포인터복습

```
/*

#include <stdio.h>

int main(void)
{

    int arr[3] = {1, 2, 3};
    int *p = arr;

    int i;

    for(i = 0; i < 3; i++)
            printf("p[%d] = %d₩n", i, p[i]);
    return 0;

}




#include <stdio.h>

int main(void)

{
    int arr[3][4];

    printf("arr size = %p₩n", sizeof(arr) );
    printf("arr[0] size = %lu₩n", sizeof(arr[0]));
     printf("arr[1] size = %p₩n", &arr[1][1] );
     printf("arr[1] size = %p₩n", &arr[1][0] );
      printf("arr[2] size = %lu₩n", sizeof(arr[2]));


      return 0;
}






#include <stdio.h>

void arr_print(int arr[])                  // *arr     , arr[3]

{
    int i;
    for(i =0; i<3; i++)
        printf("%4d₩n", arr[i]);   // 4d 는 4자리를 확보해라

}

int main(void)

{
    int arr[3] = {3, 33, 333};
    arr_print(arr);                  // arr = &arr[0]
    return 0;
}
```

```
#include <stdio.h>

void add_arr(int *arr)
{
        int i;
        for(i=0; i<3; i++)
        {
            arr[i] += 7;            //arr[i] + 7 = arr[i]
        }
}


void print_arr(int *arr)
{
        int i;
        for(i =0; i <3; i++)
        {
            printf("arr[%d] = %dWn", i, arr[i]);
        }
}

int main(void)

{
        int arr[3] = {1, 2, 3};
        add_arr(arr);                       //주소값전달
        print_arr(arr);                     //주소값전달
        return 0;

}
```

```
#include <stdio.h>

int main(void)

{
int *ptr = NULL;

printf("ptr = %pWn",ptr);
printf("ptr value = %dWn",*ptr);



*ptr = 27;

printf("ptr value = %dWn", *ptr);           // Segmentation fault   (접근하지말아야할곳에 접근
함)
return 0;                                  //쓰레기값 0xcccccccc

}
```

```
#include <stdio.h>

int main(void)

{
int num =3;

*(&num) += 30;              // *(&num) num의 주소값을 받아서 계산
printf("num = %dWn", num);
```

```
return 0;
}
```

```
#include <stdio.h>

int main(void)
{
    char str1[33] = "pointer is important!";
    char *str2 = "pointer is important!";

    printf("str1 = %s₩n", str1);
    printf("str2 = %s₩n", str2);

    return 0;

}
```

```
   // *별하나당 주소
   // int *p = &num;
   // int **p = &p;
   // p(vum)
   // p(*p)
   // p (**pp)
```

```
#include <stdio.h>

int main(void)

{

int num =3;
int *p = &num;
int **pp = &p;

    printf("num = %d₩n", num);
    printf("*p = %d₩n", *p);
    printf("**pp = %d₩n", **pp);

    return 0;

}
```

```
#include <stdio.h>

int main(void)
{
    int num1 = 3, num2 = 7;
    int *temp = NULL;
    int *num1_p = &num1;                  //&주소를 반환
    int *num2_p = &num2;                 // * 내용을 반환
    int **num_p_p = &num1_p;

    printf("*num1_p = %d₩n", *num1_p);
    printf("*num2_p = %d₩n", *num2_p);
```

```
    temp = *num_p_p;                // *주의!
    *num_p_p = num2_p;
    num2_p = temp;

    printf("*num1_p = %d₩n", *num1_p);
    printf("*num2_p = %d₩n", *num2_p);

    return 0;
}



#include <stdio.h>

int main(void)
{
int i,j,n1,n2,n3;
int a[2][2] = {{10,20},{30,40}};
int*  arr_ptr[3] = {&n1, &n2, &n3};
int (*p)[2] = a;                    //int (*p)[2] = int (*)[2] p              int *p[2]
                                    //     int형 두개짜리 포인터 8바이트
                                    //


    for(i = 0; i < 3; i++)
        *arr_ptr[i] = i;


    for(i = 0; i < 3; i++)
        printf("n%d = %d₩n", i, *(arr_ptr[i]);          // int* arr-ptr[3]이랑  *arr_ptr 다름


    for(i = 0; i < 2; i++)
        printf("p[%d] = %d₩n", i, *p[i]);

return 0;


}

*/



/*#include <stdio.h>
int main(void)
{
    int i;
    int num[7];

    for (i = 0; i < 7; i++)
    {
        num[i] = i+1;
        printf("num[%d] = %d₩n", i, num[i]);
    }
    return 0;
}



#include <stdio.h>

int main(void)
{
int i;
int num1_arr[ ] = {1, 2, 3, 4, 5};
int num2_arr[3] = {1, 2, 3};
```

```
int len1 = sizeof(num1_arr)/sizeof(int);
int len2 = sizeof(num2_arr)/sizeof(int);
    printf("num1_arr length = %d₩n", len1);
    printf("num2_arr length = %d₩n", len2);
for(i = 0; i < len1; i++)
{
    printf("num1_arr[%d] = %d₩n", i, num1_arr[i]);
}

for(i = 0; i < len2; i++)
{
    printf("num2_arr[%d] = %d₩n", i, num2_arr[i]);
}
return 0;
}
```

```
#include <stdio.h>

int main(void)
{
    int i;
    int num1_arr[7] = {1, 2, 3, 4, 5, 6, 7};

    for(i = 0; i < 7; i++)
    {
        printf("num1_arr[%d] = %d₩n", i, num1_arr[i]);

    }
        return 0;

}
```

```
#include <stdio.h>

int main(void)

{
    char str1[2] = "AAA";
    char str2[2] = "BBB";
    char str3[2] = {'A','B','C'};
    char str4[2] = {'A','B','C', '₩0'};

    printf("str1 = %s₩n", str1);
    printf("str2 = %s₩n", str2);
    printf("str3 = %s₩n", str3);
    printf("str4 = %s₩n", str4);


    str1[0] = 'E';
    str1[1] = 'H';
    printf("str1 = %s₩n", str1);

    return 0;
}
```

```
#include <stdio.h>
 int main(void)

{
    int arr[4][4];
    int i, j;
```

```
    for(i=0; i <4; i++)
    {
        for(j=0; j<4; j++)
            {
                if(i == j)
                    arr[i][j] =1;
                else
                    arr[i][j] =0;
            }

    }

    for(i =0; i<4; i++)
    {
        for(j =0; j<4; j++)
        {
            printf("%d", arr[i][j]);
        }
        printf("₩n");
    }

        return 0;
}



#include <stdio.h>

int main(void)
{
    int arr[2][2] = {{10, 20}, {18, 18}};
    int i, j;

    for(i =0; i < 2; i++)
    {
        for(j =0; j<2; j++)
        {
            printf("arr[%d][%d]=%d₩n", i, j, arr[i][j]);
        }


    }

  return 0;



}









#include <stdio.h>
int main(void)
{
    int arr[4] = {10, 20, 30, 40};
    int i;
        printf("address = %p₩n", arr);
        for(i =0; i < 4; i++)
{
        printf("address= %p, arr[%d] = %d₩n", &arr[i], i, arr[i]);
}
```

```
return 0;

}
```

문제---

```
//1. 배열에 문자열을 입력 받고,
// 각 배열 요소가 짝수인 경우만을 출력하는 함수를 작성하라.
/*
#include <stdio.h>


void even(char array[])

{
    int i;
    for(i =0; i < 10; i++ )
        if (i%2 == 0)
            printf("%sWn", array[i]);  // 문자열의 내용을 출력

}

int main(void)
{
    int i;
    char array[10];    // 인 char형 배열을 선언

    printf("문자열을 입력하세요: ");

    for(i = 0; i <10; i++)

    {
            scanf("%s", array[i]);     // 표준 입력을 받아서 배열 형태의 문자열에 저장
    }

    even(array);


    return 0;
}



*/
//3번
#include <stdio.h>

void reverseArrayInt(int* array, int size);

#define SIZE 8

int main(void) {

  int a[] = { 3, 6, 9, 12, 15, 18, 21, 24 };


  // 숫자 배열 순서 거꾸로 뒤집기
  reverseArrayInt(a, SIZE);
```

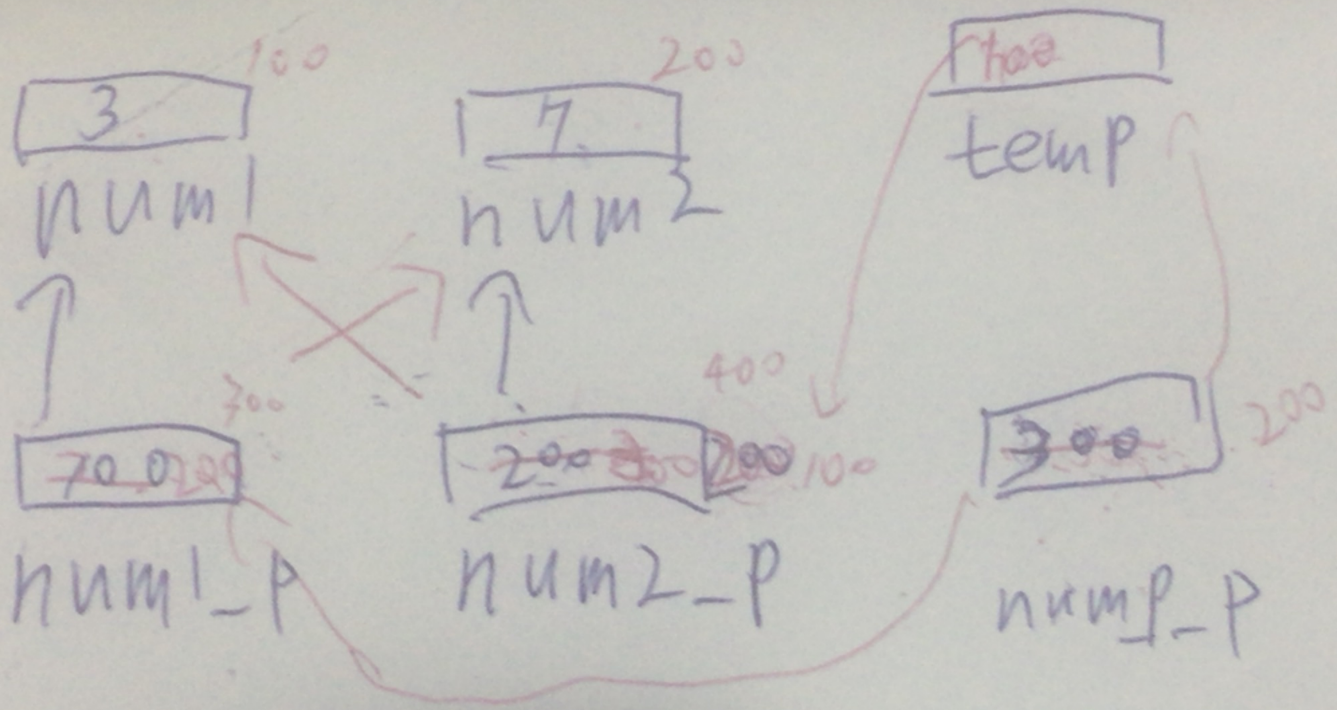
```
  // 순서 뒤집은 배열을 화면에 출력하기
  for (int i = 0; i < SIZE; i++)
    printf("%d, ", a[i]);

  return 0;
}



void reverseArrayInt(int* array, int size) {
  int temp;

  for (int i = 0; i < size / 2; i++)
  {
    temp = array[i];
    array[i] = array[(size - 1) - i];
    array[(size - 1) - i] = temp;
  }
```



**num_p_p = 3