



R E P O R T

C프로그래밍2 과제4

과목명	C 프로그래밍 II
분반	2 분반
교수	정 구 철
학번	2020136129
이름	최 수 연
제출일	2020년 10월 6일 화요일

(문제 1~3).

int a=100, int b=200으로 정의한다.

두 변수를 교체하는 swap_val의 함수 원형은 다음과 같이 정의된다.

- void swap_val(int x, int y)

두 변수를 교체하는 swap_ref의 함수 원형은 다음과 같이 정의된다.

- void swap_ref(int* px, int* py)

두 함수 모두 변수를 교체한 후 함수 내부에서 바뀐 변수 값을 출력할 수 있어야 한다.

1. a,b 값을 출력하시오

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

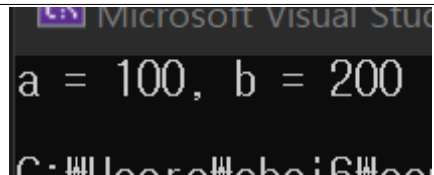
```
    int a = 100;
```

```
    int b = 200;
```

```
    printf("a = %d, b = %d\n", a, b);
```

```
    return 0;
```

```
}
```



```
Microsoft Visual Studio  
a = 100, b = 200  
C:\Users\user\Desktop
```

2. swap_val을 구현하시오. swap_val을 사용해 a,b값을 교체 후 함수 내에서 a,b를 출력한 뒤, 함수 바깥에서 한 번 더 a,b값을 출력하시오

- ☐ swap_val에서 한 번, swap_val 바깥에서 한 번 더 a,b값이 출력되므로 a,b는 총 두 번 출력 되어야 함

```
#include <stdio.h>

void swap_val(int x, int y);

int main(void)
{
    int a = 100;
    int b = 200;

    swap_val(a, b);

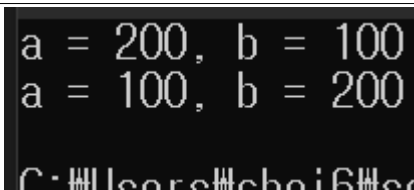
    printf("a = %d, b = %d\n", a, b);

    return 0;
}

void swap_val(int x, int y)
{
    int tmp;

    tmp = x;
    x = y;
    y = tmp;

    printf("a = %d, b = %d\n", x, y);
}
```



```
a = 200, b = 100
a = 100, b = 200
C:\Users\cheoi6\Miniconda3\Scripts\
```

3. swap_ref을 구현하시오. swap_ref을 사용해 a,b값을 교체 후 함수 내에서 a,b를 출력한 뒤, 함수 바깥에서 한 번 더 a,b값을 출력하시오

- ☐ swap_ref에서 한번, swap_ref 바깥에서 한 번 더 a,b값이 출력되므로 a,b는 총 두 번 출력 되어야 함

```
#include <stdio.h>

void swap_ref(int* px, int* py);

int main(void)
{
    int a = 100;
    int b = 200;

    swap_ref(&a, &b);

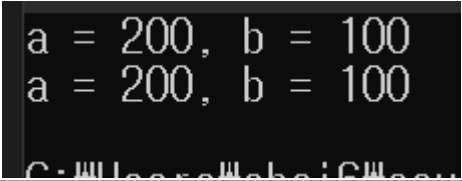
    printf("a = %d, b = %d\n", a, b);

    return 0;
}

void swap_ref(int* px, int* py)
{
    int tmp;

    tmp = *px;
    *px = *py;
    *py = tmp;

    printf("a = %d, b = %d\n", *px, *py);
}
```



```
a = 200, b = 100
a = 200, b = 100
```

(문제 4~6)

정수형 벡터 a,b는 다음과 같이 정의된다.

- `int a[10] = {1,2,3,4,5,6,7,8,9,0}`
- `int b[10] = {0}`

정수형 벡터를 출력하는 함수의 원형은 다음과 같이 정의된다.

- `void print_vec(int a[], int size)`

정수형 벡터를 얇은 복사(shallow copy)하는 함수의 원형은 다음과 같이 정의된다.

- `int copy_shallow(int *target, int *dest)`

정수형 벡터를 깊은 복사(deep copy)하는 함수의 원형은 다음과 같이 정의된다.

- `int copy_deep(int *target, int *dest, int size)`

두 함수 모두 벡터를 복사한 뒤, 벡터 dest를 출력하여야 한다.

4. print_vec을 구현하고 a,b를 출력하시오

```
#include <stdio.h>


void print_vec(int a[], int size);

int main(void)
{
    int a[10] = { 1,2,3,4,5,6,7,8,9,0 };
    int b[10] = { 0 };

    printf("a = ");
    print_vec(a, 10);
    printf("\n");
    printf("b = ");
    print_vec(b, 10);
    printf("\n");

    return 0;
}

void print_vec(int a[], int size)
{
    for (int i = 0; i < size; i++)
    {
        printf("%d  ", a[i]);
    }
}
```



```
a = 1 2 3 4 5 6 7 8 9 0
b = 0 0 0 0 0 0 0 0 0 0
```

C:\Users\chei6\source\repos\Proj

5. 얇은 복사를 하는 함수 `copy_shallow`를 구현하고, `a`를 `b`에 복사한 후 함수 내에서 `b`를 출력하고 함수 바깥에서 `b`의 값을 출력하시오.

```
#include <stdio.h>

int copy_shallow(int* target, int* dest);

int main(void)
{
    int a[10] = { 1,2,3,4,5,6,7,8,9,0 };
    int b[10] = { 0 };

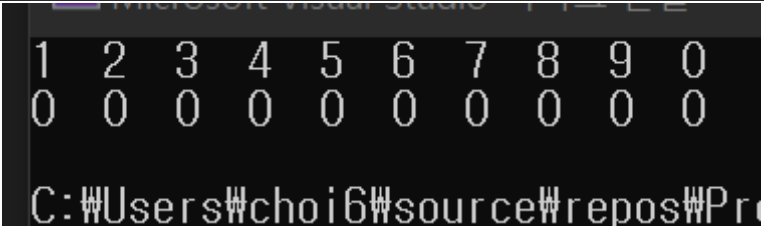
    copy_shallow(a, b);
    printf("\n");

    for (int i = 0; i < 10; i++)
    {
        printf("%d ", b[i]);
    }
    printf("\n");
    return 0;
}

int copy_shallow(int* target, int* dest)
{
    dest = target;

    for (int i = 0; i < 10; i++)
    {
        printf("%d ", dest[i]);
    }

    return 0;
}
```



6. 깊은 복사를 하는 함수 `copy_deep`을 구현하고, `a`를 `b`에 복사한 후 함수 내에서 `b`를 출력하고 함수 바깥에서 `b`의 값을 출력하시오.

```
#include <stdio.h>

int copy_deep(int* target, int* dest, int size);

int main(void)
{
    int a[10] = { 1,2,3,4,5,6,7,8,9,0 };
    int b[10] = { 0 };

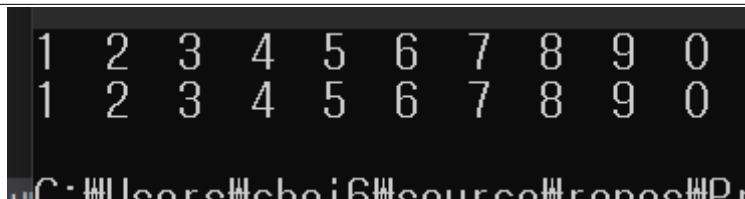
    copy_deep(a, b, 10);
    printf("\n");

    for (int i = 0; i < 10; i++)
    {
        printf("%d  ", b[i]);
    }

    printf("\n");
    return 0;
}

int copy_deep(int* target, int* dest, int size)
{
    for (int i = 0; i < size; i++)
    {
        dest[i] = target[i];
        printf("%d  ", dest[i]);
    }

    return 0;
}
```



```
1 2 3 4 5 6 7 8 9 0
1 2 3 4 5 6 7 8 9 0
C:\Users\choi6\source\repos\Pr
```


(문제 7~10).

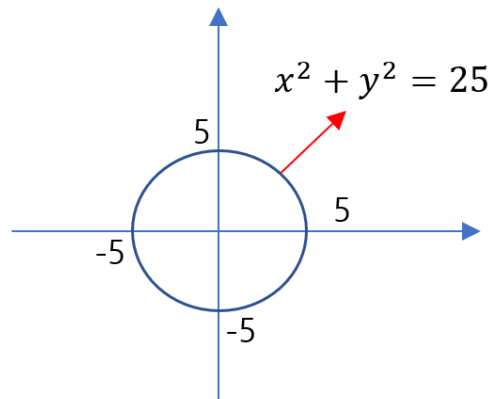


그림1. 반지름이 5이고, 중심 좌표가 (0,0)인 원

그림1과 같이 반지름이 5이고, 중심 좌표가 (0,0)인 원이 있다고 하자.

x 좌표 하나를 넣었을 때, y좌표 두 개를 출력하는 함수 `get_single_y`의 원형은 다음과 같이 정의된다.

- `int get_single_y(double x, double *y1, double *y2)`

x좌표들을 넣었을 때(배열을 넣었을 때), y 값들을 출력하는 함수 `get_multi_y`의 원형은 다음과 같이 정의된다.

- `int get_multi_y(double *x, double *y1, double *y2)`

7. `sqrt`함수를 사용하여 25의 제곱근을 구하고 출력하시오

☐ Hint, `sqrt` 함수는 `#include <math.h>`를 통해 import 가능하다.

```
#include <stdio.h>
#include <math.h>

int main(void)
{
    printf("result = %.0f\n", sqrt(25));
    return 0;
}
```

```
result = 5
C:\Users\choi6\
```

8. get_single_y를 구현하시오. 그리고 x는 7을 집어넣었을 때 결과값을 출력하시오.

☐ x의 값이 반지름의 범위를 넘었을 때, -1을 반환하여야 함.

☐ main에서 함수가 -1을 반환했을 때 "input range is out"이라고 출력되어야함(함수 내에서 출력하는 것 아님)

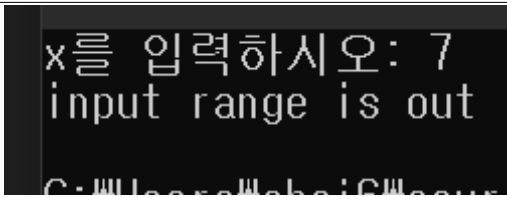
```
#include <stdio.h>
#include <math.h>
int get_single_y(double x, double* y1, double* y2);

int main(void)
{
    double a, x = 0;
    double y1[] = { NULL };
    double y2[] = { NULL };

    printf("x를 입력하시오: ");
    scanf_s("%lf", &x);
    a = get_single_y(x, y1, y2);

    if (a == -1)
    {
        printf("input range is out\n");
    }
    return 0;
}

int get_single_y(double x, double* y1, double* y2)
{
    if (x >= -5 && x <= 5)
    {
        *y1 = sqrt(25 - x * x);
        *y2 = -sqrt(25 - x * x);
        printf("y = %lf, %lf\n", *y1, *y2);
    }
    else
        return -1;
}
```



```
x를 입력하시오: 7
input range is out
```

9. 8에서 만든 get_single_y를 가지고 x에 3을 집어넣었을 때 결과값을 출력하시오.

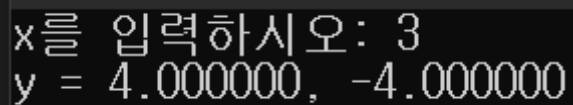
```
#include <stdio.h>
#include <math.h>
int get_single_y(double x, double* y1, double* y2);

int main(void)
{
    double a, x = 0;
    double y1[] = { NULL };
    double y2[] = { NULL };

    printf("x를 입력하시오: ");
    scanf_s("%lf", &x);
    a = get_single_y(x, y1, y2);

    if (a == -1)
    {
        printf("input range is out\n");
    }
    return 0;
}

int get_single_y(double x, double* y1, double* y2)
{
    if (x >= -5 && x <= 5)
    {
        *y1 = sqrt(25 - x * x);
        *y2 = -sqrt(25 - x * x);
        printf("y = %lf, %lf\n", *y1, *y2);
    }
    else
        return -1;
}
```



x를 입력하시오: 3
y = 4.000000, -4.000000

10. get_multi_y(double *x, double *y1, double *y2)를 구현하고 결과값을 출력하시오.

- ☐ x의 값이 하나라도 반지름의 범위를 넘으면, -1을 반환하여야 함.
- ☐ 배열 x,y1,y2의 SIZE는 5로 통일
- ☐ x[SIZE] = {1, 1.5, 2, 2.5, 3}로 정의

```
#include <stdio.h>
#include <math.h>
#define SIZE 5
int get_multi_y(double* x, double* y1, double* y2);

int main(void)
{
    double x[SIZE] = { 1, 1.5, 2, 2.5, 3 };
    double y1[SIZE] = { NULL }, y2[SIZE] = { NULL }, a;
    a = get_multi_y(x, y1, y2);

    if (a == -1)
    {
        printf("input range is out\n");
    }
    return 0;
}

int get_multi_y(double* x, double* y1, double* y2)
{
    if (*x >= -5 && *x <= 5)
    {
        for (int i = 0; i < 5; i++)
        {
            y1[i] = sqrt(25 - x[i] * x[i]);
            y2[i] = -sqrt(25 - x[i] * x[i]);
            printf("x = %.1f일 때, y = %lf, %lf\n", x[i], y1[i], y2[i]);
        }
    }
    else
        return -1;
}
```

```
x = 1.0일 때, y = 4.898979, -4.898979
x = 1.5일 때, y = 4.769696, -4.769696
x = 2.0일 때, y = 4.582576, -4.582576
x = 2.5일 때, y = 4.330127, -4.330127
x = 3.0일 때, y = 4.000000, -4.000000
```