-Part3-

제3장 콘솔 입출력과 파일 입출력

학습목차

3.1 스트림이란

3.1 스트림이란

▶배울 내용

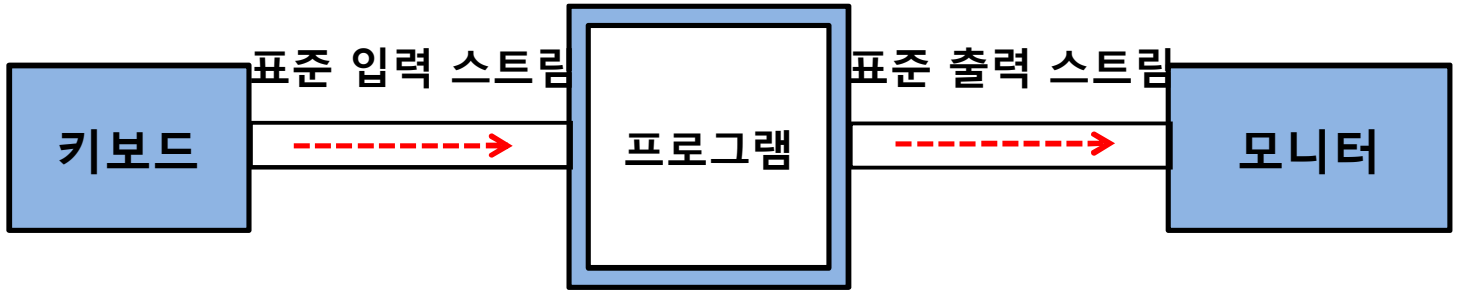
① 스트림

② 버퍼와 버퍼링

3.1 스트림이란 (1/5)

▶ 스트림(Stream)

- ✓ 데이터를 입력하고 출력하기 위한 다리
- ✓ 키보드로 데이터를 입력 ➔ **표준 입력 스트림**
- ✓ 모니터로 데이터를 출력 ➔ **표준 출력 스트림**



3.1 스트림이란 (2/5)

▶ 표준 입출력 스트림

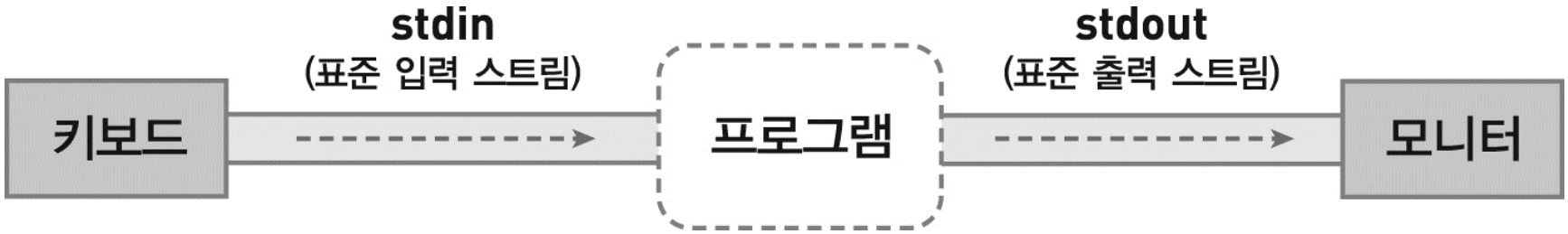
스트림	설명	장치
stdin	표준 입력을 담당	키보드
stdout	표준 출력을 담당	모니터
stderr	표준 에러을 담당	모니터

▶ 표준 입출력 스트림의 생성과 소멸

- ✓ 생성 : 프로그램 시작 시
- ✓ 소멸 : 프로그램 종료 시

3.1 스트림이란 (3/5)

▶ 표준 입출력 스트림



▶ 파일 입출력 스트림

3.1 스트림이란

▶배울 내용

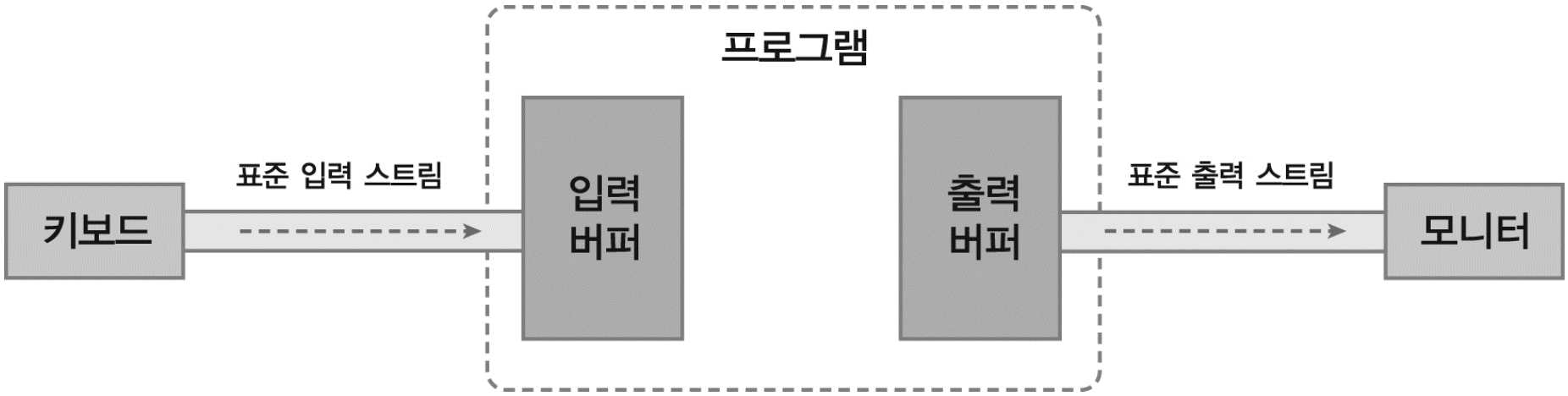
① 스트림

② 버퍼와 버퍼링

3.1 스트림이란 (4/5)

▶ 버퍼(Buffer)

✓ 처리할 데이터를 임시로 저장하는 장소



✓ **입력 버퍼** : 입력 데이터를 저장하기 위한 버퍼

✓ **출력 버퍼** : 출력 데이터를 저장하기 위한 버퍼

3.1 스트림이란 (5/5)

▶ 버퍼링

✓ **버퍼를 채우는 동작**

✓ 버퍼링을 하는 **이유**

- **성능 향상**을 위해...
- 문자를 개별 처리하는 것보다 **문자들을 모아 일괄 처리**하는 것이 효율적

3.2 콘솔 입출력

3.2 콘솔 입출력

▶배울 내용

① 콘솔 표준 입출력 함수

② 콘솔 비표준 입출력 함수

3.2 콘솔 입출력 (1/4)

▶ 콘솔 입출력 함수의 종류

✓ 헤더파일 : **stdio.h**

함수의 원형	기능	헤더파일
int **getchar** (void);	키보드로 부터 한 문자를 입력 받는다.	stdio.h
int **putchar** (int c);	모니터에 한 문자를 출력한다.	stdio.h
char* **gets** (char *s);	키보드로 부터 문자열을 입력 받는다.	stdio.h
int **puts** (char* str);	모니터에 문자열을 출력한다.	stdio.h
int **scanf** (const char* format, ...);	키보드로 부터 데이터를 서식에 맞춰 출력한다.	stdio.h
int **printf** (const char* format, ...);	모니터에 데이터를 서식에 맞춰 출력한다.	stdio.h

3.2 콘솔 입출력 (2/4)

▶ getchar( ) 함수

✓ **문자를 입력하는 함수**

- **호출이 실패**했을 때 **EOF 반환**
- 일일이 EOF 반환 여부를 검사할 필요는 없음

▶ putchar( ) 함수

✓ **문자를 출력하는 함수**

- **호출이 실패**했을 때 **EOF 반환**
- 일일이 EOF 반환 여부를 검사할 필요는 없음

▶ EOF(End Of File)

✓ 파일의 끝을 의미

✓ stdio.h 에 -1로 매크로 상수 정의

✓ [ctr+z]를 입력하면 EOF로 인식

3.2 콘솔 입출력 (3/4)---[3-1.c 실습]

```
#include<stdio.h>
int main(void)
{
    char ch=0;
    while( ch != EOF) // EOF == -1
    {
        ch=getchar();
        putchar(ch);
    }

    return 0;
}
```

3.2 콘솔 입출력

▶배울 내용

① 콘솔 표준 입출력 함수

② 콘솔 비표준 입출력 함수

3.2 콘솔 입출력 (4/4)

▶ 콘솔 비표준 입출력 함수

✓ 헤더파일 : **conio.h**

✓ **'버퍼를 사용하지 않는다.'**

✓ **데이터를 일괄해서 처리**하는 경우에는 **비효율적**

함수의 원형	기능	헤더파일
int **getch** (void);	키보드로 부터 한 문자를 입력 받는다. (입력 화면에 입력 문자가 보이지 않는다.)	conio.h
int **getche** (int c);	키보드로 부터 한 문자를 입력 받는다. (입력 화면에 입력 문자가 보인다.)	conio.h
int **putch** (int c);	모니터에 한 문자를 출력한다.	conio.h
int **kbhit** (void);	키보드상에 있는 키가 눌려졌는지를 조사한다. (누른 경우 0이 아닌 수를 반환)	conio.h

3.3 파일 입출력

-교재 584페이지 -

3.3 파일 입출력

▶배울 내용

① 파일 입출력의 필요성

② 파일을 이용한 입출력 과정

③ fopen( ) 함수와 fclose( ) 함수

3.3 파일 입출력 (1/11)

▶ 파일 입출력의 필요성

✓ '실행 중에 데이터가 생성되면 데이터는 **메모리에 보관 된다.**'

- **문제점 :** 프로그램이 종료되면 사라짐

✓ 데이터를 프로그램이 종료된 후에도 계속해서 사용하려면?

- **파일에 저장**
  - '중요한 **데이터를 파일에 저장한다.**'
  - '필요할 때 **파일을 읽어 데이터를 사용한다.**'

3.3 파일 입출력 (2/11)

▶ 파일의 유형

✓ 텍스트 파일(Text File)

- 데이터 내용을 확인할 수 있는 문자들을 저장해놓은 파일
- 문자열과 같은 **텍스트 기반의 데이터 파일**

예) *.txt, *c, *.hwp, *.doc

✓ 바이너리 파일(Binary File)

- 일반 문서 편집기에서 **내용을 확인 할 수 없음**
- **이진 형식으로 인코딩된 데이터 파일**
- 바이너리 파일을 텍스트 모드로 열면 글자가 깨지는 현상이 발생

예) *.obj, .exe, *.lib, *.dll

3.3 파일 입출력

▶배울 내용

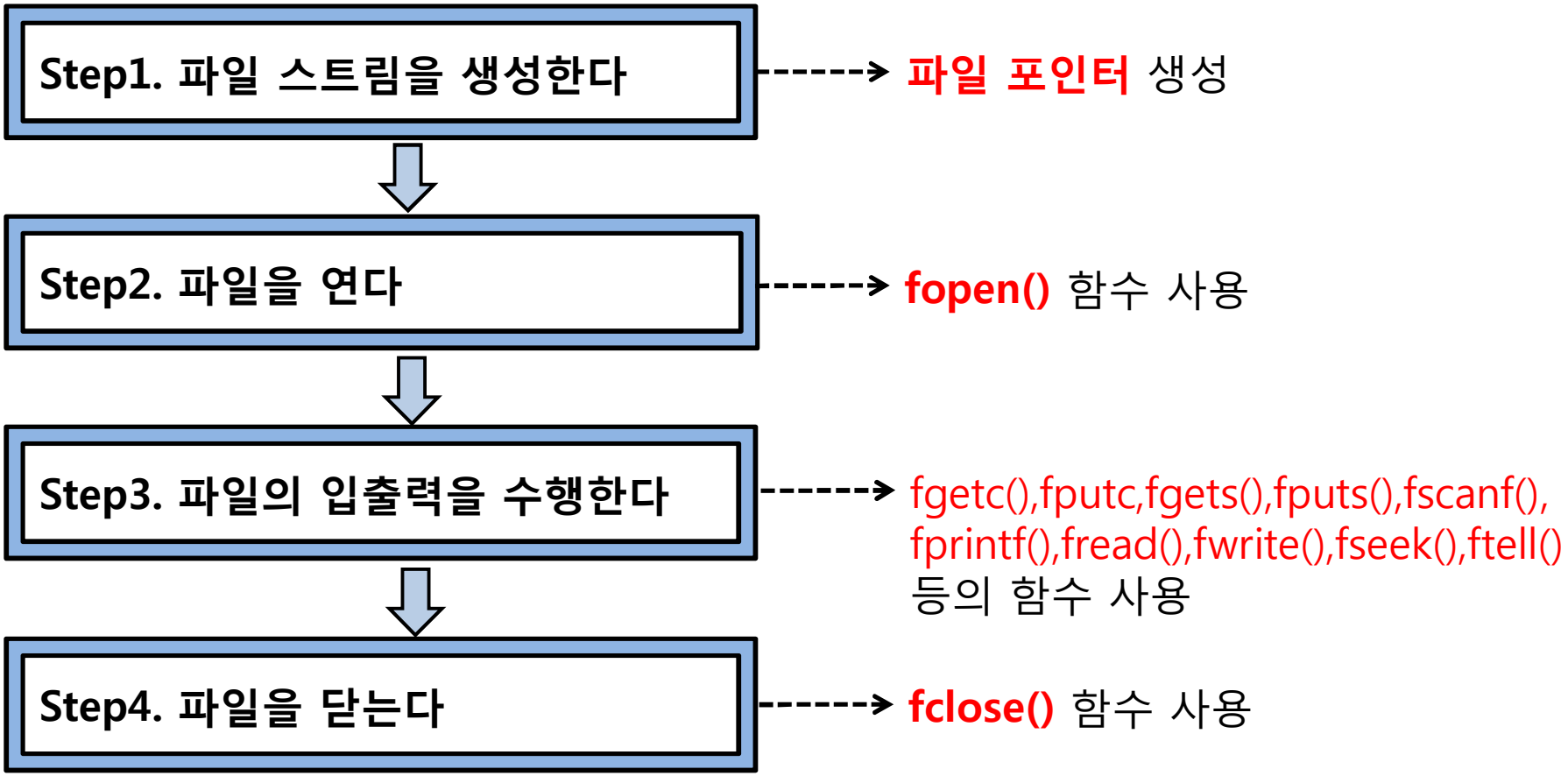
① 파일 입출력의 필요성

② 파일을 이용한 입출력 과정

③ fopen( ) 함수와 fclose( ) 함수

3.3 파일 입출력 (3/11)

▶ 파일 입출력 과정



3.3 파일 입출력

▶배울 내용

① 파일 입출력의 필요성

② 파일을 이용한 입출력 과정

③ fopen( ) 함수와 fclose( ) 함수

3.3 파일 입출력 (4/11)

▶ fopen()함수와 fclose()함수

✓ 헤더파일: **stdio.h**

✓ **fopen()함수**: '파일 스트림을 생성하고 파일을 오픈 한다.'

함수의 원형	설명
#include<stdio.h> **FILE* fopen (const char* filename, const char* mode);**	파일 스트림을 생성하고 파일을 연다. **호출 실패의 경우: NULL 반환**

✓ **fclose()함수**: '파일 스트림을 닫고, 파일도 닫는다.'

함수의 원형	기능
#include<stdio.h> **int fclose (FILE* stream);**	파일을 닫는다. **호출 실패의 경우: EOF 반환**

3.3 파일 입출력 (5/11)

▶ 파일 스트림

✓ 'FILE* 구조체 포인터'를 이용

- 예) FILE* stream;

▶ fopen( )의 인자

✓ 첫 번째 인자: filename

- 파일의 경로와 이름을 동시에 표현

✓ 두 번째 인자: mode

- 파일의 접근 모드 와 파일 입출력 모드 표현

3.3 파일 입출력 (6/11)

▶ 파일의 접근 모드 (r, w, a, r+, w+, a+)

모드	설명
r	**읽기 전용**으로 파일을 연다. **파일이 없거나 찾을 수 없는 경우**에 호출 **실패**
w	**쓰기 전용**으로 파일을 연다. - 지정한 파일명이 있는 경우: **파일 내용을 모두 지우고 새로 만든다.** - 지정한 파일명이 없는 경우: **새로운 파일을 생성 한다.**
a	**추가 쓰기 전용**으로 파일을 열다. -지정한 파일이 있으면 **파일의 끝에서부터 내용을 추가**합니다.
r+	**파일을 읽고 쓰기** 위해 연다. - 지정한 파일이 있는 경우: **기존의 내용을 덮어쓴다.** - 지정한 파일이 없는 경우: **새로운 파일을 생성**해서 데이터를 쓴다.
w+	**파일을 읽고 쓰기** 위해 연다. -지정한 파일이 있는 경우: **파일의 내용을 모두 지우고** 새 파일을 만든다.- - 지정한 파일이 없는 경우: **새로운 파일을 생성 한다.**
a+	**파일을 읽고 추가 쓰기** 위해 연다. -지정한 파일이 있으면 파일의 끝에서부터 내용을 추가한다. 나 -나머지 기능은 r+와 같다.

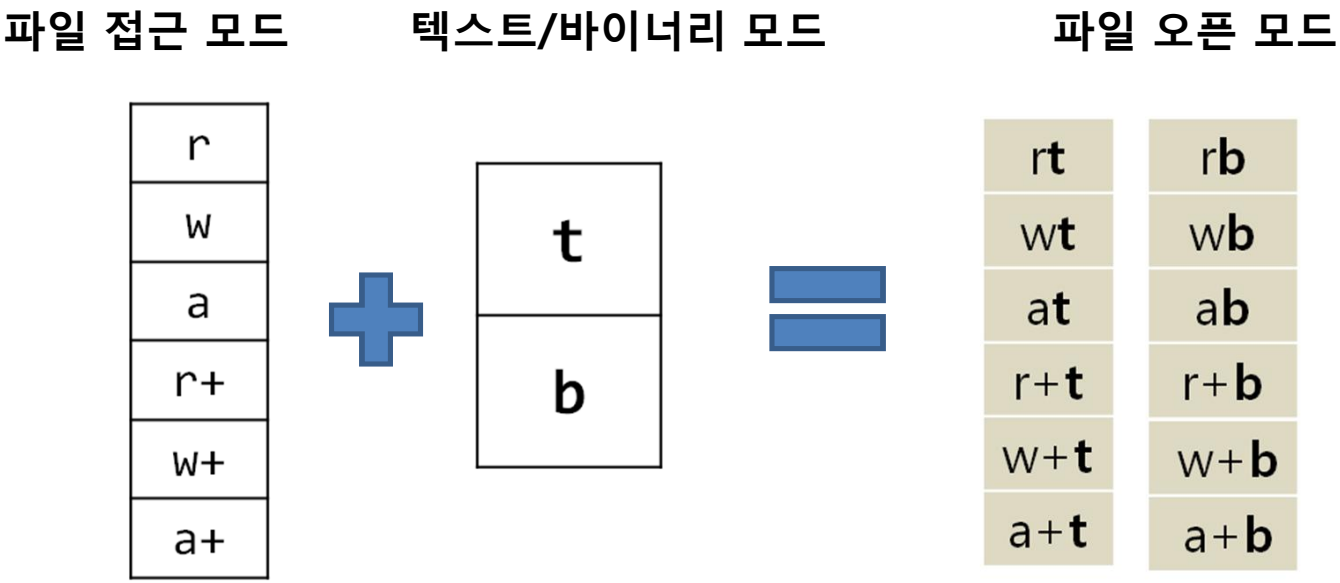
3.3 파일 입출력 (7/11)

▶ 파일 입출력 모드

- ✓ text 모드
- ✓ binary 모드

모드	설명
t	텍스트 파일 모드
b	바이너리 파일 모드

3.3 파일 입출력 (8/11)



r : 읽기 (read)
w: 쓰기 (write)
a: 추가 (append)

t : 텍스트 파일 모드(text file mode)
b: 바이너리 파일 모드 (binary file mode)

3.3 파일 입출력 (9/11)

```
/* case 1 */
FILE* stream;
stream=fopen("d:₩₩project₩₩data.txt", "rt");

/* case 2 */
FILE* stream
stream=fopen("data.txt", "rt");

/* case 3 */
FILE* stream;
stream=fopen("d:₩₩project₩₩data.txt", "r");

/* case 4 */
FILE* stream;
stream=fopen("d:₩₩project₩₩data.txt", "w");

/* case 5 */
FILE* stream;
stream=fopen("d:₩₩project₩₩data.txt", "ab");
```

텍스트 모드의 경우
't' 생략 가능

3.3 파일 입출력 (10/11)---[3-3.c 실습]

```
#include<stdio.h>
int main(void)
{
    FILE* stream;   // 파일 스트림 생성을 위한 FILE 포인터 선언
    int file_state;   // 파일의 종료를 위한 상태 체크 변수 선언

    // 파일 스트림 생성과 파일 열기
    stream = fopen("data1.txt","w");
    if (stream == NULL)
        printf("파일 열기 에러₩n");

    // 파일 닫기(파일 스트림 소멸)
    file_state = fclose(stream);
    if (file_state == EOF)
        puts("파일 닫기 에러");
    return 0;
}
```

3.3 파일 입출력 (11/11)---[3-3.c 분석]

3-3.c를 간략화한 코드

```
#include<stdio.h>
int main(void)
{
    FILE* stream;
    stream = fopen("data1.txt","w");
    fclose(stream);
    return 0;
}
```

3.4 표준 파일 입출력 함수

-교재 592페이지 -

3.4 표준 파일 입출력 함수

▶배울 내용

① 대표적인 표준 파일 입출력 함수

② fgetc( ) 함수와 fputc( ) 함수

③ fgets( ) 함수와 fputs( ) 함수

④ fprintf( ) 함수와 fscanf( ) 함수

⑤ feof( ) 함수

⑥ fflush( ) 함수

⑦ fread( ) 함수와 fwrite( ) 함수

⑧ fseek( ) 함수와 ftell( ) 함수

3.4 표준 파일 입출력 함수 (1/27)

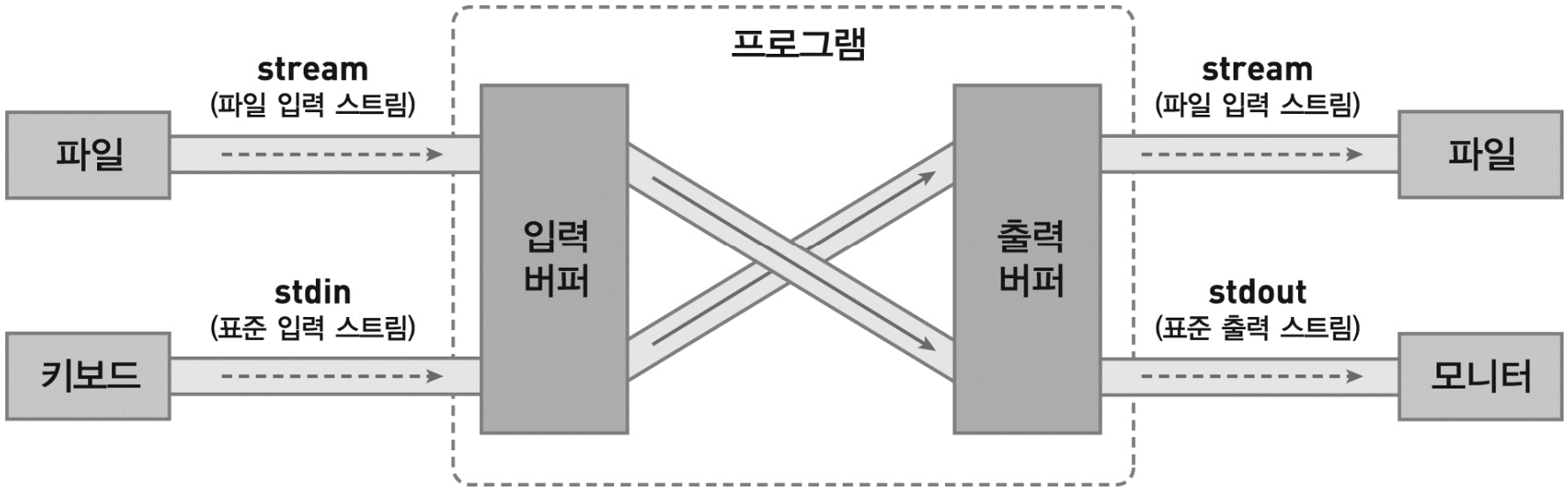
▶ 대표적인 표준 입출력 함수와 표준 파일 입출력 함수

표준 입출력 함수	표준파일 입출력 함수	기능
int **getchar** (void);	int **fgetc** (FILE* **stream**); stdin	문자 단위 입력
int **putchar** (int c);	int **fputc** (int c, FILE* **stream**); stdout	문자 단위 출력
char* **gets** (char *s);	char* **fgets** (char *s, int n, FILE* **stream**); stdin	문자열 단위 입력
int **puts** (char* str);	int **fputs** (const char* s, FILE* **stream**); stdout	문자열 단위 출력
int **scanf** (const char* format, ...);	int **fscanf** (FILE* **stream**, const char* format, ...); stdin	자료형에 맞춘 입력
int **printf** (const char* format, ...);	int **fprintf** (FILE* **stream**, const char* format, ...); stdout	자료형에 맞춘 출력

✓ **stream**을 입력하는 함수인 경우- stdin 또는 stdout 선택적 사용 가능

3.4 표준 파일 입출력 함수 (2/27)

▶ 표준 파일 입출력 함수의 선택 적용



- ✓ **Case 1 -** 파일(stream) 입력 ➔ 파일(stream) 출력
- ✓ **Case 2 -** 파일(stream) 입력 ➔ 모니터(stdout) 출력
- ✓ **Case 3 -** 키보드(stdin) 입력 ➔ 파일(stream) 출력
- ✓ **Case 4 -** 키보드(stdin) 입력 ➔ 모니터(stdout) 출력

3.4 표준 파일 입출력 함수

▶배울 내용

① 대표적인 표준 파일 입출력 함수

② fgetc( ) 함수와 fputc( ) 함수

③ fgets( ) 함수와 fputs( ) 함수

④ fprintf( ) 함수와 fscanf( ) 함수

⑤ feof( ) 함수

⑥ fflush( ) 함수

⑦ fread( ) 함수와 fwrite( ) 함수

⑧ fseek( ) 함수와 ftell( ) 함수

3.4 표준 파일 입출력 함수 (3/27)

▶ 헤더파일 : **stdio.h**

함수의 원형	설명
int **fgetc** (FILE* **stream**);	키보드/파일로 부터 한 문자를 입력 받는다. **파일의 끝에 도달 할 경우:** EOF 반환
int **fputc** (int c, FILE* **stream**);	모니터/파일에 한 문자를 출력한다. **호출 실패의 경우:** EOF 반환

3.4 표준 파일 입출력 함수 (4/27)---[3-4.c 실습]

```
#include <stdio.h>
int main(void)
{
    FILE* stream;
    int file_state;
    int input=0;

    stream=fopen("data1.txt", "w");
    if(stream==NULL)
        puts("파일 열기 에러");

    puts("데이터입력");
    while(input != EOF)
    {
        input=fgetc(stdin);
        fputc(input, stream);
    }

    file_state=fclose(stream);
    if(file_state==EOF)
        puts("파일 닫기 에러");
    return 0;
}
```

3.4 표준 파일 입출력 함수 (5/27)---[3-5.c 실습]

```
#include<stdio.h>
int main(void)
{
    FILE* stream1;    // 읽기 전용 파일 스트림 선언
    FILE* stream2;    // 쓰기 전용 파일 스트림 선언
    int input=0;

    stream1 = fopen("data1.txt","r");
    stream2 = fopen("data2.txt","w");

    puts("파일로부터 데이터를 입력");
    while( input != EOF )
    {
        input = fgetc(stream1);
        fputc(input, stream2);
        fputc(input, stdout);
    }
    fclose(stream1);
    fclose(stream2);
    return 0;
}
```

3.4 표준 파일 입출력 함수

▶배울 내용

① 대표적인 표준 파일 입출력 함수

② fgetc( ) 함수와 fputc( ) 함수

③ fgets( ) 함수와 fputs( ) 함수

④ fprintf( ) 함수와 fscanf( ) 함수

⑤ feof( ) 함수

⑥ fflush( ) 함수

⑦ fread( ) 함수와 fwrite( ) 함수

⑧ fseek( ) 함수와 ftell( ) 함수

3.4 표준 파일 입출력 함수 (6/27)

▶ 헤더파일 : **stdio.h**

함수의 원형	설명
char* **fgets** (char* s, int n, FILE* **stream**);	키보드/파일로 부터 문자열을 입력 받는다. **파일의 끝에 도달 할 경우:** NULL 포인터 반환
int **fputs** (const char* s, FILE* **stream**);	모니터/파일에 문자열을 출력한다. **호출 실패의 경우:** EOF 반환

3.4 표준 파일 입출력 함수 (7/27)---[3-6.c 실습]

```
#include<stdio.h>
#define MAX 100

int main(void)
{
    FILE* stream;
    char buffer[50];

    stream = fopen("data3.txt","w");
    if(stream == NULL)
        puts("파일 열기 오류");

    fgets(buffer, sizeof(buffer), stdin);
    fputs(buffer, stream);

    fclose(stream);

    return 0;
}
```

3.4 표준 파일 입출력 함수

▶배울 내용

① 대표적인 표준 파일 입출력 함수

② fgetc( ) 함수와 fputc( ) 함수

③ fgets( ) 함수와 fputs( ) 함수

④ fprintf( ) 함수와 fscanf( ) 함수

⑤ feof( ) 함수

⑥ fflush( ) 함수

⑦ fread( ) 함수와 fwrite( ) 함수

⑧ fseek( ) 함수와 ftell( ) 함수

3.4 표준 파일 입출력 함수 (8/27)

▶ 자료형 단위의 표준 입출력 함수

✓ fscanf(), fprintf()

✓ 헤더파일 : **stdio.h**

함수의 원형	설명
int **fscanf** (FILE* **stream**, const char* format, ...);	키보드/파일로 부터 자료형에 맞춰 데이터를 입력한다. (텍스트 데이터와 바이너리 데이터를 동시 입력) **파일의 끝에 도달한 경우:** EOF 반환
int **fprintf** (FILE* **stream**, const char* format, ...);	모니터/파일에 자료형에 맞춰 데이터를 출력한다. (텍스트 데이터와 바이너리 데이터를 동시 출력)

3.4 표준 파일 입출력 함수 (9/27)---[3-7.c 실습]

```
#include<stdio.h>
int main(void)
{
    FILE* stream;
    char name[20];
    int  kor, eng, total;

    printf("1. 이름입력: ");
    fscanf(stdin,"%s",name);                    // 키보드로부터 데이터를 입력

    printf("2. 국어점수, 영어점수입력: ");
    fscanf(stdin,"%d %d",&kor, &eng);           // 키보드로부터 데이터를 입력
    total = kor + eng;

    stream = fopen("data4.txt","w");
    fprintf(stream,"%s %d %d %d ₩n", name, kor, eng, total);  //data4.txt에 출력
    //fprintf(stdout,"%s %d %d %d ₩n", name, kor, eng, total);     // 모니터에 출력
    fclose(stream);
    return 0;
}
```

3.4 표준 파일 입출력 함수 (10/27)---[3-8.c 실습(1/2)]

```
#include<stdio.h>
int main(void)
{
    FILE* stream1;
    FILE* stream2;

    char name[10]="";
    int  kor=0, eng=0, total=0;

    stream1 = fopen("data4.txt","r");
    stream2 = fopen("data5.txt","w");

    fscanf(stream1,"%s %d %d %d ₩n", name, &kor, &eng, &total);
    fprintf(stream2,"%s %d %d %d ₩n", name, kor, eng, total);
 // fprintf(stdout, "%s %d %d %d ₩n", name, kor, eng, total);

    fclose(stream1);
    fclose(stream2);
    return 0;
}
```

3.4 표준 파일 입출력 함수

▶배울 내용

① 대표적인 표준 파일 입출력 함수

② fgetc( ) 함수와 fputc( ) 함수

③ fgets( ) 함수와 fputs( ) 함수

④ fprintf( ) 함수와 fscanf( ) 함수

⑤ feof( ) 함수

⑥ fflush( ) 함수

⑦ fread( ) 함수와 fwrite( ) 함수

⑧ fseek( ) 함수와 ftell( ) 함수

3.4 표준 파일 입출력 함수 (11/27)

▶ feof( ) 함수를 사용하는 이유

✓ 파일의 끝을 확인하는 다양한 표현

함수	파일의 끝에서 반환하는 값
fgetc( )	EOF(-1)
fgets( )	NULL(0)
fscanf( )	EOF(-1)

✓ 단점

- '파일 끝에서 반환되는 값을 일일이 기억하는 것이 불편하다.'

✓ 단점 해결

- feof()함수를 사용하자!!!

3.4 표준 파일 입출력 함수 (12/27)

▶ 파일의 끝을 검사하는 feof()함수

✓ 헤더파일 : **stdio.h**

함수 원형	설명
int feof(FILE* stream);	파일의 끝에 도달했는지 아닌지를 검사 **파일의 끝에 도달 :** 0이 아닌 값 반환 **파일의 끝에 도달하지 못한 경우 :** 0 반환

3.4 표준 파일 입출력 함수 (13/27)---[3-9.c 실습]

```
#include<stdio.h>
int main(void)
{
    FILE* stream1;
    FILE* stream2;
    char buffer[50];

    stream1 = fopen("data1.txt","r");
    stream2 = fopen("data2.txt","w");

    while( !feof(stream1) )
    {
        fgets(buffer,sizeof(buffer),stream1);
        fputs(buffer,stream2);
    }

    fclose(stream1);
    fclose(stream2);

    return 0;
}
```

3.4 표준 파일 입출력 함수

▶배울 내용

① 대표적인 표준 파일 입출력 함수

② fgetc( ) 함수와 fputc( ) 함수

③ fgets( ) 함수와 fputs( ) 함수

④ fprintf( ) 함수와 fscanf( ) 함수

⑤ feof( ) 함수

⑥ fflush( ) 함수

⑦ fread( ) 함수와 fwrite( ) 함수

⑧ fseek( ) 함수와 ftell( ) 함수

3.4 표준 파일 입출력 함수 (14/27)

▶ 버퍼를 비우는 함수

- ✓ **fflush()함수**
- ✓ 헤더파일 : **stdio.h**

함수 원형	설명
int fflush(FILE* stream);	버퍼를 비움 **실패 :** EOF 반환

3.4 표준 파일 입출력 함수 (15/27)---[3-10.c 실습]

```
#include<stdio.h>
int main(void)
{
    int age;
    char name[20];

    printf("나이입력: ");
    scanf("%d",&age);

    //fflush(stdin); // 입력 버퍼를비운다.

    printf("이름을 입력: ");
    fgets(name, sizeof(name), stdin);

    printf("%d₩n",age);
    printf("%s₩n",name);

    return 0;
}
```

3.4 표준 파일 입출력 함수 (16/27)---[3-10.c 분석]

▶ 문제가 생기는 이유

▶ 문제점 해결

✓ fflush(stdin)의 주석을 제거하면 정상적으로 출력

3.4 표준 파일 입출력 함수

▶배울 내용

① 대표적인 표준 파일 입출력 함수

② fgetc( ) 함수와 fputc( ) 함수

③ fgets( ) 함수와 fputs( ) 함수

④ fprintf( ) 함수와 fscanf( ) 함수

⑤ feof( ) 함수

⑥ fflush( ) 함수

⑦ fread( ) 함수와 fwrite( ) 함수

⑧ fseek( ) 함수와 ftell( ) 함수

3.4 표준 파일 입출력 함수 (17/27)

▶ 바이너리 파일 입출력을 지원하는 함수

✓ fwrite()함수

✓ 헤더파일 : **stdio.h**

함수의 원형	기능
size_t **fwrite**(const void* buffer①, size_t size②, size_t count③, FILE* stream④);	buffer에 저장된 데이터를 파일에 출력한다. **호출 성공 시:** count(반복 횟수) 반환 **호출 실패 시:** count 보다 작은 값 반환

① buffer: 출력 데이터를 저장한 버퍼를 가리키는 포인터

② size: 한번에 출력할 데이터의 바이트 크기

③ count: 반복 횟수

④ stream: 파일 출력 스트림

3.4 표준 파일 입출력 함수 (18/27)

▶ 바이너리 파일 입출력을 지원하는 함수

✓ **fread()함수**

✓ 헤더파일 : **stdio.h**

함수의 원형	기능
size_t **fread** (void* buffer, size_t size, size_t count, FILE* stream); ① ② ③ ④	파일로 부터 바이너리 데이터를 받아 buffer로 입력 한다. **호출 성공 시:** count(반복 횟수) 반환 **호출 실패 시:** count 보다 작은 값 반환

① buffer: 파일로 부터 입력 받은 데이터를 저장하는 버퍼를 가리키는 포인터

② size: 한번에 입력 받을 데이터의 바이트 크기

③ count: 입력 횟수

④ stream: 파일 입력 스트림

✓ ① buffer가 **void*(void형 포인터)**인 이유

- **어떤 유형의 buffer**를 사용할지 **자유롭게 선택**

3.4 표준 파일 입출력 함수 (19/27)---[3-11.c 실습]

```
#include<stdio.h>
int main(void)
{
    int buffer1[5] = {0xff, 0x56, 0x78, 0xfa, 0xf1};
    int buffer2[5];

    FILE* stream;
    stream = fopen("student.dat","wb");     // 바이너리 모드, 쓰기 모드
    fwrite(buffer1,sizeof(int),5,stream);
    fclose(stream);

    stream = fopen("student.dat","rb");     // 바이너리 모드, 읽기 모드
    fread(buffer2,sizeof(int),5,stream);
    printf("%x %x %x %x %x ₩n", buffer2[0], buffer2[1], buffer2[2], buffer2[3], buffer2[4]);
    fclose(stream);
    return 0;
}
```

3.4 표준 파일 입출력 함수

▶배울 내용

① 대표적인 표준 파일 입출력 함수

② fgetc( ) 함수와 fputc( ) 함수

③ fgets( ) 함수와 fputs( ) 함수

④ fprintf( ) 함수와 fscanf( ) 함수

⑤ feof( ) 함수

⑥ fflush( ) 함수

⑦ fread( ) 함수와 fwrite( ) 함수

⑧ fseek( ) 함수와 ftell( ) 함수

3.4 표준 파일 입출력 함수 (20/27)

▶ 랜덤 접근 함수 fseek()함수

✓ 헤더파일 : **stdio.h**

함수 원형	설명
int fseek(FILE* stream, long offset, int **start**);	start부터 offset까지 스트림을 이동시킨다. **성공** : 0 반환 **실패** : 0이 아닌 값 반환

✓ 세 번째 인자 **start**

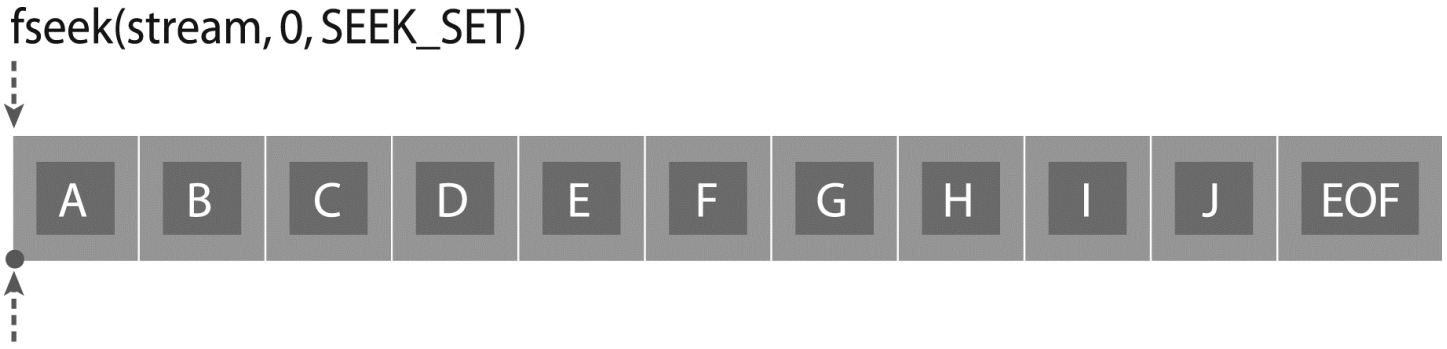
기호 상수	값	설명
SEEK_SET	0	파일의 시작 위치
SEEK_CUR	1	파일의 현재 위치
SEEK_END	2	파일의 끝 위치

3.4 표준 파일 입출력 함수 (21/27)---[3-13.c 실습]

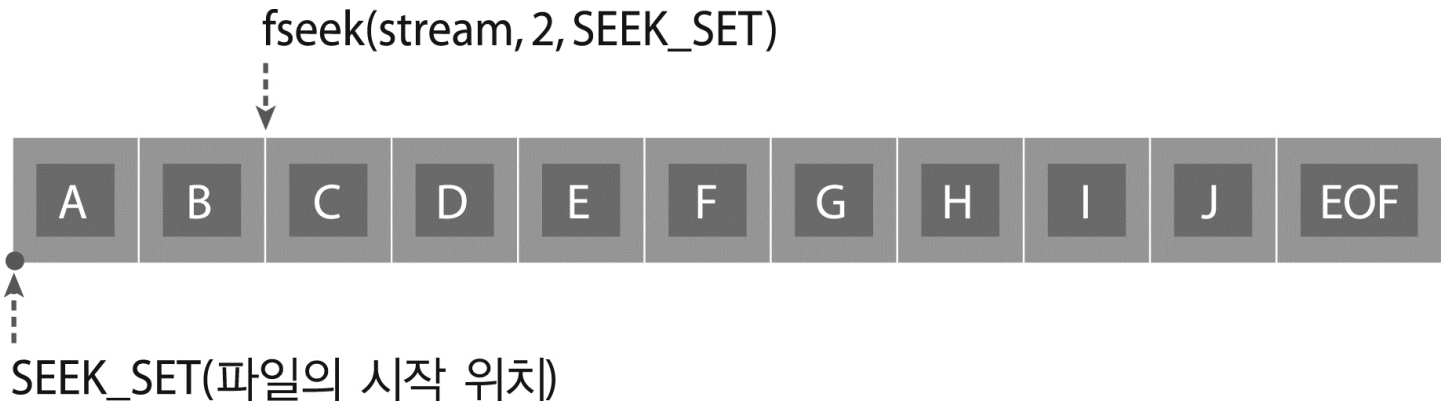
```
#include <stdio.h>
int main(void)
{
        FILE* stream;
        stream=fopen("seek.txt", "w");          // 쓰기 모드
        fputs("ABCDEFGHIJ", stream);
        fclose(stream);

        stream=fopen("seek.txt", "r");          // 읽기 모드

        fseek(stream, 0, SEEK_SET);
        fprintf(stdout, "%c ₩n", fgetc(stream));    // A 출력
        fseek(stream, 2, SEEK_SET);
        fprintf(stdout, "%c ₩n", fgetc(stream));    // C 출력
        fseek(stream, -1, SEEK_END);
        fprintf(stdout, "%c ₩n", fgetc(stream));    // J 출력
        fseek(stream, -2, SEEK_CUR);
        fprintf(stdout, "%c ₩n", fgetc(stream));    // I 출력

        fclose(stream);
        return 0;
}
```

3.4 표준 파일 입출력 함수 (22/27)

▶ fseek(stream, 0, SEEK_SET);

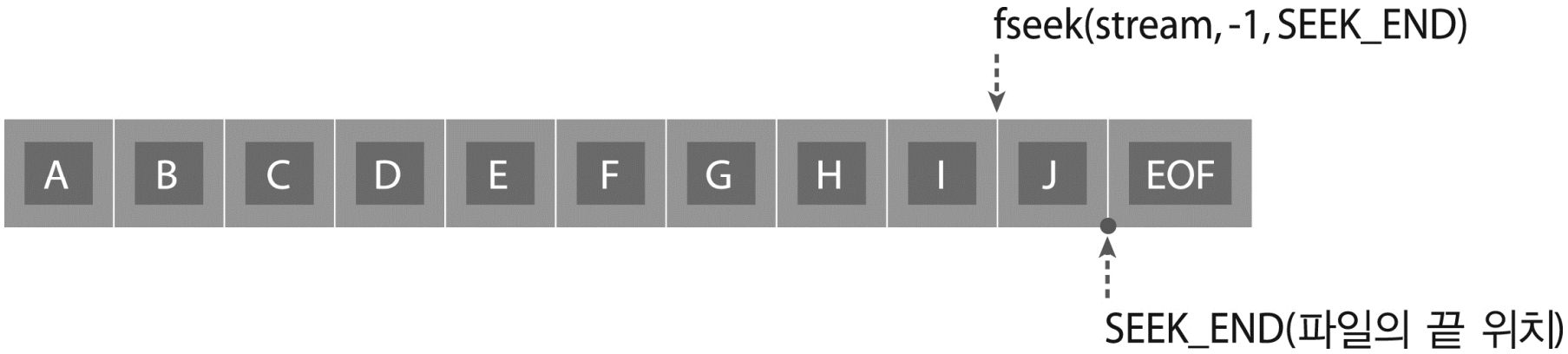


▶ fseek(stream, 2, SEEK_SET);

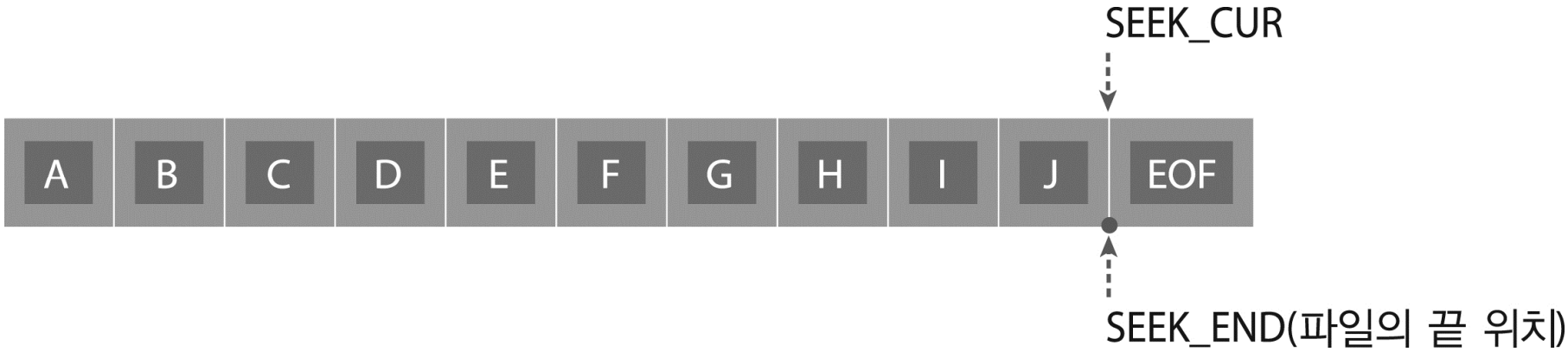


3.4 표준 파일 입출력 함수 (23/27)

▶ fseek(stream, -1, SEEK_END);

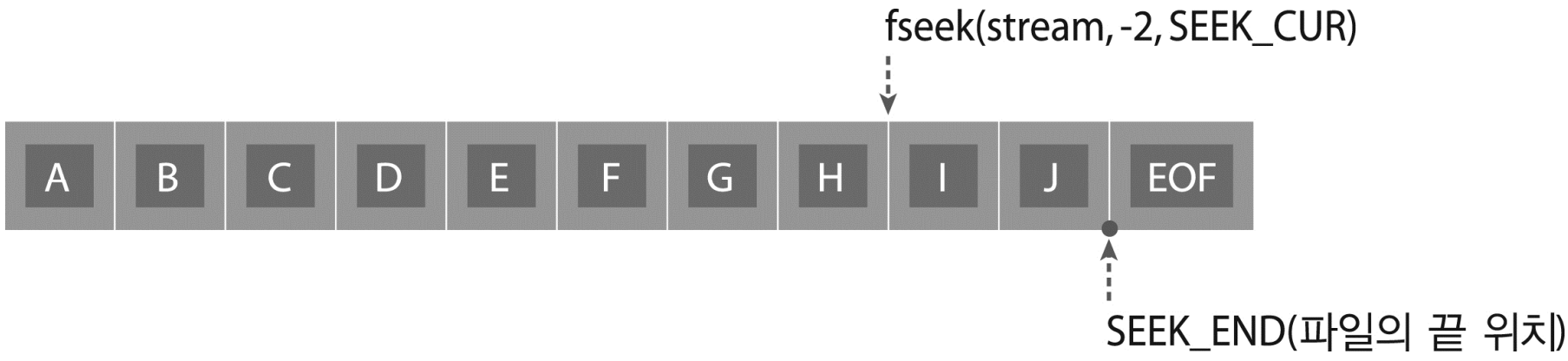


▶ SEEK_CUR의 위치



3.4 표준 파일 입출력 함수 (24/27)

▶ fseek(stream, -2, SEEK_CUR);



3.4 표준 파일 입출력 함수 (25/27)

▶ 랜덤 접근 함수 ftell()함수

- ✓ 현재의 파일 위치가 파일의 시작부터 얼마나 떨어져 있는지를 확인
- ✓ 헤더파일 : **stdio.h**

함수 원형	설명
long ftell(FILE* stream);	파일 포인터 stream의 위치를 확인 **성공 :** 파일 포인터의 위치 반환 **실패 :** EOF 반환

3.4 표준 파일 입출력 함수 (26/27)---[3-14.c 실습]

```
#include <stdio.h>
int main(void)
{
        FILE* stream;
        long distance;

        stream=fopen("ftell.txt", "w");          // 쓰기모드
        fputs("ABCDEFGHIJ", stream);
        fclose(stream);

        stream=fopen("ftell.txt", "r");          // 읽기모드

        fseek(stream, -8, SEEK_END);
        fprintf(stdout, "%c ₩n", fgetc(stream));    // C 출력

        distance=ftell(stream);
        printf("거리: %ld ₩n", distance);          // 거리3
        fclose(stream);

        return 0;
}
```

3.4 표준 파일 입출력 함수 (27/27)---[3-15.c 실습]

```
#include<stdio.h>
int main(void)
{
    FILE* stream = fopen("ftell.txt", "rb");
    fseek(stream, 0, SEEK_END);
    printf("ftell.txt 파일의 크기 : %d 바이트₩n", ftell(stream));
    fclose(stream);

    return 0;
}
```

공부한 내용 떠올리기

- ▶ 스트림, 버퍼, 버퍼링이 무엇인지
- ▶ 콘솔 표준 입출력 함수와 콘솔 비표준 입출력 함수
- ▶ 파일을 이용한 입출력 과정
- ▶ 표준 파일 입출력 함수

헛되지 않은 10년 (출처: 사랑과 지혜의 탈무드)