

Name: T. Nikhil Kumar Reddy

Reg-No: 192372024

26. Construct a C program to implement the file management operations.

Aim

To develop a C program to perform basic file management operations: create, read, write, and append data to a file.

Algorithm

1. Start the program.
2. Display a menu for file operations (Create/Write, Read, Append, Exit).
3. Based on the user's choice:
 - **Create/Write:** Open a file in write mode, input data, and save it.
 - **Read:** Open a file in read mode and display its contents.
 - **Append:** Open a file in append mode and add new data.
4. Close the file after each operation.
5. Repeat until the user chooses to exit.
6. End the program.

Procedure

1. Use `fopen()` to create/open a file.
2. Perform operations using `fprintf()` for writing, `fscanf()` or `fgets()` for reading, and `fprintf()` for appending.
3. Handle user inputs and perform error checking (e.g., file not found).
4. Close the file using `fclose()`.

Code:

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

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

```
void createFile() {
```

```
    FILE *file = fopen("file.txt", "w");
```

```
    if (file == NULL) {
```

```
        printf("Error creating file.\n");
```

```
        return;
    }

    printf("File created successfully.\n");

    fclose(file);
}
```

```
void writeFile() {

    FILE *file = fopen("file.txt", "w");

    if (file == NULL) {

        printf("Error opening file.\n");

        return;

    }

    char data[100];

    printf("Enter content to write into the file: ");

    getchar();

    fgets(data, 100, stdin);

    fprintf(file, "%s", data);

    printf("Data written successfully.\n");

    fclose(file);

}
```

```
void readFile() {

    FILE *file = fopen("file.txt", "r");
```

```
if (file == NULL) {  
    printf("Error opening file.\n");  
    return;  
}  
  
char ch;  
  
printf("File content:\n");  
  
while ((ch = fgetc(file)) != EOF) {  
    putchar(ch);  
}  
  
fclose(file);  
}  
  
void appendFile() {  
    FILE *file = fopen("file.txt", "a");  
  
    if (file == NULL) {  
        printf("Error opening file.\n");  
        return;  
    }  
  
    char data[100];  
  
    printf("Enter content to append to the file: ");  
  
    getchar();  
  
    fgets(data, 100, stdin);  
  
    fprintf(file, "%s", data);  
}
```

```
printf("Data appended successfully.\n");

fclose(file);

}


int main() {

    int choice;

    do {

        printf("\nFile Management System\n");

        printf("1. Create File\n");

        printf("2. Write to File\n");

        printf("3. Read File\n");

        printf("4. Append to File\n");

        printf("5. Exit\n");

        printf("Enter your choice: ");

        scanf("%d", &choice);

        switch (choice) {

            case 1: createFile(); break;

            case 2: writeFile(); break;

            case 3: readFile(); break;

            case 4: appendFile(); break;

            case 5: printf("Exiting...\n"); break;

            default: printf("Invalid choice. Try again.\n");

        }

    }
```

```

    } while (choice != 5);

    return 0;

}

```

Result

The program successfully implements file management operations:

1. **Create File:** Creates an empty file named `file.txt`.
2. **Write File:** Writes user input to the file.
3. **Read File:** Reads and displays the file's contents.
4. **Append File:** Appends additional content to the file.

Output:

The screenshot shows the OnlineGDB web interface. On the left is a sidebar with navigation links: 'Create New Project', 'My Projects', 'Classroom' (marked as 'new'), 'Learn Programming', 'Programming Questions', 'Upgrade', and 'Logout'. The main area displays the output of a C++ program. The program is a 'File Management System' with a menu: 1. Create File, 2. Write to File, 3. Read File, 4. Append to File, 5. Exit. The user enters choice 2, then provides content to write to the file, and sees 'Data written successfully.'. The user then enters choice 6, which is invalid. Finally, the user enters choice 5, and the program exits.

```

File Management System
1. Create File
2. Write to File
3. Read File
4. Append to File
5. Exit
Enter your choice: 2
Enter content to write into the file: 4
Data written successfully.

File Management System
1. Create File
2. Write to File
3. Read File
4. Append to File
5. Exit
Enter your choice: 6
Invalid choice. Try again.

< File Management System
1. Create File
2. Write to File
3. Read File
4. Append to File
5. Exit
Enter your choice: 5
Exiting...

```