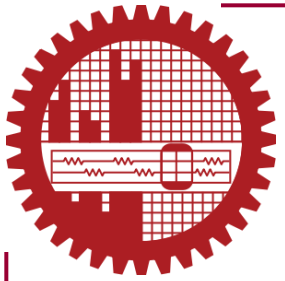

ICT 5101

Lecture 11

Dr. Hossen A Mustafa

Input and Output

- Standard I/O
  - printf
  - scanf
  - getchar
  - putchar
  - etc.
- File I/O
  - Read from a file from the filesystem
  - Write a new file
  - Append to an existing file

Reading File

- Variable

  FILE *fp;

- Function

  FILE *fopen(char *name, char *mode);

- Code Example:

```
FILE * fp; // variable declaration for file pointer
fp = fopen("input.txt", "r"); // open the file in read mode
fscanf(fp, "%d", &n); // read a value from file
fclose(fp);
```

Reading File Example

```
int main(){
    int n;
    FILE * fp;
    fp = fopen(“input.txt”, "r");
    if (fp == NULL) {
        printf(“can't open\n,);
        return 1;
    }
    fscanf(fp, “%d”, &n);
    printf(“n = %d\n”, n);
    fclose(fp);
    return 0;
}
```

Reading File

- Functions to read
  - fscanf(FILE *fp, char* format)
  - fgetc(FILE *fp)
  - fgets(char *array, int size, FILE *fp)

Writing File

- Variable

  FILE *fp;

- Function

  FILE *fopen(char *name, char *mode);

- Code Example:

```
FILE * fp; // variable declaration for file pointer
fp = fopen("output.txt", "w"); // open file in write mode
fprintf(fp, "The value of n is %d", n); // write to file
fclose(fp);
```

Writing File Example

```
int main(){
    int n = 500;
    FILE * fp;
    fp = fopen("output.txt", "w");
    if (fp == NULL) {
        printf("can't open\n,);
        return 1;
    }
    fprintf("n = %d\n", n);
    fclose(fp);
    return 0;
}
```

Writing File

- Functions to read
  - fprintf(FILE *fp, char* format)
  - fputc(FILE *fp)
  - fputs(char *array, FILE *fp)

Appending File

- Variable

  FILE *fp;

- Function

  FILE *fopen(char *name, char *mode);

- Code Example:

```
FILE * fp; // variable declaration for file pointer
fp = fopen("output.txt", "a"); // open file in write mode
fprintf(fp, "The value of n is %d", n); // write to file
fclose(fp);
```

Appending File Example

```
int main(){
    int n = 500;
    FILE * fp;
    fp = fopen("output.txt", "a");
    if (fp == NULL) {
        printf("can't open\n,);
        return 1;
    }
    fprintf("n = %d\n", n);
    fclose(fp);
    return 0;
}
```

** New write will be appended at the end of the file

More about fopen()

- For “r” mode,
  - fopen returns NULL if the file cannot be found
- For “w” mode,
  - fopen will create a new file with the filename
  - If file exists, that will be replaced
- For “a” mode,
  - fopen will open the file if exists
  - If the file doesn’t exist, a new file will be created
- C allows to open the same file in “r”, “w” mode
  - But, it should be avoided if possible

fflush and fclose

- When we do a write to a file, e.g., using fprintf
  - The OS doesn't write it instantly in the file
  - The OS maintains a buffer, and writes to file when the buffer crosses a limit or when fclose is used
  - This is to maintain OS performance
- We can use fflush(FILE *fp) to force the OS to write to a file. fflush function must be used after the fprintf
- We must use fclose when file operations are complete
  - Otherwise, the OS may not write the file at all

Class Assignment

- Write a program named classassignment11.c
- The program should
  - read file password.txt (available in LMS). Each line of the file contains a password.
  - save the passwords in an array of strings (2D char array)
  - prompt user to enter a password from keyboard
  - match the user's password with the passwords in the array (use strcmp function from string.h)
  - If password matches, show success message
  - Otherwise, append the password in the password.txt