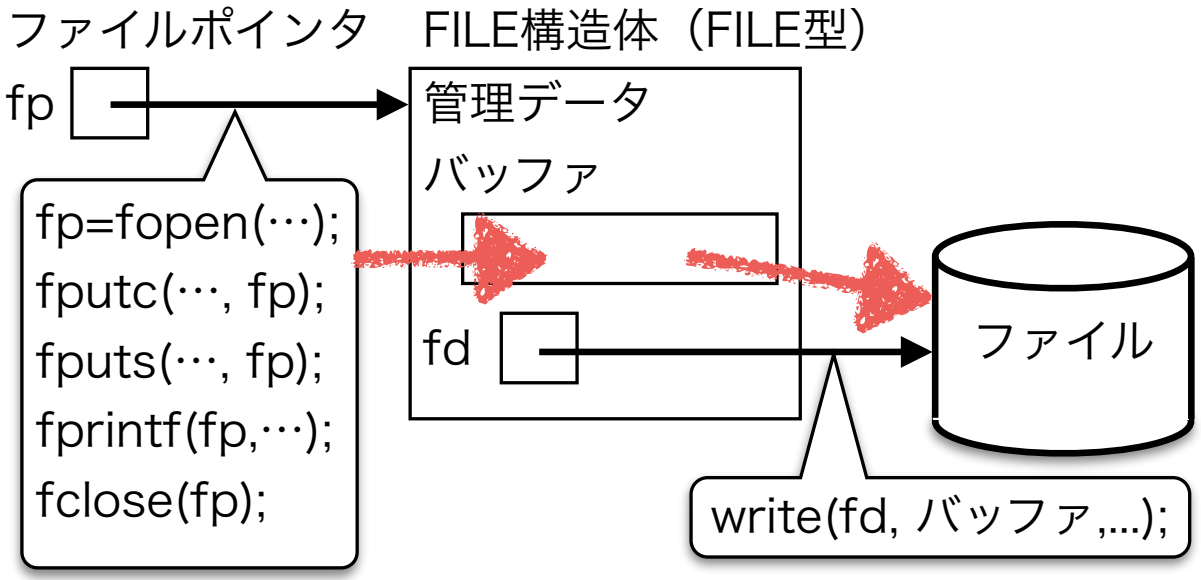
オペレーティングシステムの機能を使ってみよう
第３章 高水準入出力と低水準入出力

高水準入出力と低水準入出力

ファイルを読み書きするための機能
(API：Application Proguram Interface)

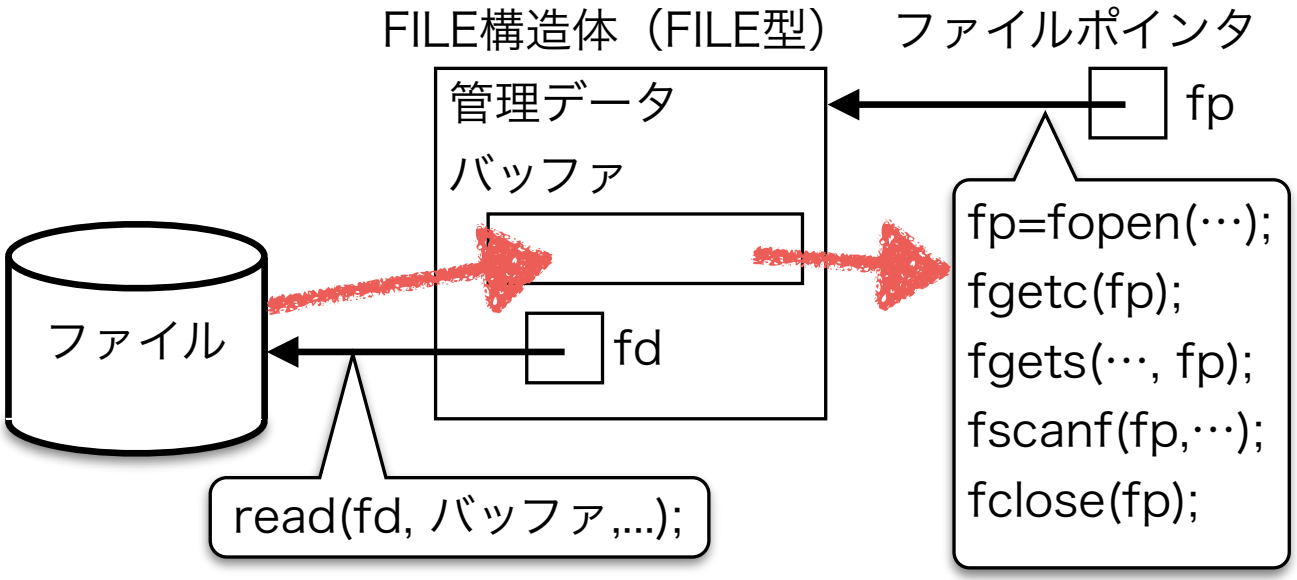
- 高水準入出力（高水準 *I/O*)
  多くの高機能な関数群
  (`fprintf()`, `fscanf()`, `fputc()`, `fgetc()`, ...)
- 低水準出力（高水準 *I/O*)
  システムコールのこと
  少なく，かつ，シンプルな API
  (`open()`, `read()`, `write()`, `lseek()`, `close()`)

高水準I/Oのデータ構造（書き込み）



- ファイルポインタ（fp）
- FILE構造体
- バッファリング
- writeシステムコール

高水準I/Oのデータ構造（読み出し）



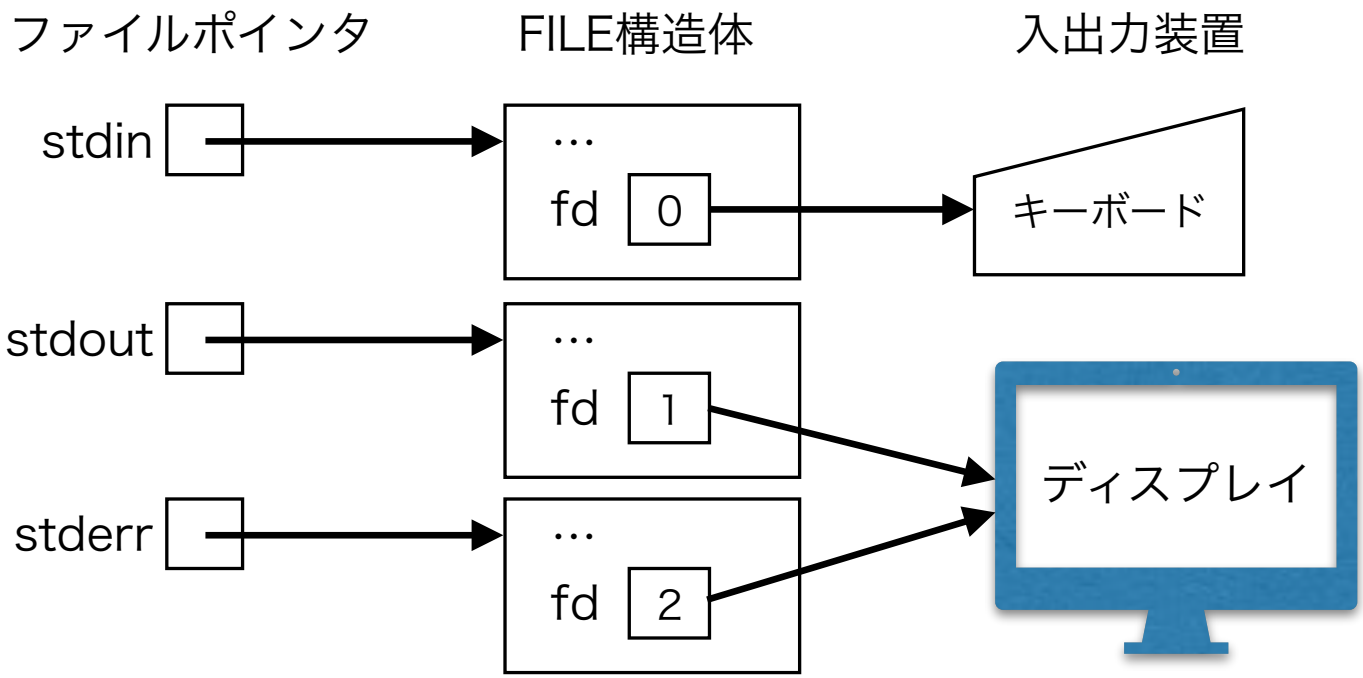
- ファイルポインタ（fp）
- FILE構造体
- readシステムコール
- バッファリング

標準入出力（標準入出力ストリーム）

名称	*fd*	*fp*	通常の接続先
標準入力ストリーム	0	stdin	キーボード
標準出力ストリーム	1	stdout	ディスプレイ
標準エラー出力ストリーム	2	stderr	ディスプレイ

fd：ファイルディスクリプタ

fp：ファイルポインタ

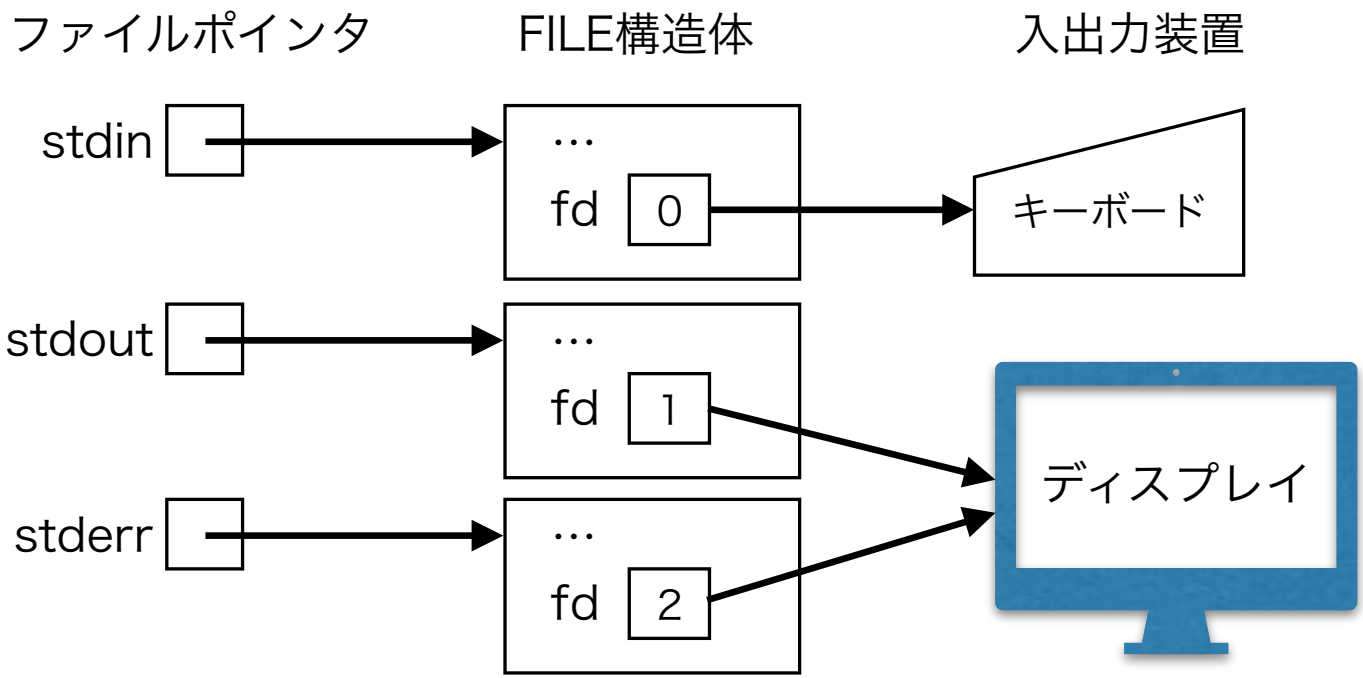


ユニファイド I/O

標準ストリーム	同じ意味の呼出し	役割り
`scanf(...)`	`fscanf(stdin, ...)`	書式付きの入力
`getchar()`	`fgetc(stdin)`	1 文字入力
-	`fgets(stdin, ...)`	1 行入力
`printf(...)`	`fprintf(stdout, ...)`	書式付きの出力
`putchar(c)`	`fputc(c, stdout)`	1 文字出力
`puts(buf)`	`fputs(buf, stdout)`	1 行出力

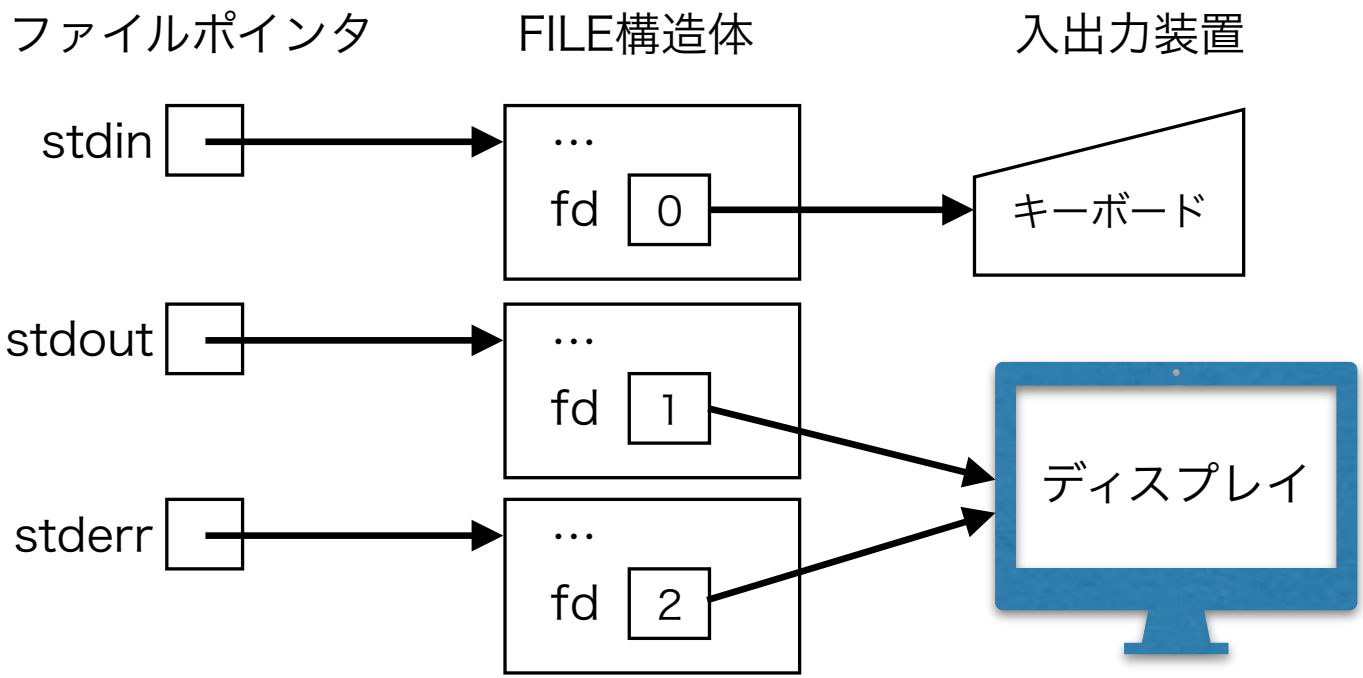
- `printf(...)` と `fprintf(stdout,...)` は同じ
- `fp` の代わりに `stdin`, `stdout` 等が使用できる.
- キーボードやディスプレイ（入出力装置）とファイルを同じ要領で操作できる.
- 入出力装置をファイルに統合＝（ユニファイド *I/O*）

標準入力ストリーム



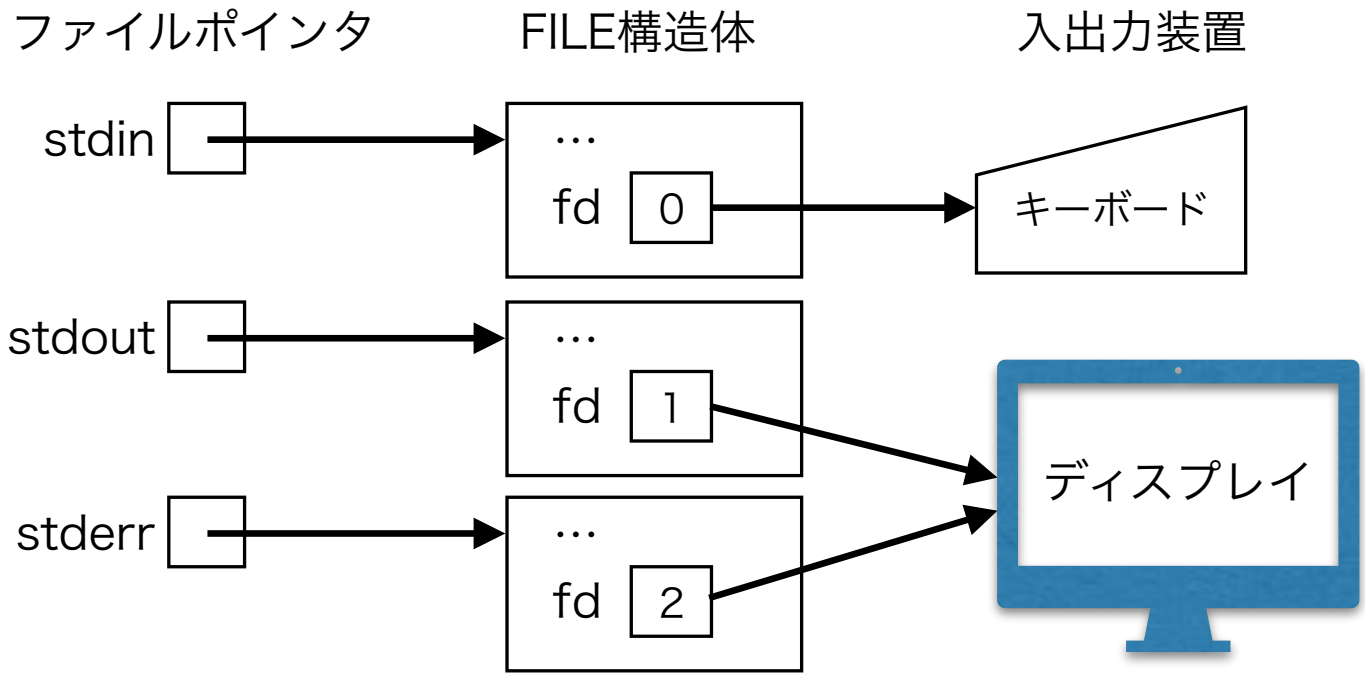
- ファイルポインタは `stdin`
- ファイルディスクリプタは 0 番
- ファイルディスクリプタ 0 は通常キーボードに接続
- ファイルポインタと FILE 構造体はプログラム起動時に初期化
- シェルはファイルディスクリプタ 0 をリダイレクト可能

標準出力ストリーム



- ファイルポインタは `stdout`
- ファイルディスクリプタは 1 番
- ファイルディスクリプタ 1 は通常ディスプレイに接続
- ファイルポインタと FILE 構造体はプログラム起動時に初期化
- シェルはファイルディスクリプタ 1 をリダイレクト可能

標準エラー出力ストリーム



- エーラメッセージ出力用のストリーム
- ファイルポインタは `stderr`
- ファイルディスクリプタは 2 番
- ファイルディスクリプタ 2 は通常ディスプレイに接続
- ファイルポインタと FILE 構造体はプログラム起動時に初期化
- シェルはファイルディスクリプタ 2 をリダイレクト可能

課題 No.2：三つのプログラムの性能比較

次の三つのプログラムの性能測定を行う．測定に適した実行時間になるようにファイルサイズは調整すること．演習方法の詳細は指示された Github の README を参照すること．

1. mycp（高水準 I/O を使用）
2. mycp2_1（バッファサイズ 1 バイトで低水準 I/O を使用）
3. mycp2_1024（バッファサイズ 1 Ki バイトで低水準 I/O を使用）