Data Structures and Algorithms

FILES

API's

Lab Code: **17ECSP201**

File Operations

- Create a file
- Open a file
- Read from a file
- Write to a file
- Move to a specified location in a file
- Close a file

File Pointer

How do I gather the characteristics of a character pointer?

File Structure

```
typedef struct  {
    int level;                  /* fill/empty level of buffer    */
    unsigned  flags;            /* File status flags             */
    char  fd;                   /* File descriptor               */
     unsigned char hold;        /* Ungetc char if no buffer      */
     int bsize;                 /* Buffer size                   */
     unsigned char  *buffer;    /* Data transfer buffer          */
     unsigned char  *curp;      /* Current active pointer        */
     unsigned  istemp;          /* Temporary file indicator      */
     short token;               /* Used for validity checking    */
} FILE;
```

To use files, we need to create a variable of type **FILE.**

File Pointer


```
FILE * fp;
```

File Open API

fopen( );

What parameters do I need to pass to the function?

File Open API

fopen() will need following details:

- Name of the file
- Permissions on the file
- File pointer

Syntax:

fp = fopen(char * filename, char * mode);

File Open permissions

Mode	Operations
r	Reading from the file, file pointer is set to the first character in the file.
w	If file exists then contents are overwritten else a new file is created. File pointer is at beginning.
a	Adding new contents to the end of the file. File pointer points to the end of the file.
r+	Read existing contents, writing new contents, modifying existing contents.
w+	Writing, reading and modifying existing contents. File contents are overwritten if file already exists.
a+	Reading, appending new contents to end of file. Cannot modify existing contents.

File Close API

fclose( );

What parameters do I need to pass to the function?

File Close API

fclose() will need following details:

- File pointer

Syntax:

```
int fclose(FILE *fp);
```

Writing to a File

We use the variant of

printf() **,**

called ***fprintf()***

The question is about parameters passed!

Writing to a File Contd..,

fprintf(file pointer, “control string”, variable list);

Example:

fprintf(fp, “%d”, num);

Reading from a File

We use the variant of

scanf() **,**

called ***fscanf()***

The question is about parameters passed!

Reading from a File Contd..,

fscanf(file pointer, “control string”, variable list);

Example:

fscanf(fp, “%d”, &num);

Error Handling, Why?

- Unable to open the file
- Reading beyond the end of the file
- Invalid file name
- No permissions etc.

Error Handling API's

- **To check for error in file**

ferror(fp);

- **To check for end of the file**

feof(fp);

Random Access

- **To know the current position of file pointer**

long ftell(FILE *fp);

Example:

long pos = ftell (fp);

- **To move the file pointer to the starting position**

rewind(FILE *fp);

Example:

rewind(fp);

Random Access Contd..,

- **To seek to the required position in file**

int fseek(FILE * fp, long offset, int start_point);

Parameters:

fp	File pointer
offset	Number of bytes to move. It can take positive, negative or zero value
start_point	0 → beginning 1 → current position 2 → end of the file

Random Access Contd..,

- **To seek to the required position in file**

int fseek(FILE * fp, long offset, int start_point);

Parameters:

fp	File pointer
offset	Number of bytes to move. It can take positive, negative or zero value
start_point	0 → beginning 1 → current position 2 → end of the file

Random Access Contd..,

Examples:

fseek(fp, 10, 1);

Move 10 bytes forward from current position

fseek(fp, -5, 2);

Move 5 bytes backward from end of the file

fseek(fp, 0, 0);

Move 0 bytes from start of the file

Thank you.

(File Programs in the Next Lab session)