C 프로그래밍 및 실습

[11주차 1강]
함수(2)

학습 내용

8.3 함수 호출과 반환

학습 목표

8.3 함수 호출 과정을 이해하고,
함수를 호출하고 반환하는 방법을 익힌다.

C 프로그래밍 및 실습

8.3 함수 호출과 반환

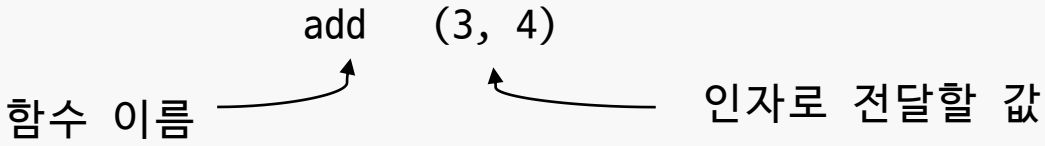
세종대학교

함수 호출(사용) 방법

- 함수 이름을 쓰고, 소괄호 안에 함수 인자에 넣을 값을 차례로 적음

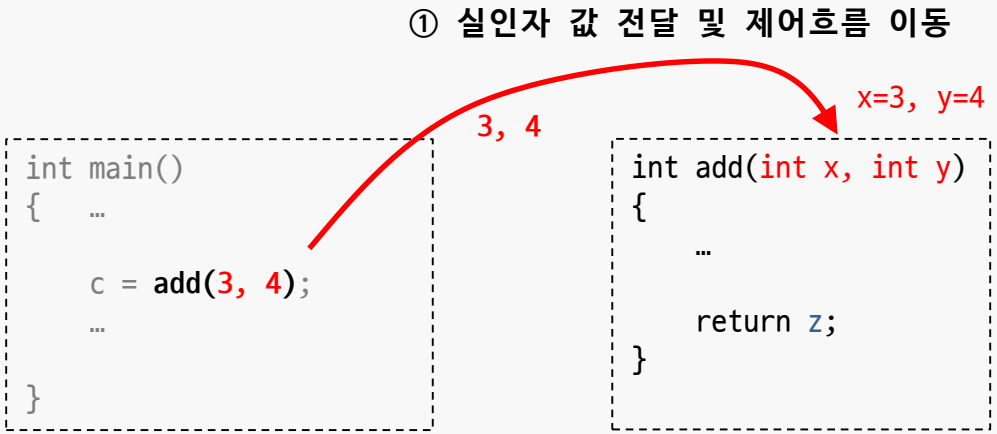
```
int add(int x, int y)
{
    int z;

    z = x+y;

    return z;
}
```

```
int main()
{
    int c;

    c = add(3, 4); // 함수 호출부

    return 0;
}
```



함수 호출 과정

① **add() 함수 호출** : 인자 값 3과 4가 add 함수에 전달되고, 프로그램의 제어는 add()함수로 넘어감



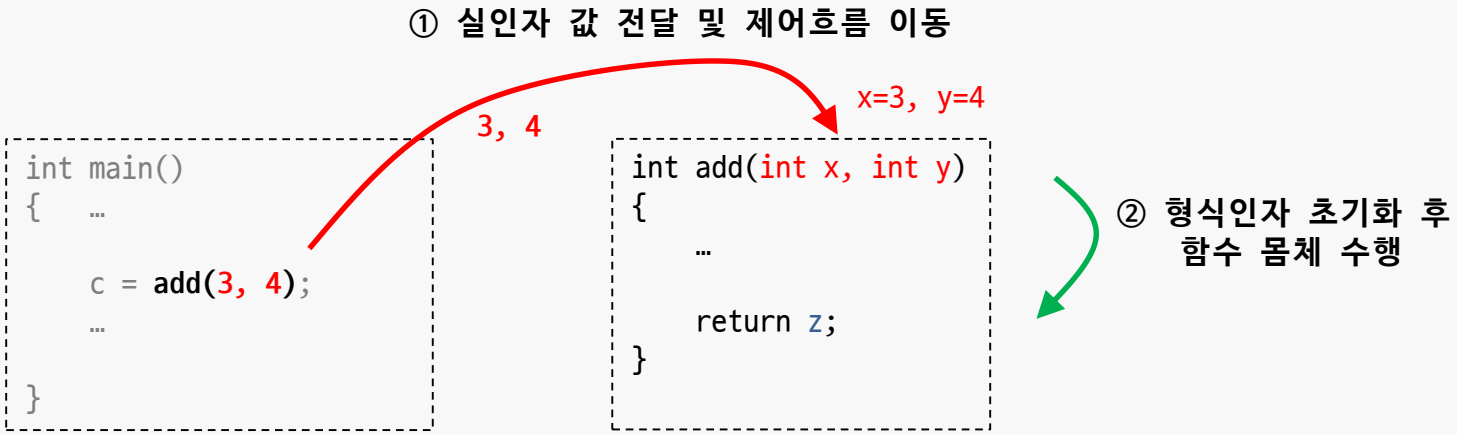
세종대학교

함수 호출 과정

② **add() 함수 수행** : 인자를 x=3, y=4로 초기화(대입) 한 후, 몸체 수행

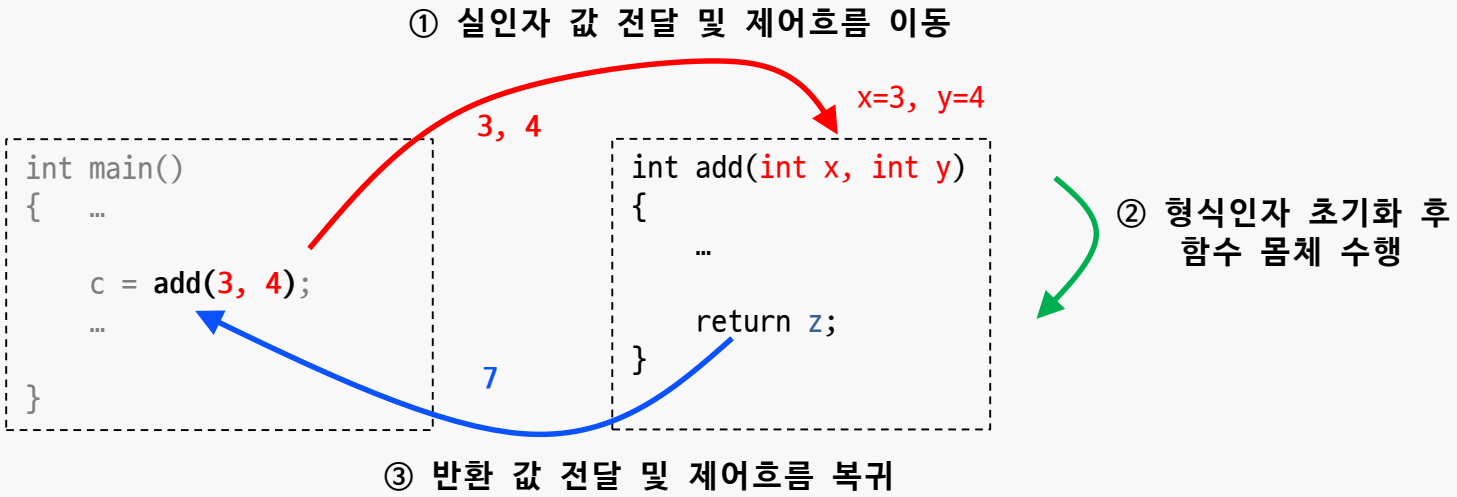
✓형식인자 : add()함수의 정의에 사용된 x, y

✓실 인자 : add()함수 호출 시 넘겨 받는 값(3,4)



함수 호출 과정

③ **add() 함수 종료** : 프로그램 제어는 함수를 호출했던 라인(c=add(3,4))으로 복귀
(반환값을 사용해 나머지 부분 수행)



함수 호출 과정 확인

- 다음과 같이 각 함수의 시작과 끝에 printf문을 삽입하여
  함수 호출 시 제어흐름과 값이 전달되는 과정을 확인해보자.
  - ✓ 함수 인자 전달과 반환이 제대로 수행되고 있는 지 점검하는 기본적인 방법

```
int add(int x, int y)
{
   int z;
   printf("add start: x=%d,y=%d\n",
            x,y);

   z = x + y;

   printf("add end: z=%d\n", z);
   return z;
}
```

```
int main()
{
   int c = 0;
   printf("main start: c=%d\n", c);

   c = add(3, 4); // 함수 호출부

   printf("main end: c=%d\n", c);
   return 0;
}
```

세종대학교

함수 호출의 다양한 형태

- 실인자의 다양한 형태

```
int add(int x, int y)
{
    …
    return z;
}
```

```
int main()
{
    int a=4, b=3;
    int v1,v2,v3,v4,sum;

    v1 = add(a, a+b);
    v2 = add(1, a+2);
    v3 = add(1+2, a) - 3;
    sum = add(1, b)+add(a, 2);
    v4 = add(a, add(1, 2));

    return 0;
}
```

세종대학교

함수 호출의 다양한 형태

- main( ) → func( ) → add( )

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

```
int func(int a, int b)
{
   int z = add(a,b);

   if(z > 0) return 1;
   if(z < 0) return -1;
   return 0;
}
```

return 문 여러 개 사용됨

```
int main() {
   int c;

   c = func(1,2) ;
   printf("c = %d\n",c);

   return 0;
}
```

세종대학교

printf문을 이용한 함수 호출 과정 확인

- 초기화 등 코드 약간 수정

```
int add(int x, int y) {
    int z=0;
    printf(…);
    z = x+y;
    printf(…);
    return z;
}

int func(int a, int b) {
    int z = 0;
    printf(…);
    z = add(a,b);

    if(z > 0){
        printf(…);
        return 1;
    }
    if(z < 0){
        printf(…);
        return -1;
    }
    printf(…);
    return 0;
}

int main()
{
    int c=-1;
    printf(…);
    c = func(1,2);
    printf(…);
    return 0;
}
```

실습 하기

[예제 8.2] (1~2번)
예제 8.1에서 정의된 함수를 이용하여 다음 프로그램을 작성하시오.

1. 문자 'a' 는 한 번, 문자 'b' 는 두 번, …, 문자 'z'는 26번 출력
   - ✓각 문자별 한 줄에 하나씩 출력
   - ✓print_characters 함수를 반복 호출

실행 결과

```
a
bb
ccc
dddd
... (생략)
```

실습 하기

[예제 8.2] (1~2번)
예제 8.1에서 정의된 함수를 이용하여 다음 프로그램을 작성하시오.

2. 4개의 정수 a, b, c, d를 입력 받아, 최댓값 출력
   ① a와 b 중 큰 값을 찾기 위해 max 함수 호출
   ② c와 d 중 큰 값을 찾기 위해 max 함수 호출
   ③ 위 두 결과값 중 큰 값을 찾기 위해 max 함수 호출

입력 예시1

```
5 2 9 4
```

출력 예시1

```
9
```

✓ **(추가)** max함수의 인자에서 max함수를 호출하는 방식으로, 하나의 문장(수식)만을 사용하여 최댓값을 찾도록 프로그램을 수정해 보시오.

함수 원형 선언 (함수 선언)

- add() 함수 정의를 main() 함수 뒤에 작성하면?

```
int main()
{
   . . .
   c = add(3,4);
   . . .
}

int add (int x, int y)
{
   . . .
}
```

오류 발생
add()가 정의 되지 않았습니다.

- 컴파일 오류 or 경고 발생
- **함수 정의**를 함수 호출 위치보다 앞에 작성 해주어야 함
- 다수의 함수를 정의해 사용할 때, 함수 호출 순서에 맞추어 함수를 배치하는 것은 불편

세종대학교

함수 원형 선언 (함수 선언)

- 이 문제를 해결하는 방법은?

```
int add (int x, int y);  ← 함수 원형 선언

int main(){
    . . .
    c = add(3,4);  ← 정상 동작
    . . .
}

int add (int x, int y){   ← 함수 정의
    . . .
}
```

- 함수의 형태를 표현하는 **함수 원형**을 앞 부분에 **선언**
- 인자가 두 개의 int형 변수이고, 반환형이 int형인 add() 함수가 어딘가 정의 되어있다는 것을 알려주는 (선언) 역할
- 용어 **'선언'** 과 **'정의'**를 혼동하지 말자

세종대학교

함수 원형 선언 (함수 선언)

- 함수의 헤더와 동일한 형태를 가지는데, 마지막에 세미콜론(;)을 붙여줌

```
반환형  함수이름 ( 인자선언1, 인자선언2,…) ;
```

- 함수의 형태를 지정해 주는 것이므로, 인자 선언에서 변수명은 무시됨
  - ✓ 인자 이름을 생략해도 되고, 심지어 함수 정의에 사용된 변수명과 다른 변수명 명시하는 것도 가능
  - ✓ 아래 세 선언은 모두 동일

```
int add(int x, int y);
int add(int a, int b);
int add(int, int);     → 권장 형태
```

실습 하기

[예제 8.2] (3~4번)
예제 8.1에 정의된 함수를 이용하여 다음 프로그램을 작성하시오.
(함수 원형 선언 사용)

3. 영문자 10개를 입력 받아, 모두 대문자로 변환하여 출력
   ✓ 소문자를 대문자로 변환하기 위해 atoA 함수를 반복 호출

입력 예시1

```
HellOWorLd
```

출력 예시1

```
HELLOWORLD
```

실습 하기

[예제 8.2] (3~4번)
예제 8.1에 정의된 함수를 이용하여 다음 프로그램을 작성하시오.
(함수 원형 선언 사용)

4. 6개의 정수 a, b, c, d, e, f를 입력 받아, 다음 수식의 결과를 출력하시오.

a/b + c/d + e/f (실수 연산 수행)

- ✓ 분수를 계산하기 위해 divide 함수 호출
- ✓ 덧셈을 계산하기 위해 add3 함수 호출

입력 예시1

```
5 2 9 4 6 4
```

출력 예시1

```
6.250000
```

- ✓ **(추가) add3() 함수의 인자에서 divide() 함수를 호출하는 방식으로, 하나의 문장(수식)으로 수식의 결과 값이 계산되도록 프로그램을 수정해 보시오.**

학습 정리

- 함수를 호출하면 프로그램 **제어**는 호출된 함수로 넘어가고, 호출된 함수가 종료되면 다시 호출한 위치로 돌아옴
- 함수에 전달된 값을 **실인자**라 하고, 전달된 값을 저장하기 위한 변수를 **형식인자**라 함
- 함수의 **원형 선언**은 함수의 형태를 미리 선언하는 것으로, 보통 프로그램 앞 부분에 나열함