

실습 1.

```
#include <stdio.h>
//global variable definition
int z;

int main() {
    printf("value of z = %d \n", z);

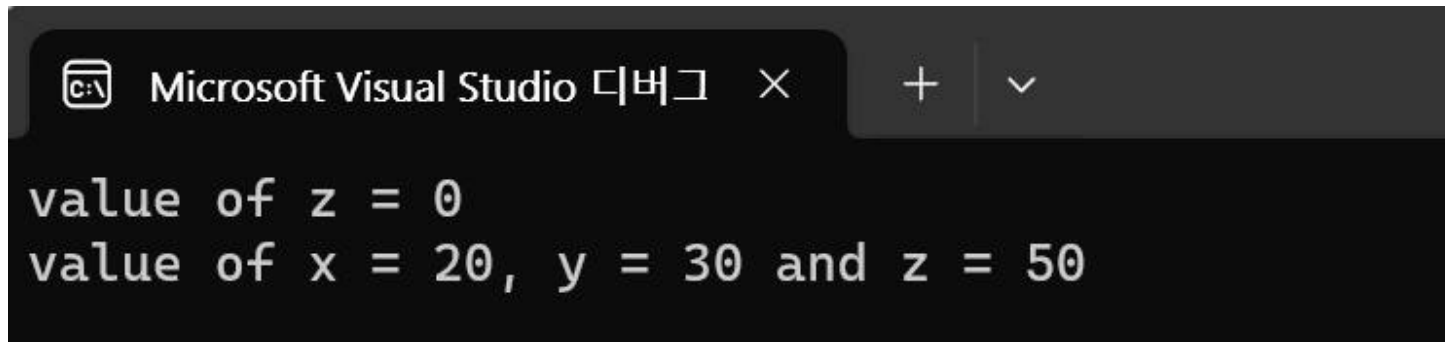
    //local variable definition and initialization
    int x, y;

    // cause complie error!
    // printf("value of x = %d, y = %d \n", x, y);

    //actual initialization
    x = 20;
    y = 30;
    z = x + y;
    printf("value of x = %d, y = %d and z = %d\n", x, y, z);

    return 0;
}
```

실습 1 실행 화면.

The image shows a screenshot of the Microsoft Visual Studio debugger's output window. The window has a dark theme and a title bar that reads "Microsoft Visual Studio 디버거". The output text is displayed in a monospaced font and shows the results of the program's execution. The first line of output is "value of z = 0", and the second line is "value of x = 20, y = 30 and z = 50".

```
Microsoft Visual Studio 디버거 × + ▾
value of z = 0
value of x = 20, y = 30 and z = 50
```

실습 2.

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

```
int find_larger(); // 함수 원형 선언
```

```
int n1, n2, max; // 전역 변수 선언
```

```
int main() {
```

```
    int width, height; //main의 지역 변수 선언
```

```
    printf("첫째 정수? "); scanf("%d", &n1);
```

```
    printf("둘째 정수? "); scanf("%d", &n2);
```

```
    // 전역 변수 n1과 n2 중 큰 값을 구하여 max에 저장하기
```

```
    max = find_larger();
```

```
    printf("n1=%d, n2=%d 중 큰 값은 %d \n", n1, n2, max);
```

```
    // main의 width와 height 중 큰 값을 구하여 ma에 저장하기
```

```
    width = n1 * 4;
```

```
    height = n2;
```

```
    max = find_larger();
```

```
    printf("width=%d, height=%d 중 큰 값은 %d\n", width, height, max);
```

```
    return 0;
```

```
}
```

```
//전역 변수 n1과 n2 중 큰 값을 반환하는 함수
```

```
int find_larger() {
```

```
    if (n1 > n2)
```

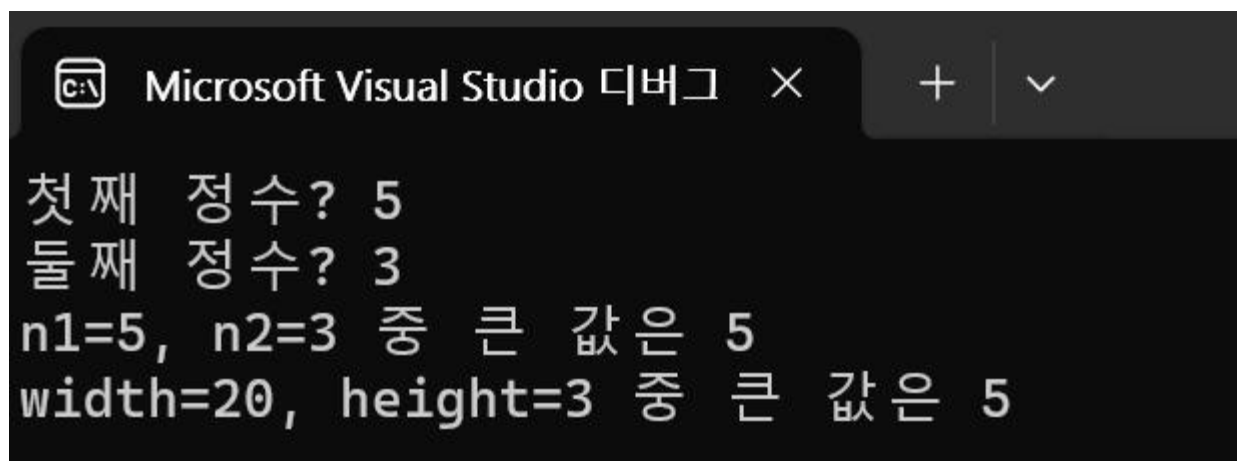
```
        return n1;
```

```
    else
```

```
        return n2;
```

```
}
```

실습 2 실행 화면.



```
Microsoft Visual Studio 디버그 × + ▾

첫째 정수? 5
둘째 정수? 3
n1=5, n2=3 중 큰 값은 5
width=20, height=3 중 큰 값은 5
```

실습 3.

```
/* program to illustrate name hiding and scope resolution */
#include<stdio.h>

/* global variables */
int num1 = 10;

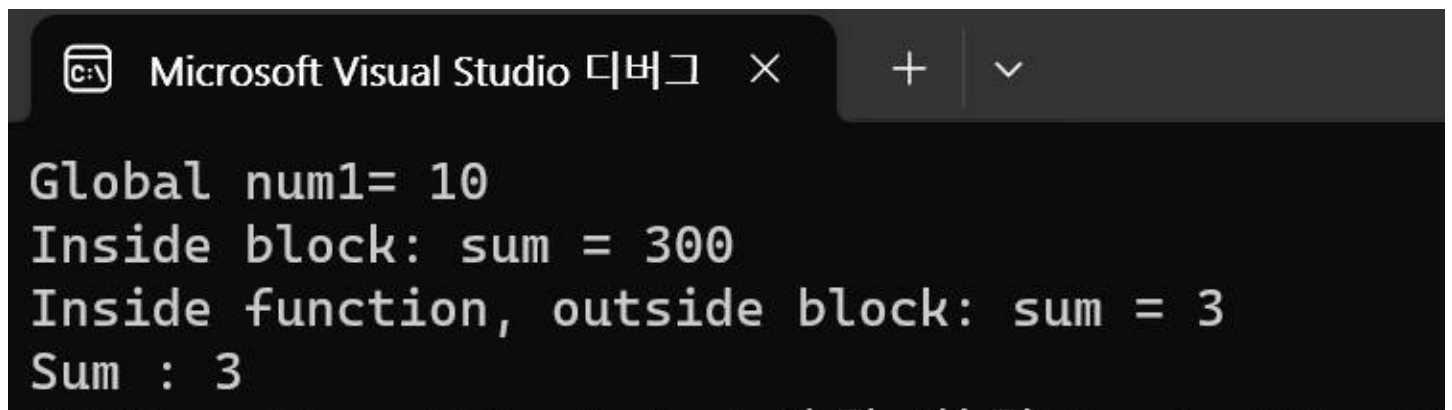
/* This function takes two int arguments, and returns an int. */
int add_integers(int int1, int int2) {

    int sum = int1 + int2;
    {
        //begining of block scope
        int sum = 300;
        printf("Inside block: sum = %d \n", sum);
    } //end of block scope: sum goes out of scope
    printf("Inside function, outside block: sum = %d \n", sum);
    return sum;
} //end of function: int1, int2, sum goes out of scope

/* A function that calls add_integers() */
void caller_function(void) {
    int num1 = 1, num2 = 2, result; //num1 supresses global num1
    result = add_integers(num1, num2); /* result =3 */
    printf("Sum : %d", result);
}

int main() {
    printf("Global num1= %d \n", num1); //prints global value of int1 i.e. 10
    caller_function();
    return 0;
}
```

실습 3 실행 화면.



The screenshot shows the Microsoft Visual Studio debug console window. The title bar reads "Microsoft Visual Studio 디버그" with a close button. The console output is as follows:

```
Global num1= 10
Inside block: sum = 300
Inside function, outside block: sum = 3
Sum : 3
```

2019313550_박병현

실습 4.

ex_header.h 코드

```
#pragma once
```

```
extern int extern_test;
```

ex_body.c 코드

```
#include "ex_header.h"
```

```
int extern_test = 10032023;
```

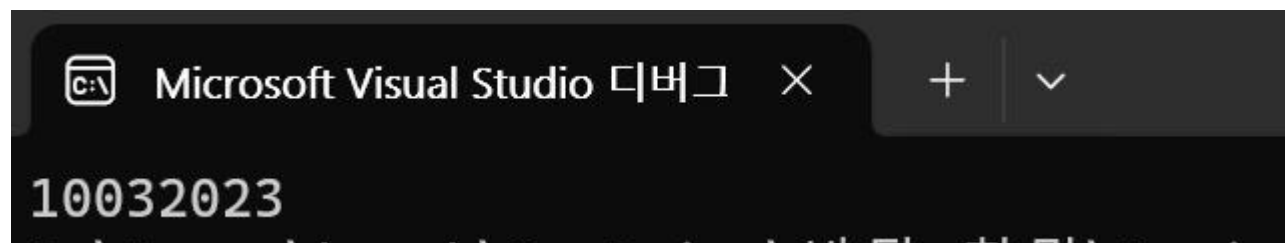
Lab09_4 코드

```
#include "ex_header.h"
```

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

```
int main(void) {  
    printf("%d", extern_test);  
}
```

실습 4 실행 화면.



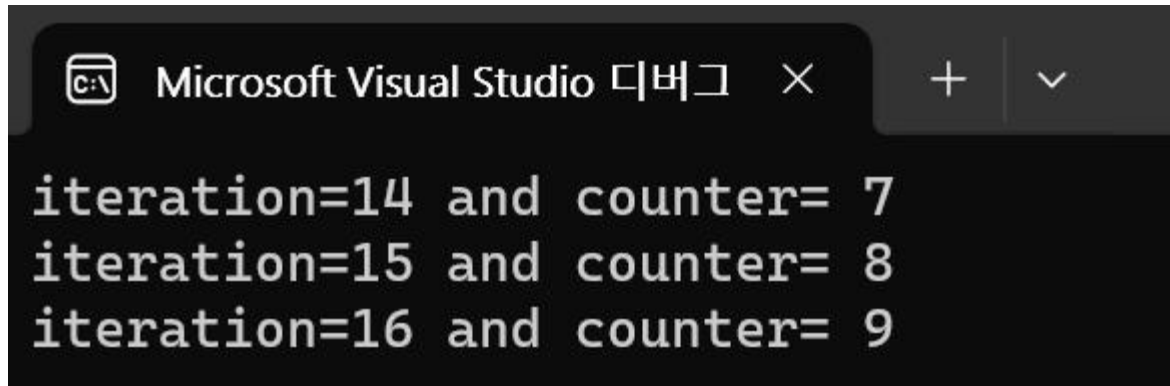
실습 5.

```
#include <stdio.h>
void next(void); /* function declararion */
int counter = 7; /* global variable */

int main() {
    while (counter < 10) {
        next();
        counter++;
    }
    return 0;
}

void next(void) { /* function definition*/
    /* local static variable */
    static int iteration = 13;
    //int iteration = 13;
    iteration++;
    printf("iteration=%d and counter= %d\n", iteration, counter);
}
```

실습 5 실행 화면.



```
iteration=14 and counter= 7
iteration=15 and counter= 8
iteration=16 and counter= 9
```

실습 6.

Lab09_6_1 코드

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

```
//static double x; //"global" static variable
```

```
double x;
```

```
void func() {
```

```
    x = 12345;
```

```
    printf("x = %lf\n", x); /* uses "static double x" */
```

```
}
```

Lab09_6_1 코드

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

```
extern double x;
```

```
extern void func();
```

```
int main() {
```

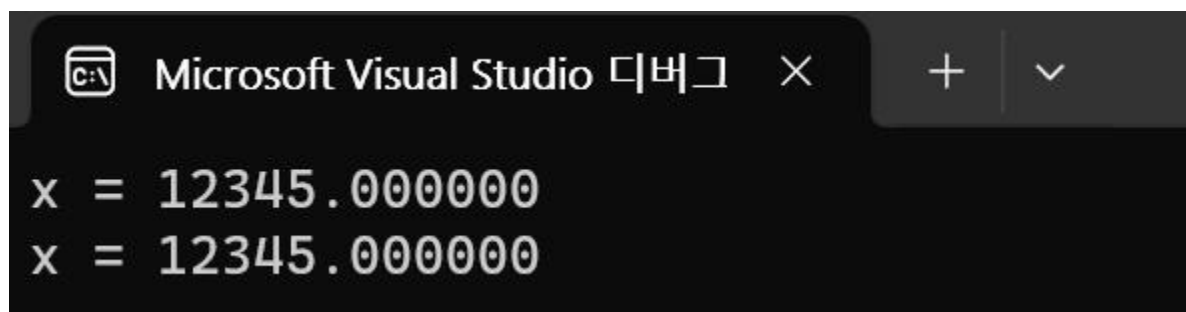
```
    func();
```

```
    printf("x = %lf\n", x); /* Try to uses "static double x" */
```

```
    return 0;
```

```
}
```

실습 6 실행 화면.



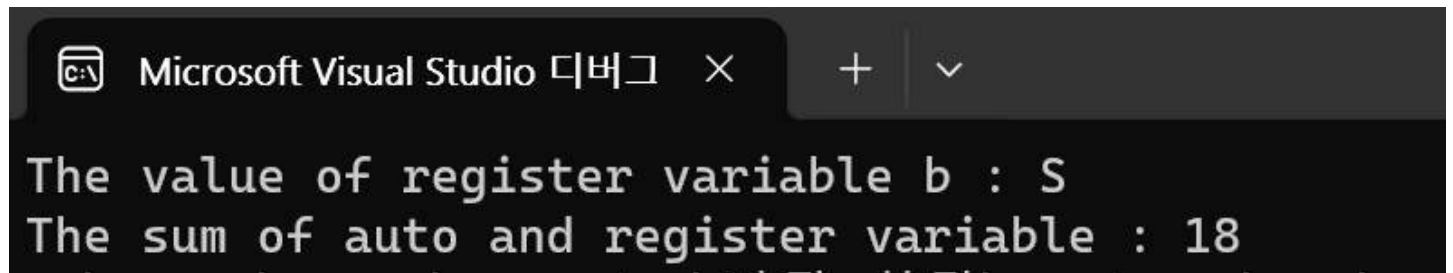
The screenshot shows the Microsoft Visual Studio debug console window. The title bar reads "Microsoft Visual Studio 디버그" with a close button. The console output displays two lines: "x = 12345.000000" and "x = 12345.000000".

실습 7.

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

```
int main() {  
    register char x = 'S';  
    register int a = 10;  
  
    auto int b = 8;  
    printf("The value of register variable b : %c\n", x);  
    printf("The sum of auto and register variable : %d", (a + b));  
    return 0;  
}
```

실습 7 실행 화면.

A screenshot of the Microsoft Visual Studio debug console. The title bar at the top shows the Visual Studio icon, the text "Microsoft Visual Studio 디버그", and window control buttons. The console output displays two lines of text in a monospaced font: "The value of register variable b : S" followed by a newline, and "The sum of auto and register variable : 18".

```
Microsoft Visual Studio 디버그 × + ▾  
  
The value of register variable b : S  
The sum of auto and register variable : 18
```