**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **20/06/2020** | **Name:** | **Krishna Swetha** |
| **Course:** | **Solo learning** | **USN:** | **4AL16EC032** |
| **Topic:** | **Files and Error**  **Handling,Pre-Processor** | **Semester & Section:** | **6th,B** |
| **Github Repository:** | **Krishna-Swetha** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
|  |
| **Report –**  **Accessing Files**  **An external file can be opened, read from, and written to in a C program. For these**  **operations, C includes the FILE type for defining a file stream. The file stream keeps track of**  **where reading and writing last occurred.**  **The stdio.h library includes file handling functions:**  **FILE Typedef for defining a file pointer.**  **fopen(filename, mode) Returns a FILE pointer to file filename which is opened using mode. If**  **a file cannot be opened, NULL is returned.**  **Mode options are:**  **- r open for reading (file must exist)**  **- w open for writing (file need not exist)**  **- a open for append (file need not exist)**  **- r+ open for reading and writing from beginning**  **- w+ open for reading and writing, overwriting file**  **- a+ open for reading and writing, appending to file**  **fclose(fp) Closes file opened with FILE fp, returning 0 if close was successful. EOF (end of file)**  **is returned if there is an error in closing.**  **#include <stdio.h>**  **int main() {**  **FILE \*fptr;**  **fptr = fopen("myfile.txt", "w");**  **if (fptr == NULL) {**  **printf("Error opening file.");**  **return -1;**  **}**  **fclose(fptr);**  **return 0;**  **}**  **Reading from a File**  **The stdio.h library also includes functions for reading from an open file. A file can be read one**  **character at a time or an entire string can be read into a character buffer, which is typically a**  **char array used for temporary storage.**  **fgetc(fp) Returns the next character from the file pointed to by fp. If the end of the file has**  **been reached, then EOF is returned.**  **fgets(buff, n, fp) Reads n-1 characters from the file pointed to by fp and stores the string in**  **buff. A NULL character '\0' is appended as the last character in buff. If fgets encounters a**  **newline character or the end of file before n-1 characters is reached, then only the characters**  **up to that point are stored in buff.**  **fscanf(fp, conversion\_specifiers, vars) Reads characters from the file pointed to by fp and**  **assigns input to a list of variable pointers vars using conversion\_specifiers. As with scanf,**  **fscanf stops reading a string when a space or newline is encountered.** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | |
|  |  |  |  | |
|  |  |  |  | |
|  | | | |
|  | | | |
|  | | | |