Exp:24

Code:

#include <stdio.h>

#include <stdlib.h>

#include <fcntl.h>

#include <unistd.h>

#include <string.h>

#define BUFFER\_SIZE 1024

void error(const char \*msg) {

perror(msg);

exit(EXIT\_FAILURE);

}

int main() {

int fd;

char buffer[BUFFER\_SIZE];

ssize\_t bytesRead, bytesWritten;

fd = open("example.txt", O\_WRONLY | O\_CREAT | O\_TRUNC, 0644);

if (fd == -1) {

error("Error opening file for writing");

}

const char \*text = "Hello, this is a demonstration of Unix system calls for file management.\n";

bytesWritten = write(fd, text, strlen(text));

if (bytesWritten == -1) {

error("Error writing to file");

}

printf("Wrote %ld bytes to example.txt\n", bytesWritten);

if (close(fd) == -1) {

error("Error closing file");

}

fd = open("example.txt", O\_RDONLY);

if (fd == -1) {

error("Error opening file for reading");

}

bytesRead = read(fd, buffer, BUFFER\_SIZE - 1);

if (bytesRead == -1) {

error("Error reading from file");

}

buffer[bytesRead] = '\0';

printf("Read %ld bytes: %s\n", bytesRead, buffer);

if (lseek(fd, 0, SEEK\_SET) == -1) {

error("Error seeking in file");

}

bytesRead = read(fd, buffer, BUFFER\_SIZE - 1);

if (bytesRead == -1) {

error("Error reading from file");

}

buffer[bytesRead] = '\0';

printf("Read again %ld bytes: %s\n", bytesRead, buffer);

if (close(fd) == -1) {

error("Error closing file");

}

return 0;

}

Output:

