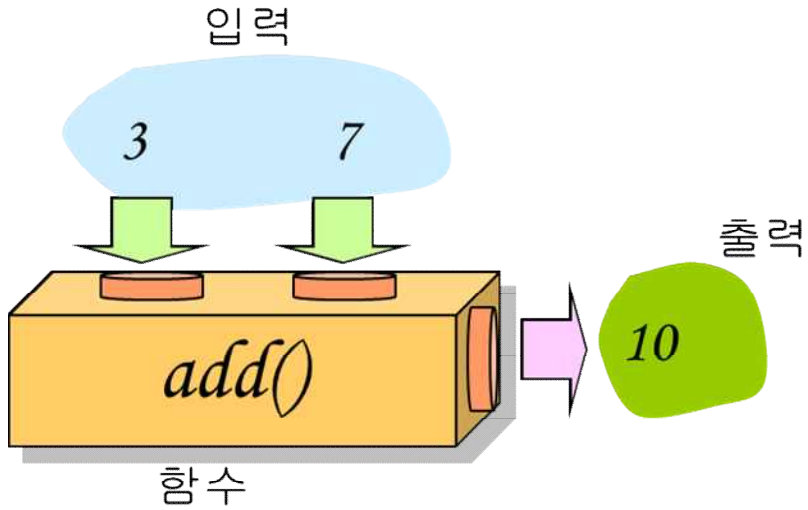
함수

함수

❖ 정의

↳ 하나의 독립적인 기능을 수행하는 별도의 모듈이다.

↳ 입력을 받아서 특정한 작업을 수행하여서 결과를 반환하는 블랙 박스(상자)와 같다.



❖ 적용

↳ 독립적 기능을 수행하는 블록을 별도의 함수로 분리

↳ 동일한 코드가 여러 곳에서 사용될때 동일한 코드를 하나의 함수로 분리

함수

❖ 필요성 및 장점

- 함수를 사용하면 코드가 **중복**되는 것을 막을 수 있다.
- 한번 작성된 함수는 여러 번 **재사용**할 수 있다.
- 함수를 사용하면 전체 프로그램을 **모듈로 분리**
- 개발 과정이 쉬워지고 보다 체계적이 되면서 **유지보수**도 쉬워진다.

❖ 종류

함수

❖ 함수의 정의

Syntax 함수 정의

예

반환형 함수 이름 매개 변수(현재는 없다)

```
void print_stars()
{
        for(int i=0; i<30; i++)
                printf("*");
}
```

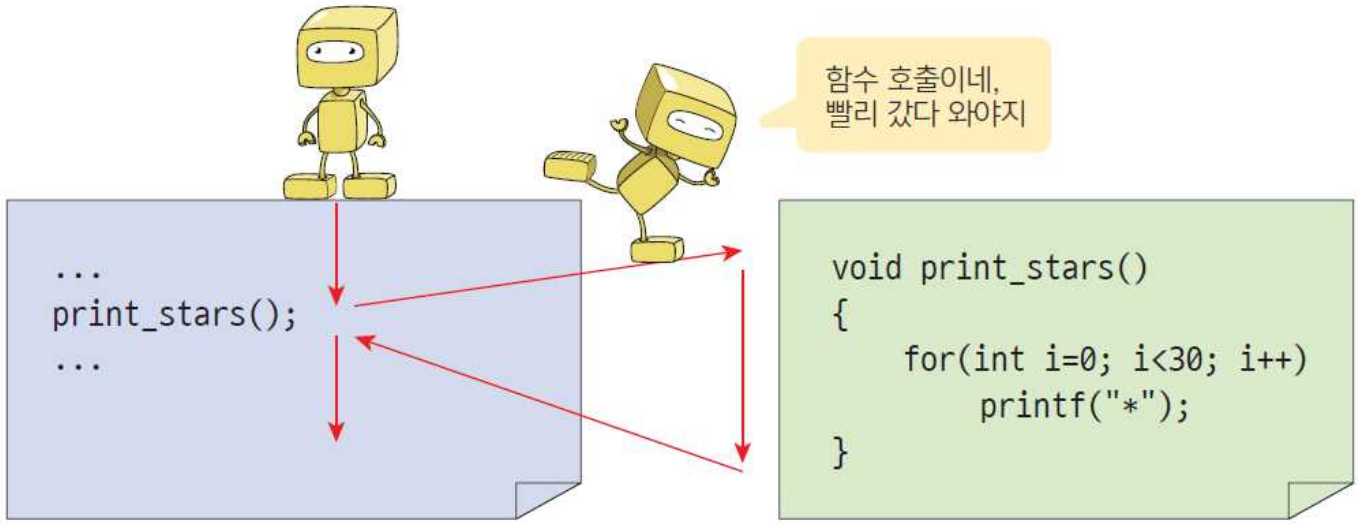
함수 몸체

↳ **반환형**은 함수가 처리를 종료한 후에 호출한 곳으로 반환하는 데이터의 유형을 말한다.

↳ **함수 이름**은 식별자에 대한 규칙만 따른다면 어떤 이름이라도 가능하다.

↳ **함수 이름**은 함수의 기능을 암시하는 **(동사+명사)**를 사용하면 좋다

❖ 함수의 호출



↳ 함수 안의 문장들은 호출되기 전까지는 전혀 실행되지 않는다.

↳ 함수를 호출하게 되면 현재 실행하고 있는 코드는 잠시 중단되고,
호출된 함수로 이동하여서 함수 몸체 안의 문장들이 순차적으로 실행된다.

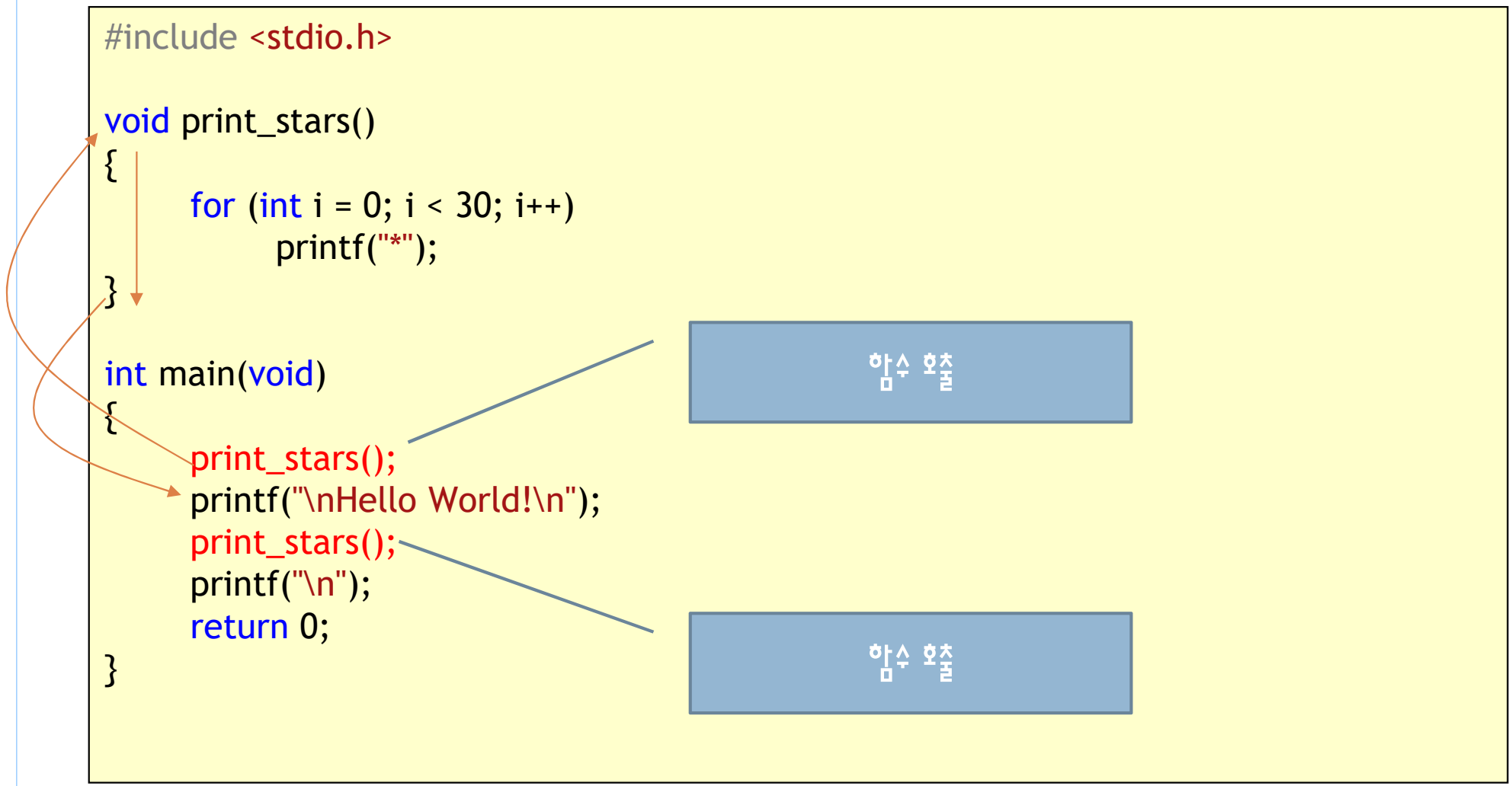
↳ 호출된 함수의 실행이 끝나면 호출한 위치로 되돌아가서 잠시 중단되었던 코드가 실행을 재개한다.

함수

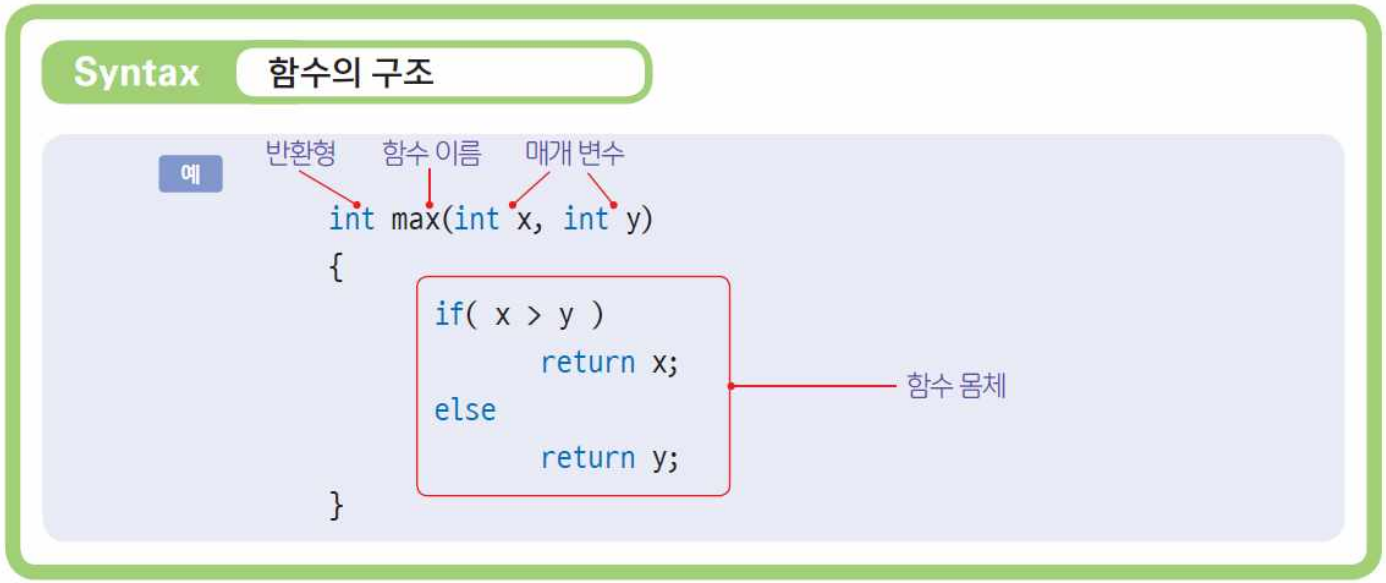
```
#include <stdio.h>

void print_stars()
{
        for (int i = 0; i < 30; i++)
                printf("*");
}

int main(void)
{
        print_stars();
        printf("\nHello World!\n");
        print_stars();
        printf("\n");
        return 0;
}
```



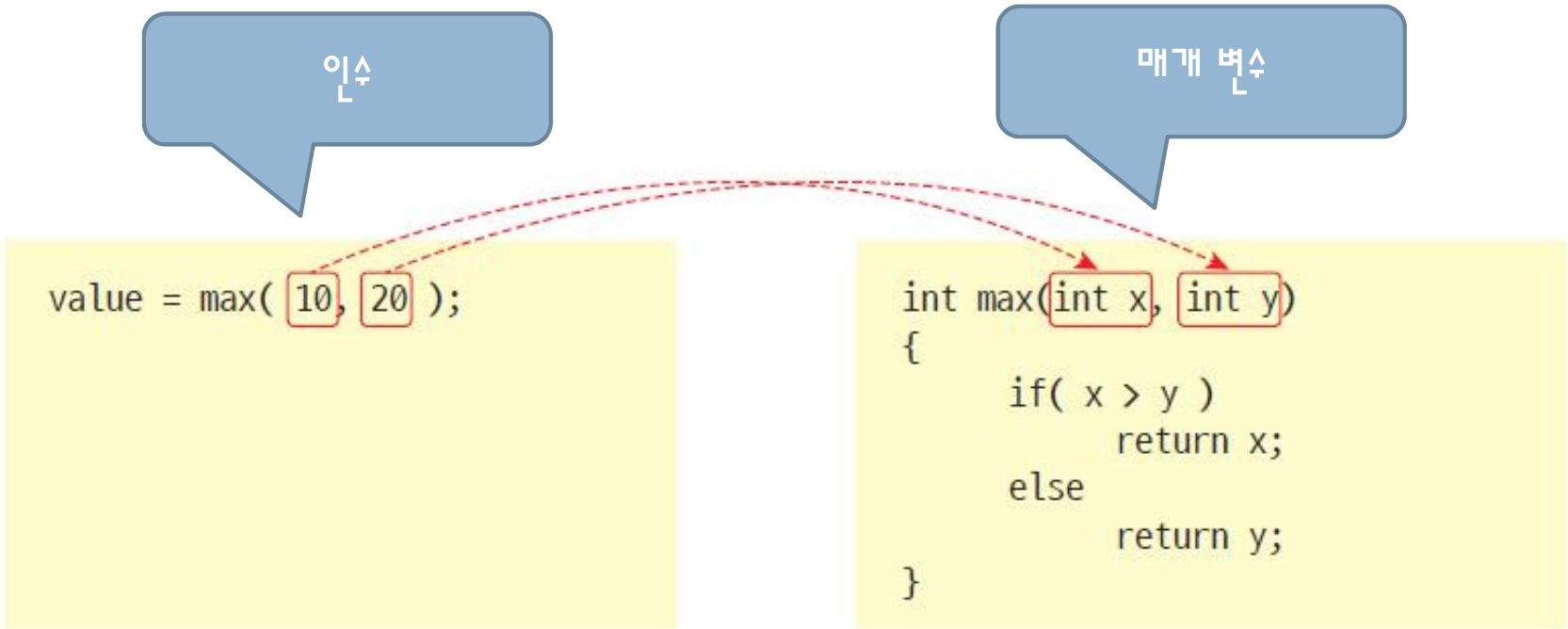
❖ 인수와 매개변수



↳ 인수(argument)는 호출 프로그램에 의하여 함수에 실제로 전달되는 값이다.

↳ 매개 변수(parameter)는 이 값을 전달받는 변수이다.

❖ 인수와 매개변수



↳ 인수(argument)는 호출 프로그램에 의하여 함수에 실제로 전달되는 값이다.

↳ 매개 변수(parameter)는 이 값을 전달받는 변수이다.

함수

❖ 만약 매개 변수가 없는 경우

↳ print_stars(void)와 같이 매개변수 위치에 void를 기술

↳ print_stars( )와 같이 아무 것도 적지 않으면 된다.

↳ 함수가 호출될 때마다 인수는 달라질 수 있다.

↳ 매개 변수의 개수는 정확히 일치하여야 한다는 점이다.

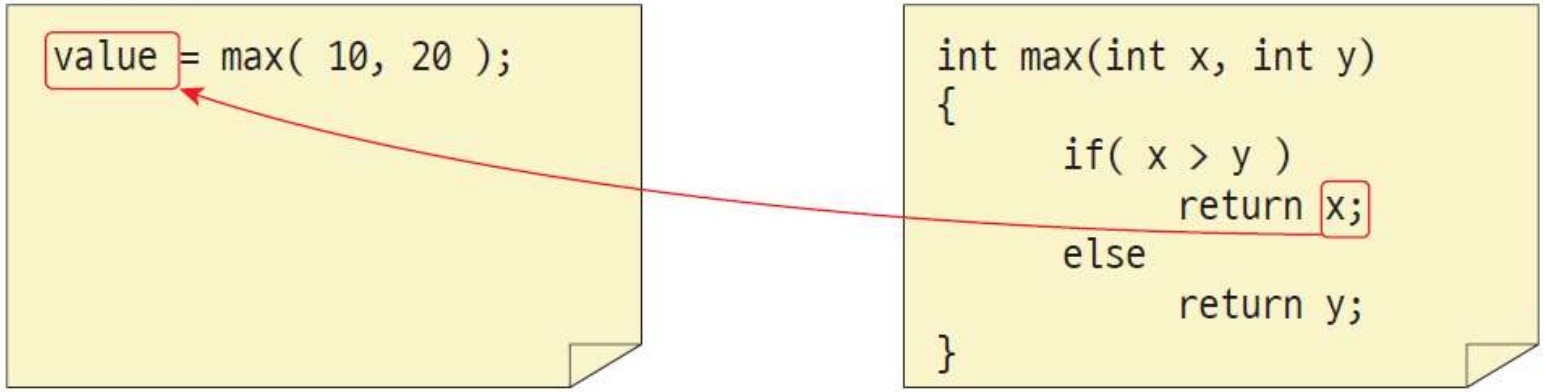
❖ 반환값

↳ 반환값(return value)은 함수가 호출한 곳으로 반환하는 작업의 결괏값이다.

↳ 값을 반환하려면 return 문장 다음에 수식을 써주면 수식의 값이 반환된다.

↳ 인수는 여러 개가 있을 수 있으나 반환값은 없거나 하나만 가능하다.

함수



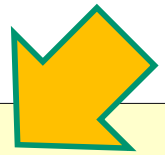
함수

```c
#include <stdio.h>

int main(void)
{
	printf("섭씨 %lf도는 화씨 %lf입니다. \n", 36.0, c_to_f(36.0));
	return 0;
}
double c_to_f(double c_temp)
{
	return 9.0 / 5.0 * c_temp + 32;
}
```

❖ 함수 원형

↳**함수 원형(function prototyping) : 컴파일러에게 함수에 대하여 미리 알리는 것**



```c
#include <stdio.h>
double c_to_f(double c_temp); // 함수 원형

int main(void)
{
	printf("섭씨 %lf도는 화씨 %lf입니다. \n", 36.0, c_to_f(36.0));
	return 0;
}
double c_to_f(double c_temp)
{
	return 9.0 / 5.0 * c_temp + 32;
}
```

❖ 함수 원형

↳ **함수 원형(function prototyping) : 컴파일러에게 함수에 대하여 미리 알리는 것**

```
#include <stdio.h>
double c_to_f(double c_temp); // 함수 원형

int main(void)
{
	printf("섭씨 %lf도는 화씨 %lf입니다. \n", 36.0, c_to_f(36.0));
	return 0;
}
double c_to_f(double c_temp)
{
	return 9.0 / 5.0 * c_temp + 32;
}
```

↳ **함수 원형은 함수의 이름, 매개변수 자료형, 반환형을 함수가 정의되기 전에 미리 알려주는 것이다.**

↳ **매개 변수의 이름은 생략해도 된다.**

```c
#include <stdio.h>
//
int get_integer()
{
	int value;
	printf("정수를 입력하시오: ");
	scanf("%d", &value);
	return value;
}

//
int add(int x, int y)
{
	return x + y;
}

int main(void)
{
	int x = get_integer();
	int y = get_integer();

	int sum = add(x, y);
	printf("두수의 합은 %d입니다. \n", sum);
	return 0;
}
```

함수

```c
#include <stdio.h>

int factorial(int n)
{
	int result = 1;

	for (int i = 1; i <= n; i++)
		result *= i;// result = result * i
	return result;
}

int main(void)
{
	int n;
	printf("알고 싶은 팩토리얼의 값은? ");
	scanf("%d", &n);
	printf("%d!의 값은 %d입니다. \n", n, factorial(n));
	return 0;
}
```

$$n! = n*(n-1)*(n-2)* \ldots *1$$

함수

$$C(n,r) = \frac{r!}{(n-r)!r!}$$

```
// 팩토리얼 값을 이용하여서 조합값을 계산
int combination(int n, int r)
{
      int iResult;

      iResult = factorial(n)/(factorial(r) * factorial(n-r));
      return (iResult);
}
```

```
int main(void)
{
        int a, b;
        a = get_integer();
        b = get_integer();

        printf("C(%d, %d) = %d \n", a, b, combination(a, b));
        return 0;
}
```

함수

```
int is_prime(int n)
{
        int i;

        for (i = 2; i < n; i++) {
            if (n%i == 0)
                return 0;
        }
        return 1;
}
```

```
int is_prime(int n)
{
    int i, iResult=1;

    for (i = 2; i < n; i++) {
        if (n%i == 0)
            iResult = 0;
    }
    return iResult;
}
```

```
int main(void)
{
        int n;
        n = get_integer();

        if (is_prime(n) == 1)
                printf("%d은 소수입니다.\n", n);
        else
                printf("%d은 소수가 아닙니다.\n", n);
        return 0;
}
```

Do You Have Any Questions?

We would be happy to help.