시스템프로그래밍기초 실습 9주차

HANYANG UNIVERSITY

fopen (stdio.h)

Open the filename pointed to, by filename using the given mode.

```
FILE *fopen(const char *filename, const char *mode)
```

- Parameters

– **filename** : This is the C string containing the name of the file to be opened.

– **mode** : This is the C string containing a file access mode.

Mode	Description
"r"	Opens a file for reading. The file must exist.
"w"	Creates an empty file for writing. If a file with the same name already exists, its content is erased and the file is considered as a new empty file.
"a"	Appends to a file. Writing operations, append data at the end of the file. The file is created if it does not exist.
"r+"	Opens a file to update both reading and writing. The file must exist.
"w+"	Creates an empty file for both reading and writing.
"a+"	Opens a file for reading and appending.

fclose (stdio.h)

Closes the stream. All buffers are flushed.

```
int fclose(FILE *stream)
```

- Parameters
  - **stream** : This is the pointer to a FILE object that specifies the stream to be closed.

- Return Value

  This method returns zero if the stream is successfully closed.

  On failure, EOF is returned.

fscanf & fprintf (stdio.h)

Reads formatted input from a stream.

```
int *fscanf(FILE *stream, const char *format, ...)
```

- Parameters
  - **stream** : This is the pointer to a FILE object that identifies the stream.
  - **format** : type specifiers (형식지정자)
- Return Value : The number of input items successfully matched and assigned.

==========

Sends formatted output to a stream.

```
int *fprintf(FILE *stream, const char *format, ...)
```

- Parameters
  - **stream** : This is the pointer to a FILE object that identifies the stream.
  - **format** : type specifiers (형식지정자)
- Return Value : If successful, the total number of characters written is returned otherwise, a negative number is returned.

실습 예제 1) fio.c

fio.c | double_space.c | dbl_with_caps.c | backward.c | script_week9

```c
#include <stdio.h>

int main(void)
{
    int a, sum = 0;
    FILE *ifp, *ofp;

    ifp = fopen("my_file", "r");    /* open for reading */
    ofp = fopen("outfile", "w");    /* open for writing */

    while (fscanf(ifp, "%d", &a) == 1)
        sum += a;
    fprintf(ofp, "The sum is %d.\n", sum);

    fclose(ifp);
    fclose(ofp);

    return 0;
}

```

실습 예제 1) fio.c 결과

↓ script 실행결과

```
[Ex. 1] fio
(cat my_file)
11
23
7abc
3
(cat outfile)
The sum is 41.
```

getc (stdio.h)

Get the next character (an unsigned char) from the specified stream and advances the position indicator for the stream..

```
int *getc(FILE *stream)
```

- Parameters
  - **stream** : This is the pointer to a FILE object that identifies the stream on which the operation is to be performed.
- Return Value : The character read as an unsigned char cast to an int or EOF on end of file or error

putc (stdio.h)

Writes a character (an unsigned char) specified by the argument char to the specified stream and advances the position indicator for the stream.

```
int putc(int char, FILE *stream)
```

- Parameters
  - **char** : This is the character to be written. The character is passed as its int promotion.
  - **stream** : This is the pointer to a FILE object that identifies the stream where the character is to be written.
- Return Value : If successful, the total number of characters written is returned otherwise, a negative number is returned.

실습 예제 2) double_space.c

fio.c | double_space.c × | dbl_with_caps.c | backward.c | script_week9

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

void double_space(FILE *, FILE *);
void prn_info(char *);

int main(int argc, char **argv)
{
    FILE *ifp, *ofp;

    if (argc != 3) {
        prn_info(argv[0]);
        exit(1);
    }

    ifp = fopen(argv[1], "r");
    ofp = fopen(argv[2], "w");
    double_space(ifp, ofp);
    fclose(ifp);
    fclose(ofp);

    return 0;
}
```

```c
void double_space(FILE *ifp, FILE *ofp)
{
    int c;

    while ((c = getc(ifp)) != EOF) {
        putc(c, ofp);
        if (c == '\n')
            putc('\n', ofp);    /* found a newline - duplicate it */
    }
}

void prn_info(char *pgm_name)
{
    printf("\n%s%s%s\n\n%s%s\n\n",
            "Usage: ", pgm_name, " infile outfile",
            "Ths contents of infile will be double-spaced ",
            "and written to outfile.");
}
```

실습 예제 2) double_space.c 결과

↓ script 실행결과

```
[Ex. 2] double_space
(cat outfile2)
11

23

7abc

3
```

tmpfile & rewind (stdio.h)

Creates a temporary file in binary update mode (wb+). The temporary file created is automatically deleted when the stream is closed (fclose) or when the program terminates.

```
FILE *tmpfile(void)
```

- Return Value : If successful, the function returns a stream pointer to the temporary file created. If the file cannot be created, then NULL is returned.

==========

Sets the file position to the beginning of the file of the given stream.ormatted output to a stream.

```
void rewind(FILE *stream)
```

- Parameters

– **stream** : This is the pointer to a FILE object that identifies the stream.

실습 예제 3) dbl_with_caps.c

dbl_with_caps.c

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

FILE *gfopen(char *filename, char *mode);

int main(int argc, char **argv)
{
    int c;
    FILE *fp, *tmp_fp;

    if (argc != 2) {
        fprintf(stderr, "\n%s%s%s\n\n%s\n\n",
                "Usage: ", argv[0], " filename",
                "The file will be doubled and some letters capitalized.");
        exit(1);
    }

    fp = gfopen(argv[1], "r+");
    tmp_fp = tmpfile();
    while ((c = getc(fp)) != EOF)
        putc(toupper(c), tmp_fp);
    rewind(tmp_fp);
    fprintf(fp, "---\n");
    while ((c = getc(tmp_fp)) != EOF)
        putc(c, fp);

    fclose(fp);
    fclose(tmp_fp);

    return 0;
}
```

실습 예제 3) dbl_with_caps.c

```
34  FILE *gfopen(char *filename, char *mode)
35  {
36      FILE *fp;
37
38      if ((fp = fopen(filename, mode)) == NULL) {
39          fprintf(stderr, "Cannot open %s - bye!\n", filename);
40          exit(1);
41      }
42
43      return fp;
44  }
45
```

실습 예제 3) dbl_with_caps.c 결과

↓ script 실행결과

```
[Ex. 3] dbl_with_caps
(cat my_file3)
A is for apple and alphabet pie.
---
A IS FOR APPLE AND ALPHABET PIE.
```

fseek (stdio.h)

Sets the file position of the **stream** to the given **offset**.

```
int fseek(FILE *stream, long int offset, int whence)
```

- Parameters
  - **stream** – This is the pointer to a FILE object that identifies the stream.
  - **offset** – This is the number of bytes to offset from whence.
  - **whence** – This is the position from where offset is added.

Constant	Description
SEEK_SET	Beginning of file
SEEK_CUR	Current position of the file pointer
SEEK_END	End of file

- Return Value : returns zero if successful, or else it returns a non-zero value.

ftell (stdio.h)

Sets the file position of the **stream** to the given **offset**.

```
long int ftell(FILE *stream)
```

- Parameters
  - **stream** : This is the pointer to a FILE object that identifies the stream.
- Return Value : The current value of the position indicator. If an error occurs, -1L is returned, and the global variable errno is set to a positive value.

putchar (stdio.h)

Writes a character (an unsigned char) specified by the argument char to stdout.

```
int *putchar(int char)
```

- Parameters
  - **char** : This is the character to be written. The character is passed as its int promotion.
- Return Value : The character written as an unsigned char cast to an int or EOF on error.

실습 예제 4) backward.c

fio.c | double_space.c | dbl_with_caps.c | backward.c × | script_week9

```c
#include <stdio.h>
#define MAXSTRING   100

int main(void)
{
    char fname[MAXSTRING];
    int c;
    FILE *ifp;

    fprintf(stderr, "Input a filename: ");
    scanf("%s", fname);
    ifp = fopen(fname, "r");
    fseek(ifp, 0, SEEK_END);        /* move to end of the file */
    fseek(ifp, -1, SEEK_CUR);       /* back up one character */
    while (ftell(ifp) > 0) {
        c = getc(ifp);
        putchar(c);
        fseek(ifp, -2, SEEK_CUR);   /*back up two characters */
    }
    putchar('\n');

    fclose(ifp);

    return 0;
}

```

실습 예제 4) backward.c 결과

↓ script 실행결과

```
[Ex. 4] backward
Input a filename:
3
cba7
32
1
```

echo (shell)

- **echo** : shell에서 텍스트, 변수 출력 시 사용하는 명령어

  -e 옵션: escape문자(\)를 사용할 경우

ex)

```
$echo hello world
$echo –e “hello\nworld”
```

redirection & pipe (shell)

- **redirection : 출력과 입력의 방향을 지정할 때 사용 (>, >>, <)**
  - 명령어 > 파일 : 명령어의 출력값을 파일 내용에 덮어씀.
    (파일이 존재하지 않을 경우, 새로운 파일 생성.)
  - 명령어 >> 파일 : 명령어의 출력값을 파일 내용 뒤에 덧붙임.
  - 명령어 < 파일 : 파일 내용을 명령어의 입력값으로 사용함.

ex)

```
$echo –e “hello\nworld” > my_file
```

- **pipe : 좌측 명령어 실행의 출력을 우측 명령어 실행의 입력으로 사용 (|)**

ex)

```
$ls –al | grep ‘fio’
```

script_week9

HANYANG UNIVERSITY
시스템프로그래밍기초

fio.c | double_space.c | dbl_with_caps.c | backward.c | script_week9 ×

```
1  echo -e "\033[1;36m[Ex. 1] fio\033[0m"
2  gcc -o fio fio.c
3  echo -e "11\n23\n7abc\n3" > my_file
4  echo "(cat my_file)"
5  cat my_file
6  ./fio
7  echo "(cat outfile)"
8  cat outfile
9  echo ""
10
11 echo -e "\033[1;36m[Ex. 2] double_space\033[0m"
12 gcc -o double_space double_space.c
13 cp my_file my_file2
14 cp outfile outfile2
15 ./double_space my_file2 outfile2
16 echo "(cat outfile2)"
17 cat outfile2
18 echo ""
19
20 echo -e "\033[1;36m[Ex. 3] dbl_with_caps\033[0m"
21 gcc -o dbl_with_caps dbl_with_caps.c
22 echo "A is for apple and alphabet pie." > my_file3
23 ./dbl_with_caps my_file3
24 echo "(cat my_file3)"
25 cat my_file3
26 echo ""
27
28 echo -e "\033[1;36m[Ex. 4] backward\033[0m"
29 gcc -o backward backward.c
30 echo "my_file" | ./backward
31
```

과제 제출 방법

1. 모든 파일은 sys_09_학번.tar.gz으로 압축하여 제출한다.
2. 메일 제목은 [시프기]_09_이름_학번으로 한다.
3. 제출 파일들을 빈 디렉토리에 넣고 그 디렉토리 안으로 이동한 후,

   다음과 같이 압축 명령어를 사용한다.(폴더가 아닌 파일들만 압축한다.)

```
$ tar –zcvf sys_09_학번.tar.gz *
```

제출 파일

1. fio.c
2. double_space.c
3. dbl_with_caps.c
4. backward.c
5. script_week9

감사합니다.

HANYANG UNIVERSITY