함수

함수정의

↳ 함수를 만드는 것

```
반환형 함수명(매개변수 1, ..., 매개변수 n)  ← 함수 원형
{
    // 함수가 수행할 명령
}
```

반환형

↳ 함수가 기능을 수행 한 후 호출한 곳으로 돌려줄 값의 자료형

매개변수

↳ 함수가 처리할 데이터를 저장하는 변수

함수호출

함수는 호출을 통해 실행된다.

ex) result = sum(a,b); // sum 함수 호출

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

인수

함수를 호출할 때는 함수에 필요한 데이터를 괄호안에 넣어준다. 이를 인수(argument)라 한다

ex) sum(30, 40) sum(a*2, b/3)

(30, 40): 상수인수 / (a*2, b/3): 수식인수

↳ 매개변수에 복사됨

함수 반환

함수는 return문을 사용하여 값을 반환한다.

```
int main(void)
{
  ...
  result = sum(a, b);
  ...
}
```

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

반환

※ 컴파일러는 함수를 호출할 때 반환값을 저장할 공간을 미리한다.

⇒ 반환값을 식의 일부로 사용 가능

함수선언

ex) `int sum(int, int);` // 매개변수 이름은 생략 가능

Why!! 함수정의가 있는데 선언을 해주어야 하는가?

reason 1. 함수 선언에서 반환값을 확인하여 컴파일러에 알릴 수 있다.

reason 2. 호출 형식에 문제가 없는지 확인 가능하다.

※ 호출 이전에 정의가 되면 선언이 없어도 호출이 가능하다.

여러함수유형

형태	구분	설명
매개변수가 없는 경우	선언	ex) int get_num(void); or int get_num();
	특징	호출 시, 인자 없이 괄호만 사용한다.
반환형이 없는 경우	선언	ex) void print_char(char ch);
	특징	return문이 없거나 return만 쓴다.
매개변수와 반환형이 모두 없는 경우	선언	ex) void print_title(void);
	특징	앞의 특징 모두 포함

* 반환형이 없는 함수는 호출 문장을 식의 일부로 사용이 불가능 *

재귀함수

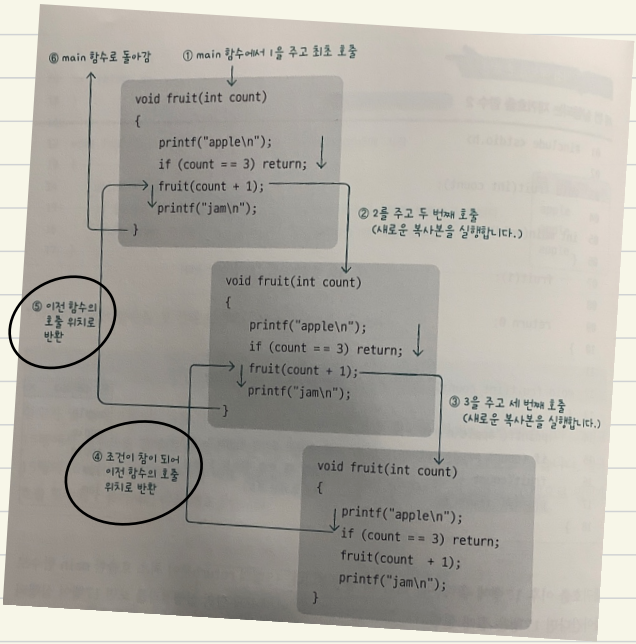
재귀함수 (recursive call function) : 자기 자신을 호출하는 함수

```
ex) void fruit(int count);
    {
      printf("apple\n");
      if(count == 3) return;  // * 반드시 조건식을 포함하여 무한 반복을 막아야 한다. *
      fruit(count+1); // 자기 자신 호출
    }
```

Why!! 그냥 반복문을 쓰지 왜 재귀함수를 사용할까?

⇒ ★ 재귀함수는 직전에 호출한 곳으로 돌아간다 ★

∴ 반복문과는 차이가 있다.



결과:
apple
apple
apple
jam
jam