C - Week - 10 - Notes

L1 Introduc" to File Handling in C

memory available to system ~ GBs

External storage ~ TBs

- storage -
  1. on-chip registers in CPU → fastest, smallest ~ KBs
  2. Dynamic RAM → slower than regs., GBs
  3. external storage → slowest, TBs
- persistence - across power cycles
- structured storage → file sys.
  1. hierarchy - indexable / findable storage
  2. named file - store arbitary data
  3. organize into folders - possible
- file → name, path on sys., arbitary data, etc.
- programming interface - sys. calls
  → direct hardware access not allowed
  → go through OS kernel
- FILE → struct
- FILE* → fopen / fclose (pointers to file struct), permissions, disk space, paths, many possible errors

- special files → stdin, stdout, stderr

```
if ((fp = fopen("test.txt", "r")) == NULL)
  {
  }
```

L2 File Handling in Text Mode - Part I

- text mode - human readable files, ASCII / unicode encoding (display on print), no. as digits
- fgetc, fgets, fscanf
  fgetc(..., stdin) ≡ gets(...)
- fputc, fputs, fprintf
  fputs(..., stdout) ≡ puts(...)
- no equivalent for stderr

```
while ((ch = fgetc(fp)) != EOF)
  {
  }
while (fgets(s, 1024, fp) != NULL)
  {
  }
```

```
fscanf(fp, "%s", s)
fgets(s, 1024, fp)
```

L3 Part-2

```
fputc('H', fp);
fputs("ello", fp);
```

- stdin, out works similar like Linux systems.

L4 File Handling in Binary Mode

- binary mode - some OS handle text seperately, OS filters data before applicatn, but can cause problems.
- it gives direct byte level access to data.
- fwrite → direct map from file to array
  fread →
- potential problem - endianness

```
hd -n 100 main
```

L5 Bit Manipula^n

- bitwise operators

& (and), | (or), ^ (xor), ~ (NOT)

- masks

0 0 1 0 1 0 0 1 1 1 0 1 → LSB

↳ 9

- func^n - check, toggle, set, largest bits