SDM COLLEGE OF ENGINEERING AND TECHNOLOGY

Dhavalagiri, Dharwad-580002, Karnataka State, India.

Email: cse.sdmcet@gmail.com

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

A Report on CTA-Assignment

COURSE CODE: 22UCSL504 COURSE TITLE: DBMS
SEMESTER: 5th DIVISION: A
COURSE TEACHER: Dr. U. P. Kulkarni

[Academic Year- 2024-25]
Date of Submission: 01-11-2024

Submitted
By

Mr. Deepak Kumar P S USN: 2SD22CS024

Table of content

1. File operations in C programming:

File operations in C programming are a key part of input/output (I/O) management, allowing programs to store, retrieve, and manipulate data in files. The concept is based on standard libraries that provide functions to interact with files in a structured manner.

File Operations:

a. Creating the file.
b. Opening of the file.
c. Closing of the file.
d. Reading from the file.
e. Writing to the file.

Problem Statement – 1: Write the C program to study all file operations related system Calls supported by UNIX OS and C libraries for file operations.

```
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <string.h>
void create_file(const char *filename) {
   int fd = creat(filename, 0644);
   if (fd == -1) {
      perror("Error creating file");
   } else {
      printf("File created successfully\n");
      close(fd);
   }
}
void open_file(const char *filename) {
   int fd = open(filename, O_RDONLY);
   if (fd == -1) {
      perror("Error opening file");
   } else {
      printf("File opened successfully\n");
      close(fd);
   }
```

```
}
void read_file(const char *filename) {
    int fd = open(filename, O_RDONLY);
    if (fd == -1) {
        perror("Error opening file");
    } else {
        char buffer[1024];
        ssize_t bytes_read = read(fd, buffer, sizeof(buffer));
        if (bytes_read == -1) {
            perror("Error reading from file");
        } else {
            printf("Read %zd bytes: %s\n", bytes_read, buffer);
        }
        close(fd);
    }
}

void write_file(const char *filename) {
    int fd = open(filename, O_WRONLY | O_CREAT, 0644);
    if (fd == -1) {
        perror("Error opening file");
    } else {
        char text[1024];
        printf("Enter text to write to file: ");
        fgets(text, sizeof(text), stdin);
        ssize_t bytes_written = write(fd, text, strlen(text));
        if (bytes_written == -1) {
            perror("Error writing to file");
        } else {
            printf("Wrote %zd bytes to file\n", bytes_written);
        }
        close(fd);
    }
}
void delete_file(const char *filename) {
    if (unlink(filename) == -1) {
        perror("Error deleting file");
    } else {
        printf("File deleted successfully\n");
    }
```

```
}

int main() {
    char filename[1024];

    printf("Enter filename: ");
    fgets(filename, sizeof(filename), stdin);
    filename[strcspn(filename, "\n")] = 0;

    int choice;
    while (1) {
        printf("\nFile Operations:\n");
        printf("1. Create a file\n");
        printf("2. Open a file\n");
        printf("3. Read from file\n");
        printf("4. Write to file\n");
        printf("5. Delete file\n");
        printf("6. Quit\n");
        printf(“Enter your choice:”)
        scanf("%d", &choice);
        getchar();

        switch (choice) {
            case 1:
                create_file(filename);
                break;
            case 2:
                open_file(filename);
                break;
            case 3:
                read_file(filename);
                break;
            case 4:
                write_file(filename);
                break;
            case 5:
                delete_file(filename);
                break;
            case 6:
                return 0;
```

```
            default:
                printf("Invalid choice. Please try again.\n");
        }
    }

    return 0;
}
```

Output:

```
Enter filename: Hi.txt
File Operations:
1. Create a file
2. Open a file
3. Read from file
4. Write to file
5. Delete file
6. Quit
Enter your choice:1
File created successfully

File Operations:
1. Create a file
2. Open a file
3. Read from file
4. Write to file
5. Delete file
6. Quit
Enter your choice:3
Read 0 bytes: ���  //Due to random file.

File Operations:
1. Create a file
2. Open a file
3. Read from file
4. Write to file
5. Delete file
6. Quit
Enter your choice:4
Enter text to write to file: Hi there
Wrote 9 bytes to file
```

File Operations:
1. Create a file
2. Open a file
3. Read from file
4. Write to file
5. Delete file
6. Quit
Enter your choice:3
Read 9 bytes: Hi there

File Operations:
1. Create a file
2. Open a file
3. Read from file
4. Write to file
5. Delete file
6. Quit
Enter your choice:5
File deleted successfully

File Operations:
1. Create a file
2. Open a file
3. Read from file
4. Write to file
5. Delete file
6. Quit
Enter your choice:6
Exiting program.

2. File indexing:

File indexing refers to a technique used to organize and manage files in a way that enables quick and efficient access to data stored on storage devices. It involves creating a structured data map (index) that acts as a reference for locating file contents, metadata, or other associated information.

Problem Statement – 2.Write a C program to demonstrate file indexing and associated operations.

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

#define MAX_RECORDS 100
#define MAX_LENGTH 100

typedef struct {
    char key[MAX_LENGTH];
    long position;
} Index;

Index indexTable[MAX_RECORDS];
int indexCount = 0;

void createIndex(const char *filename) {
    FILE *fp = fopen(filename, "r");
    if (fp == NULL) {
        perror("Error opening file");
        return;
    }

    char line[MAX_LENGTH];
    long position;
    while ((position = ftell(fp)) != -1 && fgets(line, MAX_LENGTH, fp)) {

        char key[MAX_LENGTH];
        sscanf(line, "%s", key);

        strcpy(indexTable[indexCount].key, key);
```

```
        indexTable[indexCount].position = position;
        indexCount++;
    }

    fclose(fp);
    printf("Index created successfully with %d records.\n", indexCount);
}

void searchRecord(const char *filename, const char *key) {
    FILE *fp = fopen(filename, "r");
    if (fp == NULL) {
        perror("Error opening file");
        return;
    }

    for (int i = 0; i < indexCount; i++) {
        if (strcmp(indexTable[i].key, key) == 0) {
            fseek(fp, indexTable[i].position, SEEK_SET);
            char line[MAX_LENGTH];
            fgets(line, MAX_LENGTH, fp);
            printf("Record found: %s", line);
            fclose(fp);
            return;
        }
    }

    printf("Record with key '%s' not found.\n", key);
    fclose(fp);
}

void displayIndex() {
    printf("Index Table:\n");
    for (int i = 0; i < indexCount; i++) {
        printf("Key: %s, Position: %ld\n", indexTable[i].key, indexTable[i].position);
    }
}

int main() {
    const char *filename = "records.txt";
    int choice;
```

```
    char key[MAX_LENGTH];
    do {
        printf("\nFile Indexing System\n");
        printf("1. Create Index\n");
        printf("2. Search Record\n");
        printf("3. Display Index\n");
        printf("4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                createIndex(filename);
                break;
            case 2:
                printf("Enter the key to search: ");
                scanf("%s", key);
                searchRecord(filename, key);
                break;
            case 3:
                displayIndex();
                break;
            case 4:
                printf("Exiting program.\n");
                break;
            default:
                printf("Invalid choice! Try again.\n");
        }
    } while (choice != 4);

    return 0;
}
```

File : records.txt

```
101 Aman
102 Smith
103 John
104 Suresh
105 Deepak
```

Output:

File Indexing System
1. Create Index
2. Search Record
3. Display Index
4. Exit
Enter your choice: 1
Index created successfully with 4 records.

File Indexing System
1. Create Index
2. Search Record
3. Display Index
4. Exit
Enter your choice: 3
Index Table:
Key: 101, Position: 0
Key: 102, Position: 9
Key: 103, Position: 19
Key: 104, Position: 28

File Indexing System
1. Create Index
2. Search Record
3. Display Index
4. Exit
Enter your choice: 2
Enter the key to search: 101
Record found: 101 Aman

File Indexing System
1. Create Index
2. Search Record
3. Display Index
4. Exit
Enter your choice: 4
Exiting program.

3. Accessing Excel file:

Problem Statement – 3. Write the Java program to access the given excel file with known file format.

```
import org.apache.poi.ss.usermodel.*;
import org.apache.poi.xssf.usermodel.XSSFWorkbook;

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;

public class ExcelReader {
    public static void main(String[] args) {
        String excelFilePath = "/Dbms-minor/file.xlsx"; // Update this to the correct path
        FileInputStream fileInputStream = null;
        Workbook workbook = null;

        try {
            // Open the Excel file
            fileInputStream = new FileInputStream(new File(excelFilePath));
            workbook = new XSSFWorkbook(fileInputStream);

            // Get the first sheet
            Sheet sheet = workbook.getSheetAt(0);

            // Iterate through each row in the sheet
            for (Row row : sheet) {
                // Iterate through each cell in the row
                for (Cell cell : row) {
                    // Print the cell value based on its type
                    switch (cell.getCellType()) {
                        case STRING:
                            System.out.print(cell.getStringCellValue() + "\t");
                            break;
                        case NUMERIC:
                            System.out.print(cell.getNumericCellValue() + "\t");
                            break;
                        case BOOLEAN:
                            System.out.print(cell.getBooleanCellValue() + "\t");
```

```
                    break;
                case FORMULA:
                    System.out.print(cell.getCellFormula() + "\t");
                    break;
                default:
                    System.out.print("Unknown Cell Type\t");
                    break;
            }
        }
        System.out.println(); // New line after each row
    }
  } catch (IOException e) {
    e.printStackTrace();
  } finally {
    // Close resources
    try {
      if (workbook != null) {
        workbook.close();
      }
      if (fileInputStream != null) {
        fileInputStream.close();
      }
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
 }
}
```

File : student_details.xlsx

Name	USN	Phone
Suresh	22CS001	954562123
John	22CS002	985421231
Smith	22CS003	956231205
Tom	22CS004	965621322

Output:

Name	USN	Phone
Suresh	22CS001	954562123
John	22CS002	985421231
Smith	22CS003	956231205
Tom	22CS004	965621322