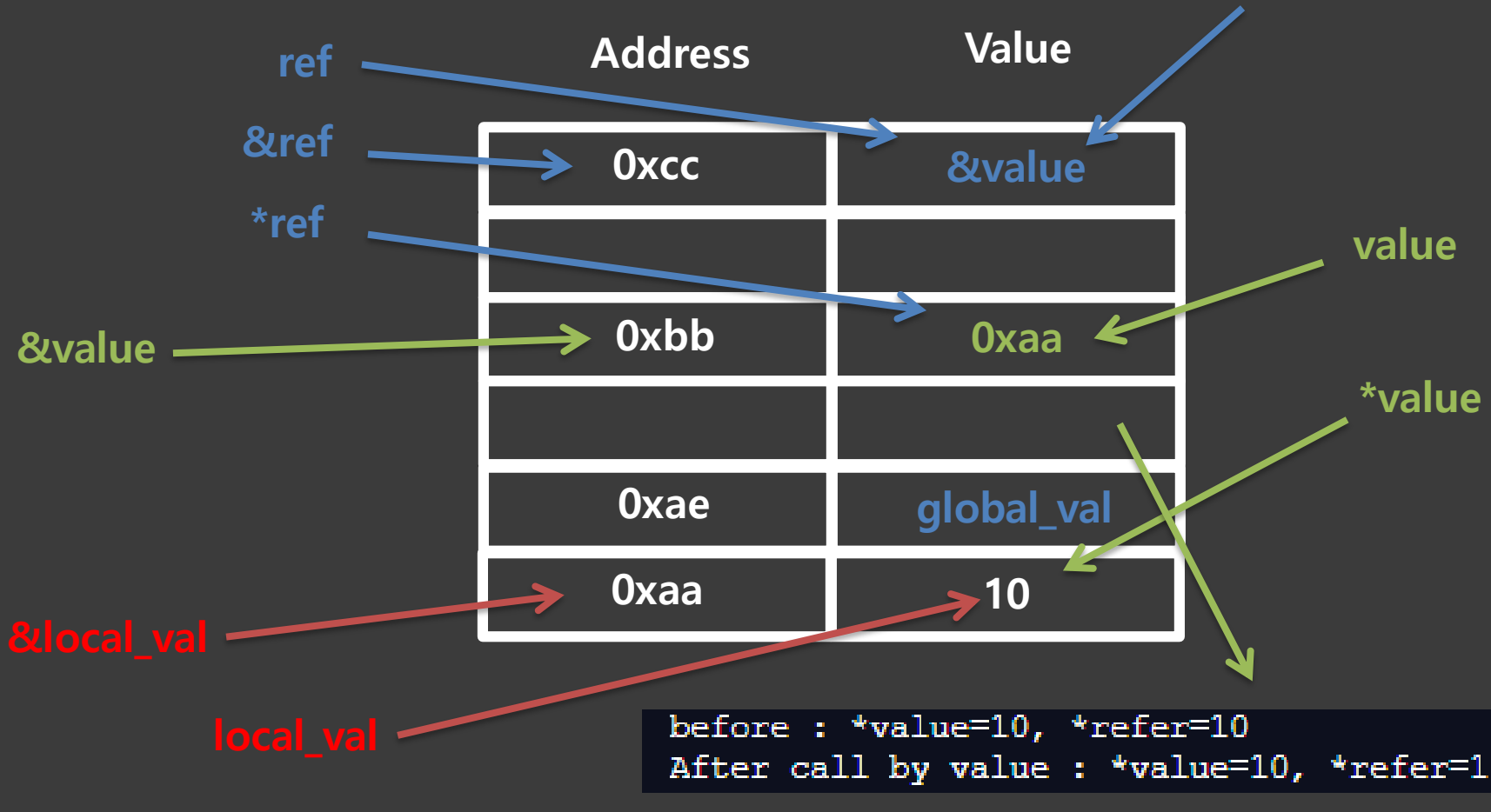
포인터의 개념 잡기

왜 포인터의 포인터를 써야 하는가?

Call by reference

```
void call_by_refer(int **ref)
{
    *ref = &global_val;
}
```

```
call_by_refer(&value);
```

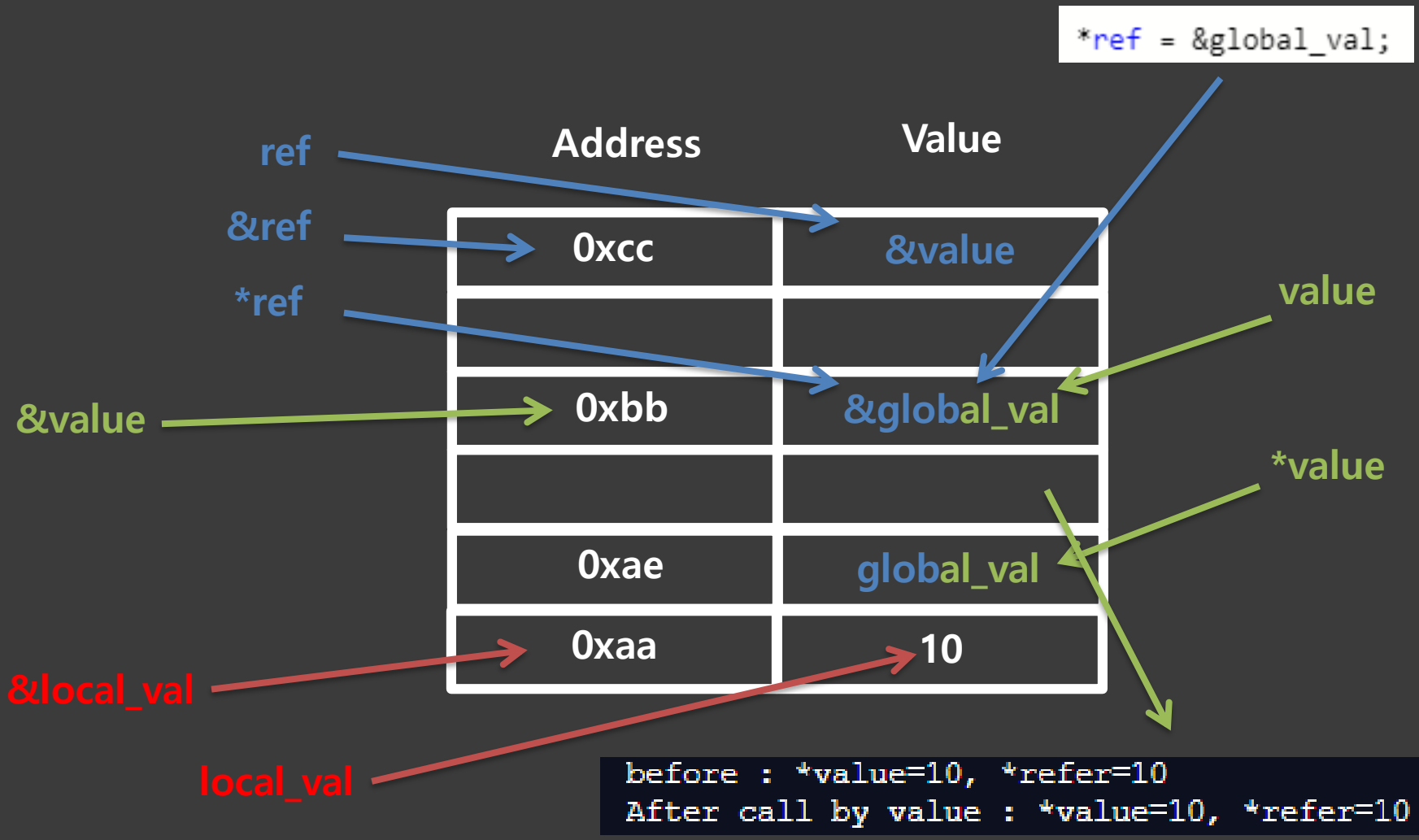



```
before : *value=10, *refer=10
After call by value : *value=10, *refer=10
```

실습 3

포인터의 개념 잡기

왜 포인터의 포인터를 써야 하는가?



과제

과제 1

배열을 이용하여 회문을 판별하는 프로그램을 작성하라 .
회문은 앞뒤 어느 쪽으로 읽어도 똑같은 문자열을 의미한다 . (회문 예 : "toot", "123343321", "C")

입력

문자열의 최대 길이는 100이며 공백을 포함하지 않는다. 문자열은 알파벳, 숫자, 특수기호 등 아스키코드로 표현할 수 있는 값들이 들어간다. ""와 같은 empty strin도 회문임을 참고하라.

출력

다음과 같이 출력하라.

< 실행화면 >

```
~/week8$ ./hw8-1
input string : toot
toot is a symmetrical word
~/week8$ ./hw8-1
input string : hello
hello is not a symmetrical word
```

과제 2

3x3 행렬의 곱셈을 수행하는 프로그램을 작성하라.
입력
3x3 행렬 A, B를 입력으로 받는다. 입력의 형태는 다음 input.txt와 같은데 첫 세 줄은 행렬 A, 그 다음 세 줄은 행렬 B를 나타낸다. 각 element는 0이상의 값을 가진다.

출력
밑의 출력 사진을 참고하여 행렬 계산의 결과를 나타내라. 꼭 똑같이 출력할 필요는 없으나 %5d 서식을 사용할 경우 예쁘게 출력할 수 있다.

```
1 2 3
4 5 6
1 2 3
100 100 0
100 100 0
100 0 0
```

```
> ./hw2 < input.txt
  600   300     0
 1500   900     0
  600   300     0
```