

함수의 구조

■ 함수의 기본형태

- 함수는 ‘매개변수(또는 ‘인수’)'를 **입력**받아 가공하고 처리한 후 ‘**반환값**’을 돌려줌

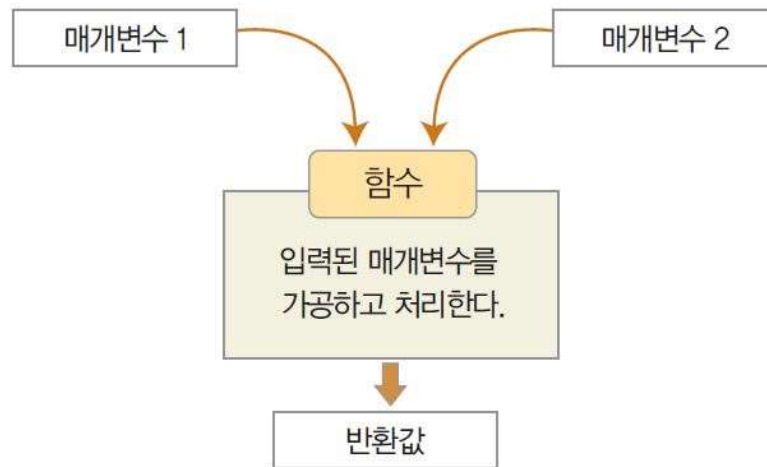


그림 10-3 함수의 형태

- | 반환값
데이터유형 | 함수이름 | (매개변수) |
|---------------------------------|------|----------|
| <pre>{ 함수본문 반환값 }</pre> | | |

예제1.

```
double square (int num)
{
    return (double) num*num;
}
```

예제2.

```
void printResult ( )
{
    printf("Test Result\n");
}
```

함수 선언 위치

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

int absolutevalue(int n1, int n2) {
    int rslt = n1 - n2;
    if (n1 - n2 > 0) {
        rslt = n1 - n2;
    }
    else {
        rslt = n2 - n1;
    }

    return rslt;
}

int main() {

    int w1 = 10, w2 = 25, rslt = 0;
    rslt = absolutevalue(w1, w2);
    printf("내가 만든 절대값 함수: %d \n", rslt);
}
```

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

int absolutevalue(int n1, int n2);

int main() {

    int w1 = 10, w2 = 25, rslt = 0;
    rslt = absolutevalue(w1, w2);
    printf("내가 만든 절대값 함수: %d \n", rslt);
}

int absolutevalue(int n1, int n2) {
    int rslt = n1 - n2;
    if (n1 - n2 > 0) {
        rslt = n1 - n2;
    }
    else {
        rslt = n2 - n1;
    }

    return rslt;
}
```

main() 함수 **위에** 사용자 정의 함수를 선언하거나,
main() 함수 **아래에** 선언하는 경우에는 main() 함수 **위에 정의**를 써준다.

main() 도 함수

int main()

1. 입력받는 매개변수는 없고, **int**형 데이터를 반환한다.
2. **main()** 함수의 종료 시점에 명시적으로 **return 0;** 를 쓰기도 한다.
3. 특정 시점에 프로그램을 종료시키기 위해 **return 0;** 를 활용하기도 한다.

함수의 필요성

- ▶ 함수를 사용하면 코드가 중복되는 것을 막을 수 있다.
- ▶ 한 번 작성된 함수는 반복하여 재사용할 수 있다.
- ▶ 함수를 사용하면 코드의 가독성이 높아진다.
- ▶ 함수를 사용하면 전체 프로그램을 더 작은 단위로 나눌 수 있어서 개발 과정이 쉬워지고 보다 체계적이 되면서 유지 보수도 쉬워진다.

[10-4] 더하는 기능을 하는 함수 만들기

```
01 #include <stdio.h>
02
03 int plus(int v1, int v2)
04 {
05     int result;
06     result = v1 + v2;
07     return result;
08 }
09
10 int main()
11 {
12     int hap;
13
14     hap = plus(100, 200);
15
16     printf("100과 200의 plus() 함수 결과는 : %d\n", hap);
17 }
```

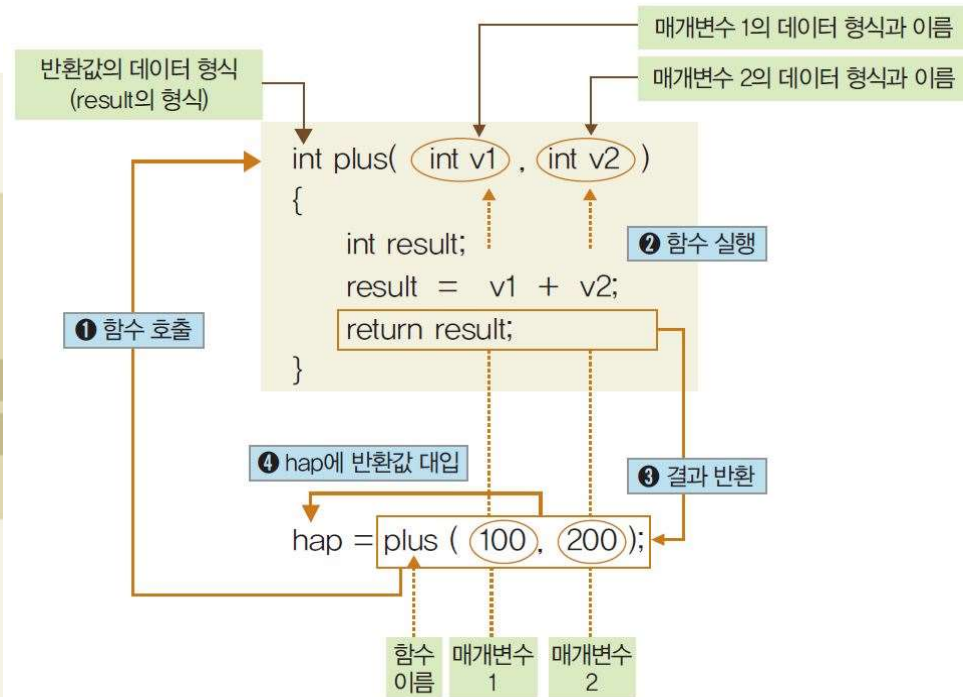
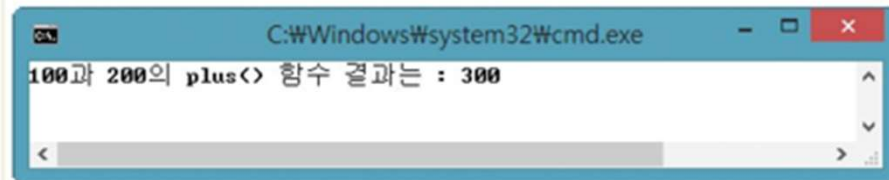


그림 10-4 plus() 함수의 형태와 호출 순서

---매개변수 2개를 지정하여 plus() 함수를 호출하고 반환값은 hap에 저장한다.

실행 결과 ▼



[10-5] calc 함수 선언하기/호출하기

- 정수형을 반환하는 calc 라는 이름을 갖는 함수를 정의한다.
- calc 함수의 매개변수는 정수형 숫자 세 개를 받는다.
 - 이 때 정수 두 개는 더할 숫자이고,
 - 나머지 정수 한 개는 연산자 종류를 선택하는 숫자로 사용한다.
- calc 함수 내부에서는 switch문을 이용해서 +, -, *, / , % 연산을 하고, 결과를 반환하도록한다.
 - 이 때, 숫자 1~5가 각각 +, -, *, / , % 연산을 의미한다.
- main 함수에서 계산할 종류를 다음 구문을 사용해서 입력받는다.
scanf("%d", &op);
- main 함수에서 계산할 두 숫자를 다음 구문을 사용해서 입력받는다.
scanf("%d,%d", &num1, &num2);
- main 함수에서 calc 함수를 호출한 계산 결과를 출력한다.

함수의 반환값에 따른 함수 구분 (1/2)

1. 반환값이 있는 함수

- 함수를 실행한 결과값은 함수의 데이터형을 따름
- 'int 함수 이름()'으로 정의했다면 결과도 정수형 변수나 정숫값이어야 함.
 - 'return 정수형 변수;' 또는 'return 정수;'로 표현해야 함

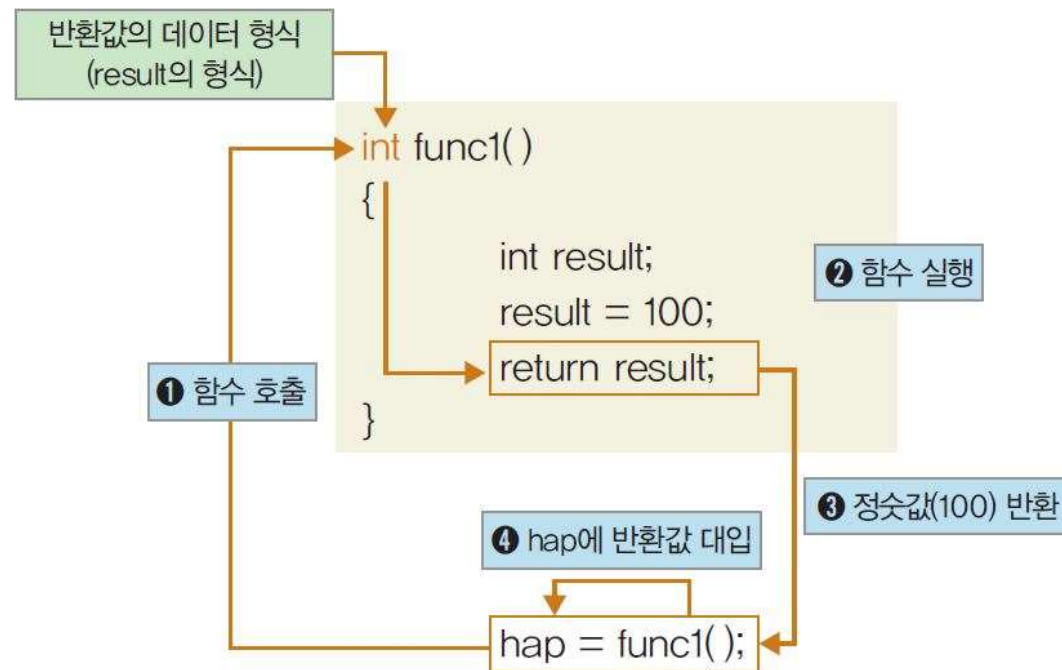


그림 10-8 int 형 값의 반환

함수의 반환값에 따른 함수 구분 (2/2)

2. 반환값이 **없는** 함수

- 함수를 실행한 결과로 돌려줄 것이 없는 경우
- 함수의 데이터형을 void로 표시 : void 형 함수를 호출할 때는 함수 이름만 쓴다.

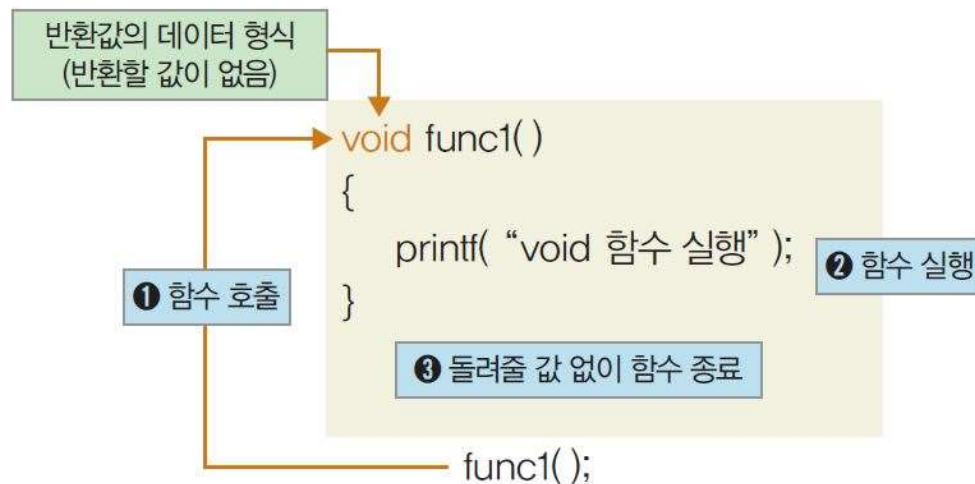


그림 10-9 void 형 함수의 작동

[10-7] 반환값 유무에 따른 함수비교

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

```
02
```

```
03 void func1()
```

---void 형 함수이므로 반환값이 없다.

```
04 {
```

```
05     printf("void 형 함수는 돌려줄게 없음.\n");
```

```
06 }
```

```
07
```

```
08 int func2()
```

---int 형 함수므로 반환값이 있다.

```
09 {
```

```
10     return 100;
```

```
11 }
```

```
12
```

```
13 int main()
```

```
14 {
```

```
15     int a;
```

```
16
```

```
17     func1();
```

---void 형 함수를 호출한다.

```
18
```

```
19     a = func2();
```

---int 형 함수를 호출한다.

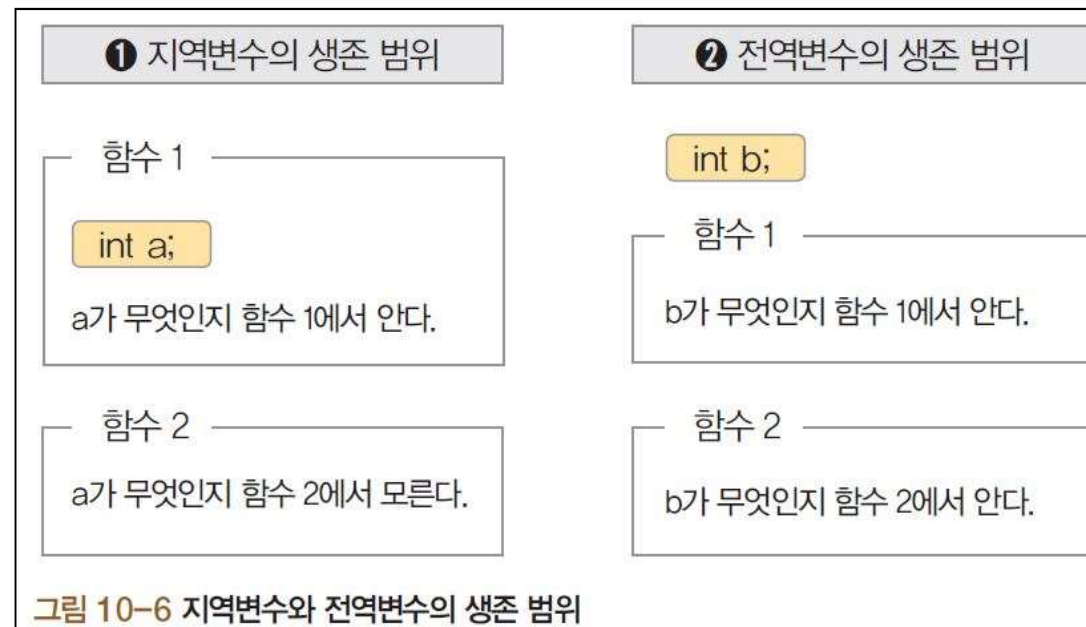
```
20     printf("int 형 함수에서 돌려준 값 == > %d\n", a);
```

```
21 }
```

변수의 생존 범위

■ 지역변수 vs. 전역변수

- 지역변수 : 한정된 지역(local)에서만 사용되는 변수
- 전역변수 : 프로그램 전체(global)에서 사용되는 변수



- ①에서 a는 '함수 1' 안에 선언한 지역변수
- ②의 b는 전역변수로, 함수(함수 1, 함수 2) 안이 아니라 함수 바깥에 선언하면 모든 함수에서 b의 존재를 알 수 있음
- 예를 들어, ①에서 함수 1과 함수 2에서 각각 a를 선언하면?
같은 a라고 해도 각각의 함수에서 따로 정의하므로 서로 다른 a이다.

Summary

1. 함수의 이해

- ❶ 어떤 값이 들어가면 그것을 처리한 후 하나의 결과값을 돌려준다.
- ❷ 간단히 '함수 이름()' 형식으로 사용한다.
- ❸ 함수는 반복적인 것을 처리할 때 유용하다

2. 함수를 정의하고 호출하는 예

```
int plus (int v1, int v2)
{
    int result;
    result = v1 + v2;
    return result;
}

hap = plus (100, 200);
```

- 함수선언하기
 - 반환값의 데이터형식
 - 함수이름
 - 매개변수 유형 및 이름
- 함수선언 위치
 - main 함수 위에 선언하기
 - main 함수 아래에 선언하면, main 함수 위에 정의부를 한 번 써줘야함
- 함수호출하기
 - 함수 이름과 매개변수유형을 맞추어 사용

3. 지역변수와 전역변수

지역변수는 선언된 함수 안에서만 유효한 변수이고, 전역변수는 모든 범위에서 유효한 변수임

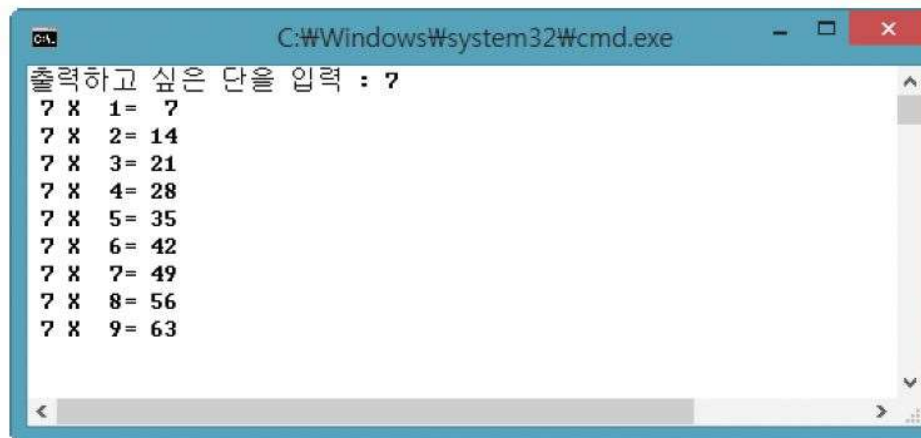
4. 함수의 반환값

- ❶ 함수에서 값을 돌려주기 위해서는 return문을 사용한다.
- ❷ 함수가 돌려줄 값에 따라 함수 이름 앞에 데이터 형식이 붙는다.
- ❸ 돌려줄 값이 없다면 함수를 void 형으로 선언한다.

[실습1] 함수를 이용한 구구단 프로그램

예제 설명 함수를 사용하여 구구단을 출력하는 프로그램이다.

실행 결과



```
C:\Windows\system32\cmd.exe
출력하고 싶은 단을 입력 : 7
7 x 1= 7
7 x 2= 14
7 x 3= 21
7 x 4= 28
7 x 5= 35
7 x 6= 42
7 x 7= 49
7 x 8= 56
7 x 9= 63
```

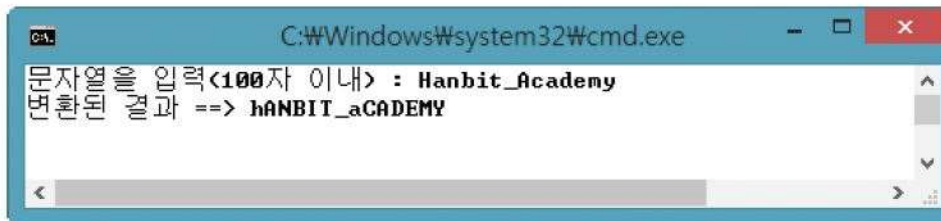
- 반환형이 없는 **gugu**라는 이름을 갖는 함수를 작성하시오
 - **gugu**라는 함수는 정수형 매개변수를 한 개 갖는다.
 - 매개변수로 받은 단을 **for**문을 이용해서 출력하도록 함수 본문을 작성한다.
- **main** 에서 출력할 단을 입력받고, **gugu** 함수를 호출해서 실행시킨다.
 - 출력할 단을 입력받을 때는 다음의 구문을 사용한다. (`scanf("%d", &input);`)

[실습2] 함수를 이용한 대소문자 변환

예제 설명 대문자는 소문자로, 소문자는 대문자로 변환하는 프로그램이다.

- ① 대문자 변환 방법: 소문자에서 대·소문자 차이를 뺀다.
- ② 소문자 변환 함수: 대문자에서 대·소문자 차이를 더한다.

실행 결과



```
C:\Windows\system32\cmd.exe
문자열을 입력<100자 이내> : Hanbit_Academy
변환된 결과 ==> hANBIT_aCADEMY
```

[실습3] 숫자 자동 추출하기

예제 설명 1~45 중에서 숫자 6개를 자동으로 뽑는 프로그램이다.

실행 결과



```
C:\Windows\system32\cmd.exe
** 로또 추첨을 시작합니다. **
추첨된 로또 번호 ==> 8 15 28 44 32 9
```

The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The window contains two lines of text: the first line is "** 로또 추첨을 시작합니다. **" and the second line is "추첨된 로또 번호 ==> 8 15 28 44 32 9". The window has a standard Windows title bar with minimize, maximize, and close buttons.