192311035

M.RAVI SAI VINAY

EXPERIMENT-25

Construct a C program to implement the I/O system calls of UNIX (fcntl, seek, stat, opendir, readdir)

AIM:-

To construct a C program that demonstrates the use of I/O system calls in UNIX, such as fcntl, lseek, stat, opendir, and readdir.

ALGORITHM:-

1. Open a File:
2. Use open to open an existing file.
3. Use fcntl to modify file descriptors.
4. Seek File Position:
5. Use lseek to move the file pointer to a specific position.
6. Retrieve File Information:
7. Use stat to fetch metadata (e.g., file size, permissions).
8. Open and Read Directory:
9. Use opendir to open a directory.
10. Use readdir to read and list the files in the directory.
11. Close Resources:
12. Close files and directories using close and closedir

CODE:-

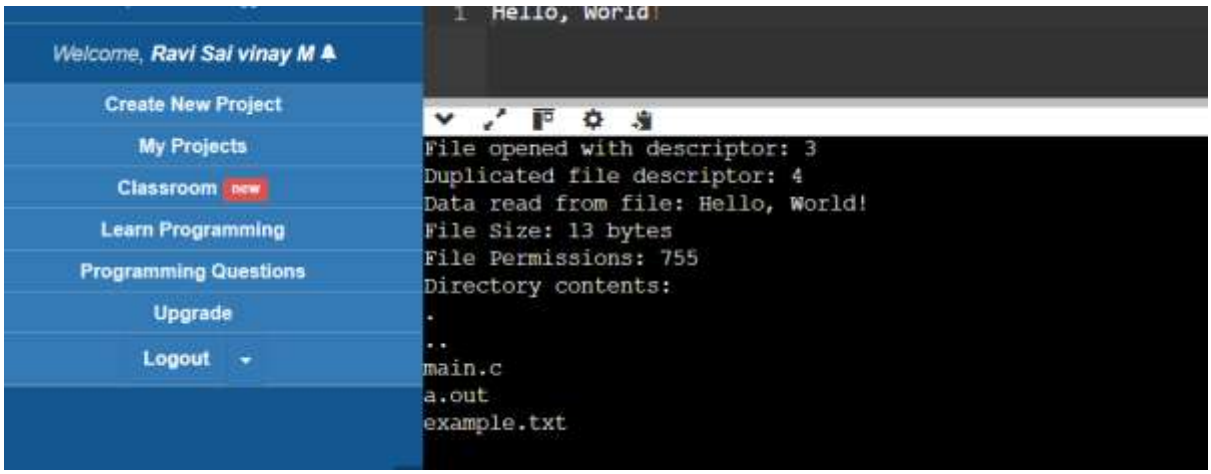
```
#include <stdio.h>

#include <fcntl.h>

#include <unistd.h>

#include <sys/stat.h>
```

```
#include <dirent.h>
#include <string.h>

int main() {
    int fd;
    struct stat fileStat;
    DIR *dir;
    struct dirent *entry;
    char buffer[50];

    // Demonstrate 'open' and 'fcntl'
    fd = open("example.txt", O_RDWR | O_CREAT, 0777);
    if (fd < 0) {
        perror("Error opening file");
        return 1;
    }
    printf("File opened with descriptor: %d\n", fd);

    // Demonstrate 'fcntl' - duplicating the file descriptor
    int new_fd = fcntl(fd, F_DUPFD, 0);
    if (new_fd < 0) {
        perror("Error duplicating file descriptor");
        return 1;
    }
```

```
printf("Duplicated file descriptor: %d\n", new_fd);

// Demonstrate 'lseek'
lseek(fd, 0, SEEK_SET);
write(fd, "Hello, World!", 13);
lseek(fd, 0, SEEK_SET); // Reset file pointer
read(fd, buffer, 13);
buffer[13] = '\0';
printf("Data read from file: %s\n", buffer);

// Demonstrate 'stat'
if (stat("example.txt", &fileStat) == 0) {
    printf("File Size: %ld bytes\n", fileStat.st_size);
    printf("File Permissions: %o\n", fileStat.st_mode & 0777);
} else {
    perror("Error retrieving file information");
}

// Demonstrate 'opendir' and 'readdir'
dir = opendir(".");
if (dir == NULL) {
    perror("Error opening directory");
    return 1;
}
```

```
    printf("Directory contents:\n");
    while ((entry = readdir(dir)) != NULL) {
        printf("%s\n", entry->d_name);
    }

    // Close file and directory
    close(fd);
    close(new_fd);
    closedir(dir);

    return 0;
}
```

OUTPUT:-



RESULT:-

The program successfully demonstrated the use of UNIX I/O system calls such as fcntl, lseek, stat, opendir, and readdir. These system calls facilitated file and directory operations in a UNIX-based environment.