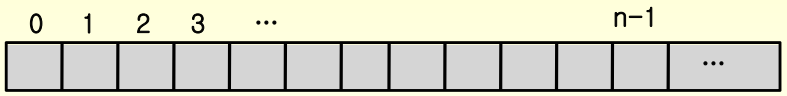
C Programming
Lecture 12 : File Processing

File

- File
  - Data are stored as a form of consecutive bytes



File Types

- Text File
  - Storing characters with ascii codes
- Binary File
  - Storing binary numbers as a consecutive bytes

Handling Files

1. Declare FILE type variable
   - Use standard library : `#include <stdio.h>`
   - ex) **`FILE* fp;`**
2. Open a file
   - Connect FILE type variable with actual file using `fopen` function
   - ex) **`fp=fopen("test.txt","r");`**
3. Perform I/O with the open file
   - Text file : **`fscanf(), fprintf(), fgets(), fputs(), …`**
   - Binary file : **`fread(), fwrite()`**
4. Close a file
   - Break the connection between FILE type variable and actual file
   - ex) **`fclose(fp);`**

File Open

- Before using a file, you must open the file
- **fopen()**
  - FILE *fopen(const char *filename, const char *mode);
- Example 1

```
FILE *fp;
fp = fopen("c:\work\text.txt", "r");
if (fp == NULL) {
    printf("file open error!\n");
}
```

- Example 2
  - `fp = fopen("outdata.txt", "w");`
  - `fp = fopen("outdata.txt", "a");`

File Open Modes

- r or rb
  - Open file for reading.
- w or wb
  - Truncate to zero length or create file for writing.
- a or ab
  - Append; open or create file for writing at end-of-file.
- r+ or rb+ or r+b
  - Open file for update (reading and writing).
- b stands for "binary"

Text File Processing

- `char* fgets(char *s, int n, FILE *fp);`
- `int fputs(const char *s, FILE *fp);`
- `int fprintf(FILE *fp, const char *format, ...);`
- `int fscanf(FILE *fp, const char *format, ...);`

deleteLine.c

```
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
    FILE *fp1;
    FILE *fp2;
    char buffer[100];

    fp1 = fopen(argv[1], "r");
    fp2 = fopen(argv[2], "w");
    if (fp1 == NULL) {
          printf("file not found");
          exit(1);
    }

    while(fgets(buffer, 100, fp1) != NULL)
          if (*buffer != '\n')
                fputs(buffer, fp2);
    fclose(fp1);
    fclose(fp2);
}
```

Binary File Processing

- `int fread(void *buf, int size, int n, FILE *fp);`
- `int fwrite(const void *buf, int size, int n, FILE *fp);`

student1.c

```
#include <stdio.h>
#include <stdlib.h>
#include "student.h"

main()
{
    struct student st, *stp = &st;
    FILE *fp = fopen("st_file", "wb");
    if(fp == NULL ) {
          printf("file open error\n");
          exit(1);
    }

    printf("student_id      Name  Year  Major\n“);
    while (scanf("%d %s %d %s", &st.stud_id, st.name, &st.year, st.major) == 4)
    {
          fwrite(stp, sizeof(struct student), 1, fp);
    }
    fclose(fp);
}
```

output:

```
Student_id  Name  year  Major
200601001 Mike    1 Computer
200601002 Tom     1 Computer
...
^Z
```

student2.c

```
#include <stdio.h>
#include <stdlib.h>
#include "student.h"

int main()
{
    struct student st, *stp = &st;
    FILE *fp = fopen("st_file", "rb");

    if (fp == NULL ) {
          printf("File Open Error.\n");
          exit(1);
    }

    printf("---------------------------------\n");
    printf(“%10s %6s %6s %10s\n", ”Student_ID“, "Name“, ”Year“, ”Major“);
    printf("---------------------------------\n");

    while (fread(stp, sizeof(struct student), 1, fp) > 0)
    {
          printf(“%10d %6s %6d %10s\n", st.stud_id, st.name, st.year, st.major);
    }
    printf("---------------------------------\n");
    fclose(fp);

    return 0;
}
```

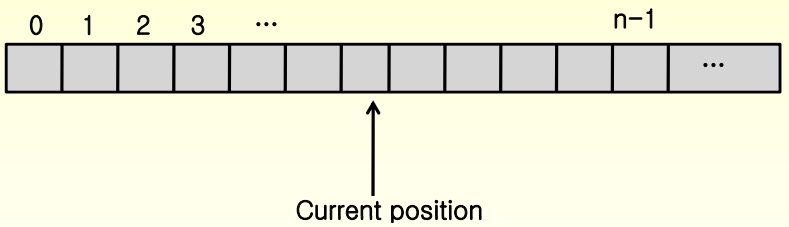
output:

```
---------------------------------
  Student_ID   Name   Year   Major
---------------------------------
    0601001   Mike         1  Computer
    0601002   Tom          1  Computer
            ...


---------------------------------
```

Random Access

- Current file position
  - Updated after file I/O is performed



Related Functions

- **int fseek(FILE *fp, long offset, int mode)**
- **void rewind(FILE *fp)**
- **int ftell(FILE *fp)**

fseek mode

mode	val	meaning
SEEK_SET	0	file start
SEEK_CUR	1	Current
SEEK_END	2	file end

- Example
  - fseek(fp, 0L, SEEK_SET) 파일처음으로 이동
  - fseek(fp, 100L, SEEK_CUR) 현재 위치에서 100 바이트 우로 이동
  - fseek(fp, 0L, SEEK_END) 파일 끝으로 이동

```
#include <stdio.h>
#include <stdlib.h>
#define  MAX  256

int main(int argc, char *argv[])
{
    FILE  *fp;
    char w[MAX];

    if ((fp = fopen(argv[1], "a+") ) == NULL ) {
          fprintf(stderr, "file open error!\n");
                    exit(1);
    }

    puts("Insert a sentence to a file. ");
    puts("(Just press enter to end the input.)\n");

    while (gets(w) != NULL && w[0] != '\0')
                    fprintf(fp, "%s", w);

    puts("file contents :");
    rewind(fp);

    while (fscanf(fp, "%s", w) == 1)
                    puts(w);

    if (fclose(fp) != 0)
          fprintf(stderr, "file close error! \n");

    return 0;
}
```

output:
C:> wordAppend message.txt
Insert a sentence to a file.
Just press enter to end the input.)
Hello world.
Thank you very much.

File Contents:
Hello
world.
Thank
you
very
much.