PROGRAMMING LAB ASSIGNMENT - 10

NAME- RUPAYAN THAKUR CHAKRABORTY

ENROLLMENT NUMBER- 2020ITB028

DEPARTMENT OF INFORMATION TECHNOLOGY

3 RD SEMESTER, 2 ND YEAR

G-Suite id- 2020itb028.rupayan@students.iiests.ac.in

INDIAN INSTITUTE OF ENGINEERING SCIENCE AND TECHNOLOGY (IIEST)

Write Short Notes On the following topic:

All the functions discussed below can be used after importing the *fnctl.h* header file. *fcntl.h* is the header in the C library that contains constructs that refer to file control for example - opening a file, retrieving and changing the permissions of file, locking a file for edit and more such operations.

1) read():

The *read*() function reads data previously written to a file. If any portion of a regular file prior to the end-of-file has not been written, *read*() shall return bytes with value 0. If data is later written at this point, subsequent reads in the gap between the previous end of data and the newly written data shall return bytes with value 0 until data is written into the gap.

Syntax in C language

size_t read (int fd, void* buf, size_t cnt);

Parameters:

- **fd: file descriptor**
- **buf: buffer to read data from**
- **cnt: length of buffer**

Returns: How many bytes were actually read

- **return Number of bytes read on success**
- **return 0 on reaching end of file**
- **return -1 on error**
- **return -1 on signal interrupt**

2) creat():

Used to Create a new empty file.

Definition: size_t read (int fd, void* buf, size_t cnt);

This function will read the file indicated by the file descriptor fd. It reads the file for cnt number of bytes and loads into the memory buffer pointed by buf. A successful read updates the last access time for the file. It returns 0 if end of file is reached, -1 on error.

3) write():

Definition: size_t write (int fd, void* buf, size_t cnt);

This function will write data into the file indicated by the file descriptor argument. It writes cnt number of bytes from the memory buffer pointed by buf into the file or socket associated with fd. However, cnt cannot be greater than INT_MAX. If cnt is 0 or on reaching the end of file write() simply returns 0, else it will return the number of bytes written

successfully. It returns a -1 on error. For write() to work, file should be open and buf should have the number of bytes specified by cnt.

4) read():

Definition: size_t read (int fd, void* buf, size_t cnt);

This function will read the file indicated by the file descriptor fd. It reads the file for cnt number of bytes and loads into the memory buffer pointed by buf. A successful read updates the last access time for the file. It returns 0 if end of file is reached, -1 on error.

5) close():

Definition: int close(int fd);

This function tells the operating system to close the file pointed by the argument fd. It returns a 0 on success and -1 on failure. It simply sets the file table entry at fd to NULL hence closing the file.

QUESTION 2:

Write a program using this five functions to count the numbers of characters in a file. If the number is even then append two arbitrary characters; otherwise append single character in thatOUTPUT: file.

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

void main()
{
   char* path= "file.txt";
   int fd= open(path, O_RDWR | O_APPEND);
   if(fd == -1)
   {
      printf("File does not exist.");
      exit(0);
   }
   char* read_buffer= malloc(300*sizeof(char));
```

```
    int wrds= read(fd, read_buffer, 500);
    printf("Number of words in file: %d\n", wrds);
    int write_counter;
    char* write_buffer= "ab";

    if(wrds%2==0)
    {
        printf("Appending 2 characters...\n");
        write_counter= write(fd, write_buffer, 2);
    }
    else
    {
        printf("Appending 1 character...\n");
        write_counter= write(fd, write_buffer, 1);
    }
    close(fd);
    printf("Task Completed Successfully");
}
```

OUTPUT: