

12 주차 Report

문제 1) 코드 아래의 빈칸을 채우시오.

```
#define SIZE 3

int get_array_sum(int *p, int n);

int main(void){

    int expenses[SIZE] = {100, 200, 200};
    printf("%d\n", get_array_sum(expenses, SIZE));
    // expenses -> 배열의 이름은 배열의 주소
}

int get_array_sum(int *p, int n){
    // int *p -> 포인터를 통하여 배열의 주소를 받는다.

    int i, result = 0;

    for(i = 0; i < n; i++)
        result += p[i]; // 포인터를 배열처럼 사용할 수 있다.

    return result;
}
```

- 메모리는 바이트 단위를 기준으로 주소가 매겨진다
- 포인터는 메모리의 주소 을(?) 저장할 수 있는 변수이다
- 변수 x 의 주소를 추출하려면 &x 라고 하면 된다.
- *p 의 의미는 포인터 p 에 가리키는 주소에 저장된 값을 반환해주는 것 이다.
- int 형 포인터 p 가 가리키는 위치에 100 을 저장하는 문장은 int *p=100; 이다
- 포인터가 아무것도 가리키고 있지 않는 경우에는 NULL 값을 넣어두는 편이 좋다
- 배열 a 에서 a 는 a[0] 의 주소이다.
- p 가 포인터라면 p[2]는 수식 *(p+ 2)와 같다.
- *p++의 의미는 p 가 가리키는 위치에서 값을 가져온 후에 p 를 증가한다는 것 이다.

- 사칙 연산 중에서 포인터에 대하여 적용할 수 있는 연산에는 덧셈, 뺄셈 이 있다.
- int 형 포인터 p 가 80 번지를 가리키고 있었다면 (p+1)은 84 번지를 가리킨다
- 함수 호출 시 인수 전달 방법 중에서 기본적인 방법은 "값에 의한 호출"이다.

문제 2) 다음 코드의 빈칸에, 주석에 알맞은 문장을 넣으시오.

```
char code;

char *p; // char 형 포인터 p 선언
p = &code; // 포인터에 변수 code 의 주소 대입
*p = 'a'; // 포인터를 통하여 변수 code 에 'a' 대입하기
```

문제 3) 다음 문제들을 풀고, 왜 그리 되었는지 설명하시오.

3-1. 아래 문장이 실행되었다고 가정하자. 다음 중 다른 문장들과 실행 결과가 다른 것은?

답 : 3) *p++;

```
int i;
int *p = &i;
```

- 1) i = i + 1; // i 를 1 씩 증가시키는 문장
- 2) i++; // i 를 1 씩 증가시키는 문장
- 3) *p++; // p 가 가리키는 i 값을 가져온 후, p 를 1 씩 증가시키는 문장
- 4) *p = *p + 1; // i 를 1 씩 증가시키는 문장, 포인터를 이용

3-2. 다음 프로그램의 출력은?

```
/* 답 :
5
5
*/

int x = 6; // 정수형 변수 x 를 6 으로 초기화
int *p = &x; // 변수 x 의 주소가 포인터 변수 p 로 초기화
printf("%d\n", --(*p));
```

```
// p 가 가리키는 위치의 값에 1 을 먼저 빼 주고, 그 값인 5 를 출력
printf("%d\n", (*p)--);
// 선행된 출력문 이후의 p 가 가리키는 위치의 값인 5 를 먼저 출력한 뒤 1 을 더해 줌
```

3-3. 다음 프로그램의 출력은?

```
/* 답 :
1008
2008
*/
int *p = (int*)1000;
// 정수형 포인터 p 를 같은 자료형으로 형변환 해주고 절대주소 1000 으로 지정
double *q = (double*)2000;
// 실수형 포인터 q 를 같은 자료형으로 형변환 해주고 절대주소 2000 으로 지정
printf("%d\n", p + 2);
// p 의 절대주소 1000 에 int 자료형 크기인 4 를 두번 더해주어 1008 출력
printf("%d\n", q + 1);
// q 의 절대주소 2000 에 double 자료형 크기인 8 을 한번 더해주어 2008 출력
```

문제 4) 다음의 프로그램 소스를 보고 한 줄 한 줄 코드가 실행될 때 마다 변수들이 어떻게 변하는지 그림 등을 이용하여 자세히 설명하시오.

```
#include <stdio.h>

void exchange(int *, int *);

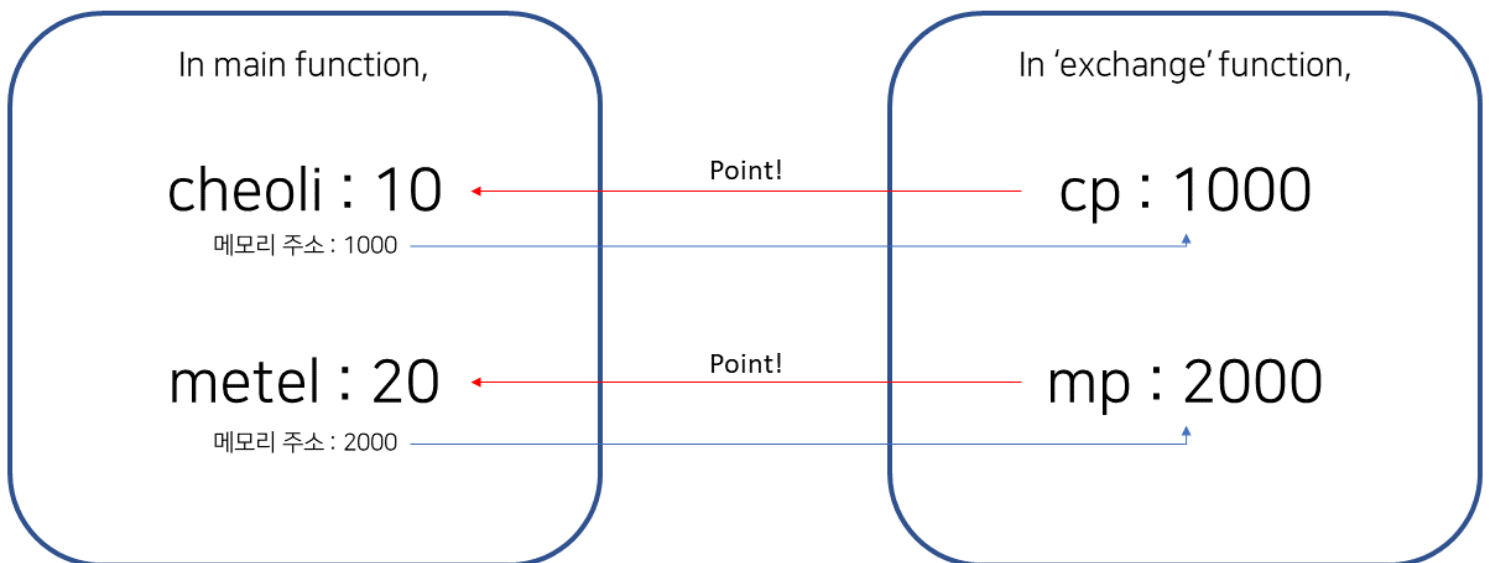
int main()
{
    int cheoli = 10; metel = 20;
    // 정수형 변수 cheoli, metel 에 각각 10, 20 을 저장

    exchange(&cheoli, &metel); // 그림 1
    // 변수의 값 바꾸는 사용자 지정함수 exchange 호출
    // 정수형 변수 cheoli, metel 가 각각 가리키는 주소를 함수의 매개변수로 넘김

    return 0;
}

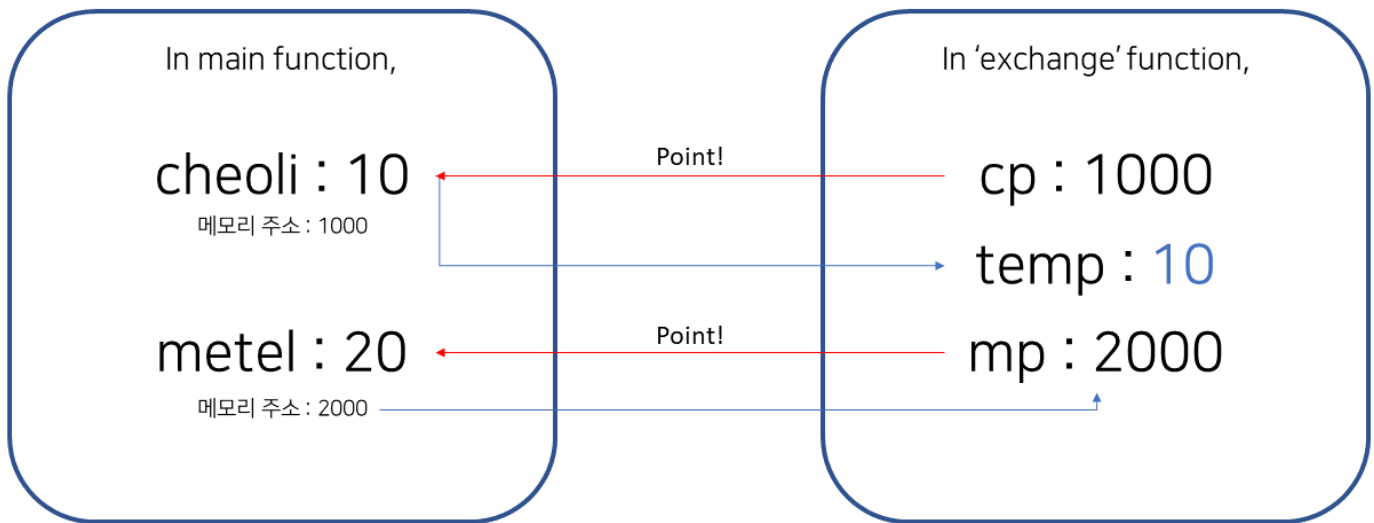
void exchange(int *cp, int *mp)
```

```
// 변수의 값 바꾸는 사용자 지정함수 exchange
// 메인함수의 두 변수에서 받은 주소를 포인터 매개변수 cp, mp 에 초기화
{
    int temp; // 변수 값 바꾸기 위해서 정수형 임시변수 선언
    temp = *cp; // cp 가 가리키는 값을 temp 에 저장, 그림 2
    *cp = *mp; // mp 의 값을 cp 가 있는 주소로 찾아가 덮어씌우기, 그림 3
    *mp = temp; // temp 의 값을 mp 가 있는 주소로 찾아가 덮어씌우기, 그림 4
}
```

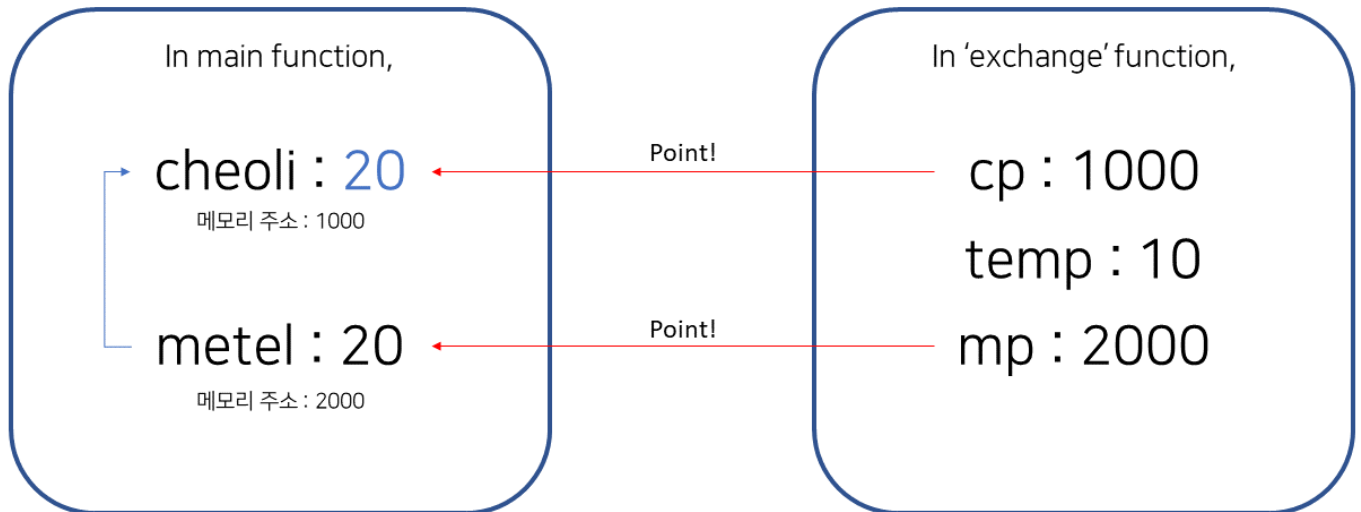


[그림 1, exchange(&cheoli, &metel);]

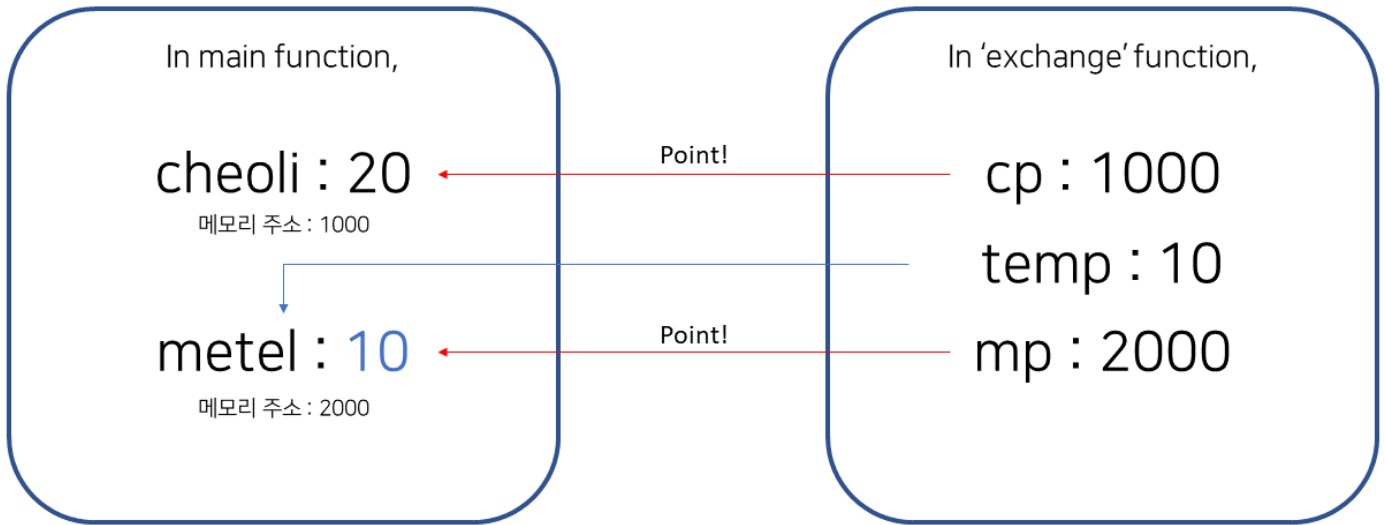
Exchange 함수가 호출되면 변수 cheoli와 metel의 주소값이 매개변수 cp, mp에 저장, 두 포인터 변수 cp, mp는 main 함수에 있는 cheoli, metel의 두 변수를 가리킴.



[그림 2, temp = *cp;]
cp가 가리키는 값을 temp에 저장



[그림 3, *cp = *mp;]
mp의 값을 cp가 있는 주소로 찾아가 덮어쓰우기



[그림 4, *mp = temp;]

temp의 값을 mp가 있는 주소로 찾아가 덮어쓰우기

문제 5) 2 개의 정수의 합과 차를 동시에 반환하는 함수를 작성하고 테스트하라. 포인터 매개 변수를 사용한다.

```
void get_sum_diff(int x, int y, int *p_sum, int *p_diff){  
    ...  
}
```

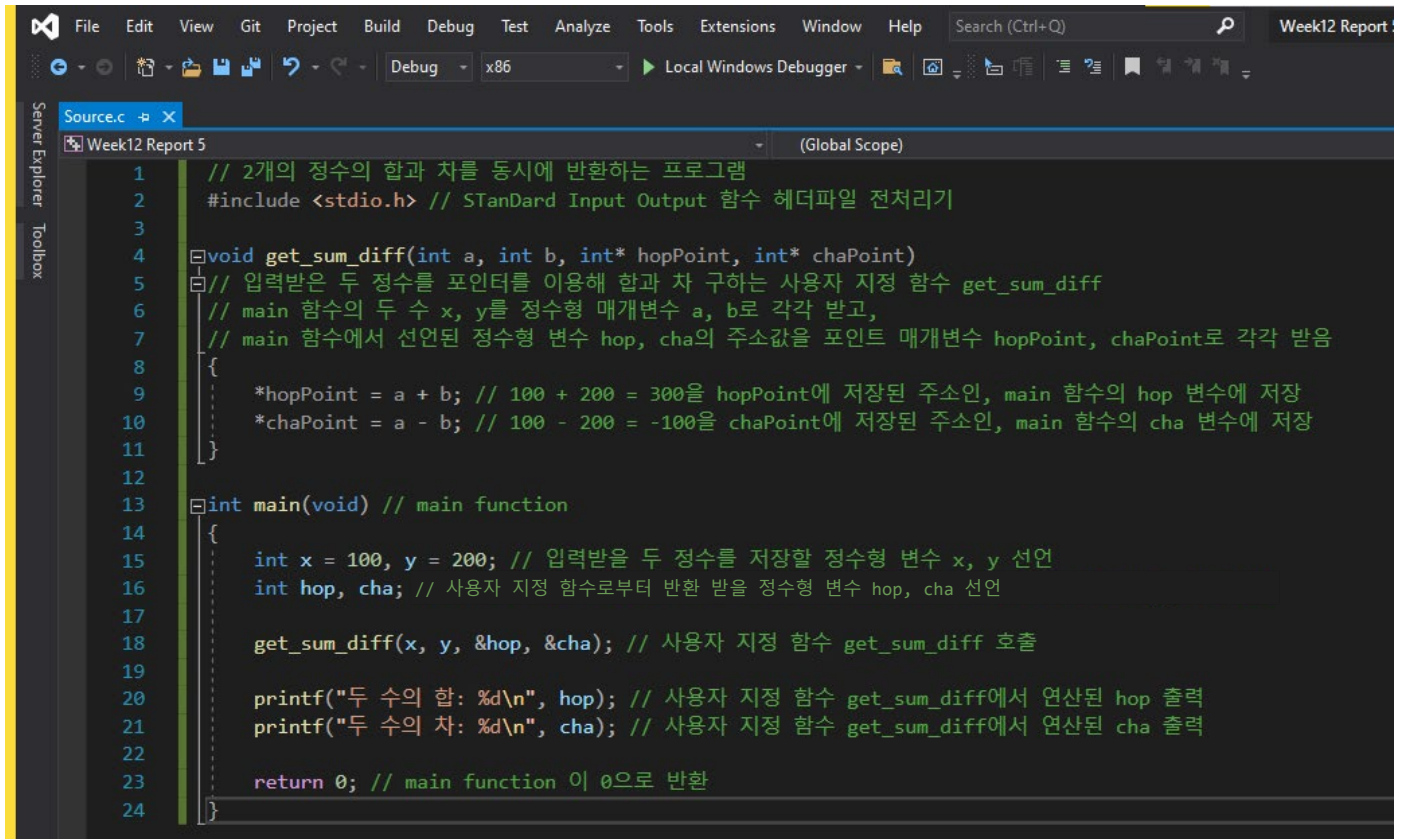
Hint : 함수 매개 변수에 포인터를 사용하면 2 개 이상의 값을 반환할 수 있다. 본문에서 직선의 기울기와 절편을 반환하는 예제를 참고하라.

- 문제 분석 & 동작 설명

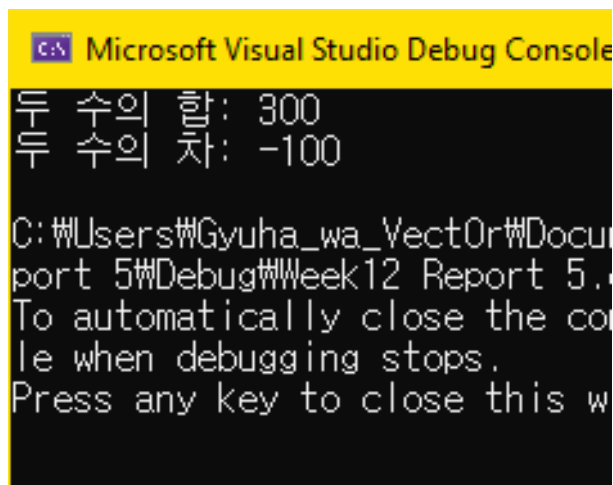
사용자 지정 함수를 만드는데 return 하지 말고, 포인터를 이용해 직접 반환하는 것이 핵심이고, Hint 에도 친히 제시되어 있다. 결과 값에 합이 300 이고, 차가 -100 이면 두 수는 각각 100, 200 이어야 하는데 이를 변수 선언할 때 바로 초기화 해주도록 하자. 그리고 문제에 나와있는 사용자 지정 함수 get_sum_diff 로 부터 합과 차를 반환 받기 위한 정수형 변수도 선언해주자. 두 수의 값을 사용자 지정 함수의 정수형 매개변수에 넣고, 합과 차를 반환 받기 위한 정수형 변수의 주소 값을 사용자 지정 함수의 포인트 매개변수에 넣도록 하자. 합과 차를 연산하여 그 결과를 포인터 변수에

저장하는데 그 포인터 변수는 메인 함수에서 사용자 지정 함수로부터 반환될 두 변수의 주소로 종속되어 있으므로, 그 주소에 해당되는 변수 내에 합과 차를 저장하도록 하자. 저장된 합과 차의 값을 출력하면서 프로그램을 종료하도록 하자.

- 프로그램 소스 (주석 포함)



```
1 // 2개의 정수의 합과 차를 동시에 반환하는 프로그램
2 #include <stdio.h> // STAnDard Input Output 함수 헤더파일 전처리기
3
4 void get_sum_diff(int a, int b, int* hopPoint, int* chaPoint)
5 {
6     // 입력받은 두 정수를 포인터를 이용해 합과 차 구하는 사용자 지정 함수 get_sum_diff
7     // main 함수의 두 수 x, y를 정수형 매개변수 a, b로 각각 받고,
8     // main 함수에서 선언된 정수형 변수 hop, cha의 주소값을 포인터 매개변수 hopPoint, chaPoint로 각각 받음
9     {
10         *hopPoint = a + b; // 100 + 200 = 300을 hopPoint에 저장된 주소인, main 함수의 hop 변수에 저장
11         *chaPoint = a - b; // 100 - 200 = -100을 chaPoint에 저장된 주소인, main 함수의 cha 변수에 저장
12     }
13
14 int main(void) // main function
15 {
16     int x = 100, y = 200; // 입력받은 두 정수를 저장할 정수형 변수 x, y 선언
17     int hop, cha; // 사용자 지정 함수로부터 반환 받을 정수형 변수 hop, cha 선언
18
19     get_sum_diff(x, y, &hop, &cha); // 사용자 지정 함수 get_sum_diff 호출
20
21     printf("두 수의 합: %d\n", hop); // 사용자 지정 함수 get_sum_diff에서 연산된 hop 출력
22     printf("두 수의 차: %d\n", cha); // 사용자 지정 함수 get_sum_diff에서 연산된 cha 출력
23
24     return 0; // main function 이 0으로 반환
25 }
```



```
C:\ Microsoft Visual Studio Debug Console
두 수의 합: 300
두 수의 차: -100

C:\Users\Gyuha_wa_VectOr\Documents\port 5\Debug\Week12 Report 5.
To automatically close the console when debugging stops,
Press any key to close this window
```