⇒ What is FILE (not file)?

⇒ Why Filepointer is not incremented?

⇒ If filepointer is incremented, then what happens?

```
ch = fgetc(fp);
fp++;
ch = fgetc(fp);
```

consider fileData is "VECTOR"

⇒ if using void *p; or char *q; we don't need stdio.h. But if we declare a file pointer then stdio.h inclusion is MUST.

why?

→ FILE is NOT a keyword, it is a typedef-structure,

---

typedef

→ A keyword, used to create new datatype, from existing type

Syntax: typedef Existingtype newtype;

e.g. typedef int integer;

```
main() { int x;
         integer y;
```

struct student {
    ≡
};

① typedef struct student STUDENT;

→ Portability

→ Readability

```
main() {
    STUDENT *buf;
    = malloc(sizeof(STUDENT));
```

char str[20]; // str is Array of 20 chars

typedef char string[20]; // string is a NEW Datatype

```
main() { string s1, s2; // s1 is 20bytes, s2 is Array of 20bytes
    strcpy(str, "vector");
```

//strcpy(string, str); // Invalid

(string: datatype)

string s[5]; // s is Array of 5 Arrays // total 100bytes

typedef struct student

{

≡

}STUDENT;

(2)

↳ New Datatype, not variable

---

typedef struct {

≡

}STUDENT;

↳ Datatype

(3)

opaque type

No Name of struct.

main() {

STUDENT *P;

can't use struct-keyword, as there's no tag

~~FILE~~

typedef struct {

≡ | different in Windows

" in Linux

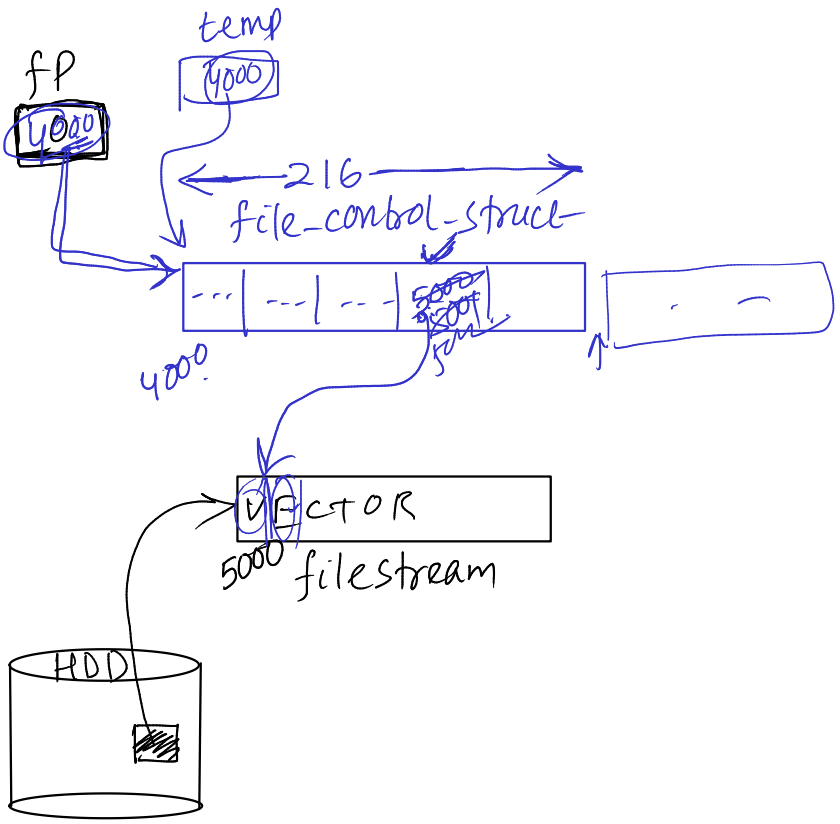
}FILE;

FILE is a struct type.

In many platforms the struct-having no tag

hence it is an opaque type

Restarting the fileHandling Basics.




```
FILE *fp = NULL;
fp = fopen("datafile.txt", "r");
ch = fgetc(fp);
ch = fgetc(fp);
fclose(fp);
```

Single step debug using gdb

1) gcc -g prog.c

2) gdb ./a.out

3) gdb> =

```
main() {
    FILE *fp;
    fp = fopen("--", "--");
    f1(fp);
```

```
f2(FILE *temp)
{

}
```