KASIRALA DURGA-192324073

1. Illustrate the various File Access Permission and different types of users in Linux.

# Aim

To understand and demonstrate file access permissions and the types of users in Linux.

# File Access Permissions

In Linux, file permissions define how files and directories are accessed by users. These permissions are represented as:

* + **Read (r)**: Allows viewing the content of a file or directory.
  + **Write (w)**: Allows modifying the content of a file or adding/deleting files in a directory.
  + **Execute (x)**: Allows running a file as a program or accessing a directory.

# Permission Categories

1. **Owner (u)**: The user who owns the file.
2. **Group (g)**: A group of users with shared access.
3. **Others (o)**: All other users on the system.

Permissions are displayed using the **ls -l** command, where: diff

Copy code

-rwxr-xr--

* + **First character**: File type (- for a file, d for a directory).
  + **Next 3 characters**: Permissions for the owner (e.g., rwx).
  + **Next 3 characters**: Permissions for the group (e.g., r-x).
  + **Last 3 characters**: Permissions for others (e.g., r--).

# Algorithm

1. Open a terminal and create a file/directory using touch or mkdir.
2. Check the current permissions using the ls -l command.
3. Modify permissions using the chmod command.
   * chmod [permissions] [filename]
   * Permissions can be set symbolically (u, g, o) or numerically (e.g., 777).
4. Validate the changes by checking permissions again with ls -l.

# Code

# #include <stdio.h>

# #include <stdlib.h>

# #include <string.h>

# // Structure to represent a file and its permissions

# struct File {

# char name[50];

# int owner; // Permissions for owner

# int group; // Permissions for group

# int others; // Permissions for others

# };

# // Function to display permissions

# void displayPermissions(struct File file) {

# char permissionString[10];

# // Convert numeric permissions to symbolic form

# snprintf(permissionString, sizeof(permissionString), "%c%c%c%c%c%c%c%c%c",

# (file.owner & 4) ? 'r' : '-',

# (file.owner & 2) ? 'w' : '-',

# (file.owner & 1) ? 'x' : '-',

# (file.group & 4) ? 'r' : '-',

# (file.group & 2) ? 'w' : '-',

# (file.group & 1) ? 'x' : '-',

# (file.others & 4) ? 'r' : '-',

# (file.others & 2) ? 'w' : '-',

# (file.others & 1) ? 'x' : '-');

# printf("File: %s\nPermissions: %s\n", file.name, permissionString);

# }

# // Function to modify permissions

# void modifyPermissions(struct File \*file, char userType, char operation, char permission) {

# int \*target;

# if (userType == 'u')

# target = &file->owner;

# else if (userType == 'g')

# target = &file->group;

# else if (userType == 'o')

# target = &file->others;

# else {

# printf("Invalid user type!\n");

# return;

# }

# int bit;

# if (permission == 'r')

# bit = 4;

# else if (permission == 'w')

# bit = 2;

# else if (permission == 'x')

# bit = 1;

# else {

# printf("Invalid permission type!\n");

# return;

# }

# if (operation == '+')

# \*target |= bit;

# else if (operation == '-')

# \*target &= ~bit;

# else {

# printf("Invalid operation!\n");

# return;

# }

# printf("Permissions updated successfully!\n");

# }

# int main() {

# struct File file;

# // Initialize file

# printf("Enter file name: ");

# scanf("%s", file.name);

# file.owner = 6; // Default: rw-

# file.group = 4; // Default: r--

# file.others = 4; // Default: r--

# int choice;

# do {

# printf("\nFile Permission Management:\n");

# printf("1. Display Permissions\n");

# printf("2. Modify Permissions\n");

# printf("3. Exit\n");

# printf("Enter your choice: ");

# scanf("%d", &choice);

# switch (choice) {

# case 1:

# displayPermissions(file);

# break;

# case 2: {

# char userType, operation, permission;

# printf("Enter user type (u: owner, g: group, o: others): ");

# scanf(" %c", &userType);

# printf("Enter operation (+ or -): ");

# scanf(" %c", &operation);

# printf("Enter permission (r, w, x): ");

# scanf(" %c", &permission);

# modifyPermissions(&file, userType, operation, permission);

# break;

# }

# case 3:

# printf("Exiting...\n");

# break;

# default:

# printf("Invalid choice!\n");

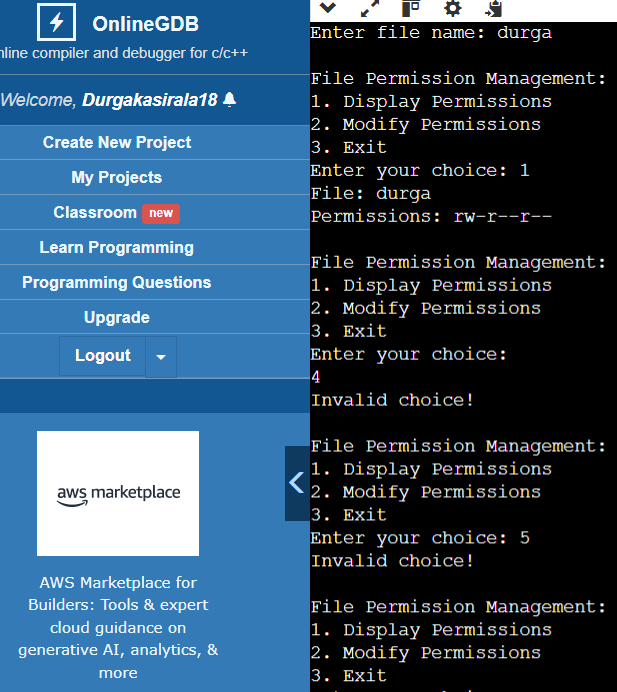
# }

# } while (choice != 3);

# return 0;

# }

# Output



# Result

* Demonstrated the default file permissions in Linux.
* Successfully modified file permissions using both symbolic and numeric modes.
* Observed how permissions affect access for the owner, group, and others.