**Description:**

The code provides a menu-driven function to perform file operations in C. It includes functions to create a file, read from a file, write to a file, obtain statistical information about a file, create and use named pipes for inter-process communication, and copy the content of one file to another using unnamed pipes.

**Header Files:**

* **<unistd.h>** : Header file for the following system calls;

int read(int fd, void \*buf, size\_t count)

int write(int fd, const void \*buf, size\_t count)

int close(int fd)

off\_t lseek(int fd, off\_t offset, int whence)

pid\_t fork(void)

void \_exit(int status)

int pipe(int pipefd[2]);

* **<fcntl.h>**

int open(const char \*pathname, int flags, mode\_t )

* **<stdio.h>**

int printf(const char \*format, ...)

* **<sys/types.h>** and **<sys/stat.h>**

int stat(const char \*pathname, struct stat \*statbuf)

int mkfifo(const char \*pathname, mod\_t mode);

* **<time.h>**

char \*ctime(const time\_t \*time)

* **<stdlib.h>**

void perror(const char \*s)

int atoi(const char \*str): Converts the string str to an integer representation.

long strtol(const char \*str, char \*\*endptr, int base): Converts the initial part of the string str to a long integer representation.

* **<string.h>**

int strcmp(const char \*s1, const char \*s2): Compares two strings.

size\_t strlen(const char \*str): Calculates the length of the null-terminated string.

* **main.h**: This header file contains function declarations and includes the required system headers.

**Functions:**

* **void create(char\* filename, mode\_t permissions):**

**Purpose**: Creates a new file with the given filename and permissions.

**Parameters**:

filename: The name of the file to be created.

permissions: The permission mode for the file (e.g., 0777 for read, write, and execute permissions for all).

**Return**: None

**Error Handling:** Exits the program with an error message if file creation fails.

* **void readFile(char\* filename, int size, int offset, int whence):**

**Purpose**: Reads data from a file and prints it to the console.

**Parameters**:

filename: The name of the file to read from.

size: The number of bytes to read from the file.

offset: The offset from where to start reading in the file.

whence: The reference point for seeking the offset (e.g., SEEK\_SET, SEEK\_CUR, SEEK\_END).

**Return**: None

**Error Handling:** Exits the program with an error message if file reading or opening fails.

* **void writeFile(char\* filename, char\* data, int offset, int whence):**

**Purpose:** Writes data to a file from the given buffer.

**Parameters**:

filename: The name of the file to write to.

data: The data to write into the file.

offset: The offset from where to start writing in the file.

whence: The reference point for seeking the offset (e.g., SEEK\_SET, SEEK\_CUR, SEEK\_END).

**Return**: None

**Error Handling**: Exits the program with an error message if file writing or opening fails.

* **void statInfo(char\* filename):**

**Purpose**: Prints statistical information about the given file.

**Parameters**:

filename: The name of the file to obtain statistical information about.

**Return**: None

**Error Handling**: Exits the program with an error message if the file does not exist.

* **void namedpipe(char\* pipename, mode\_t permission, char\* operation):**

**Purpose**: Creates and uses a named pipe for inter-process communication.

**Parameters**:

pipename: The name of the named pipe to create or use.

permission: The permission mode for the named pipe (e.g., 0777 for read, write, and execute permissions for all).

operation: The operation to perform on the named pipe (e.g., "read" or "write").

**Return**: None

**Error Handling**: Exits the program with an error message if named pipe creation or operation fails.

* **void copy(char\* filename1, char\* filename2):**

**Purpose**: Copy the content of one file to another using an unnamed pipe.

**Parameters**:

filename1: The name of the source file to copy from.

filename2: The name of the destination file to copy to.

**Return**: None

**Error Handling**: Prints an error message if the destination file already exists.

**To compile type:**

gcc -o final.exe main.c scalls.c

**To execute\run type:**

./final.exe