

Name: Shubham Patgavkar

Div: C Batch: C3

Roll No: 233074

PRN No: 22320100

Practical No: 11

Title: Implement direct access file for any Database and perform following operations on it i) Create Database ii) Display Database iii) Search a record

Code:

```
import java.io.*;

class Record {
    private int id;
    private String name;

    public Record(int id, String name) {
        this.id = id;
        this.name = name;
    }

    public int getId() {
        return id;
    }

    public String getName() {
        return name;
    }

    @Override
    public String toString() {
        return "Record{id=" + id + ", name='" + name + "'}";
    }
}

public class Assignment11 {
    private static final String FILE_PATH = "database.txt";

    public static void main(String[] args) {
        try {
```

```

        // Prompt user for the file name to be created
        BufferedReader reader = new BufferedReader(new
InputStreamReader(System.in));
        System.out.print("Enter the file name to be created: ");
        String fileName = reader.readLine();
        String filePath = fileName + ".txt";

        // Create or open the database file
        File file = new File(filePath);
        if (!file.exists()) {
            file.createNewFile();
            System.out.println("New database created.");
        }

        // Display Database
        displayDatabase(file);

        // Menu
        menu(file, reader);

    } catch (IOException | NumberFormatException e) {
        e.printStackTrace();
    }
}

private static void displayDatabase(File file) {
    try (BufferedReader reader = new BufferedReader(new
FileReader(file))) {
        System.out.println("Current Database Records:");
        String line;
        int lineNumber = 1;
        while ((line = reader.readLine()) != null) {
            String[] parts = line.split(",");
            int id = Integer.parseInt(parts[0]);
            String name = parts[1];
            System.out.println("Record " + lineNumber++ + ":
ID=" + id + ", Name=" + name);
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
}

```

```

private static void menu(File file, BufferedReader reader) {
    while (true) {
        System.out.println("\nMenu:");
        System.out.println("1. Add Record");
        System.out.println("2. Search Record");
        System.out.println("3. Display Database");
        System.out.println("4. Exit");
        System.out.print("Enter your choice: ");
        try {
            int choice = Integer.parseInt(reader.readLine());

            switch (choice) {
                case 1:
                    addRecord(file, reader);
                    break;
                case 2:
                    searchRecord(file, reader);
                    break;
                case 3:
                    displayDatabase(file);
                    break;
                case 4:
                    return;
                default:
                    System.out.println("Invalid choice. Please
enter a valid option.");
            }
        } catch (IOException | NumberFormatException e) {
            e.printStackTrace();
        }
    }
}

private static void addRecord(File file, BufferedReader reader)
{
    try (FileWriter writer = new FileWriter(file, true)) {
        System.out.print("Enter ID for the new record: ");
        int id = Integer.parseInt(reader.readLine());
        System.out.print("Enter name for the new record: ");
        String name = reader.readLine();
    }
}

```

```

        writer.write(id + "," + name + "\n");

        System.out.println("Record added successfully.");
    } catch (IOException | NumberFormatException e) {
        e.printStackTrace();
    }
}

private static void searchRecord(File file, BufferedReader
reader) {
    try (BufferedReader fileReader = new BufferedReader(new
FileReader(file))) {
        System.out.print("Enter ID to search: ");
        int idToSearch = Integer.parseInt(reader.readLine());
        String line;
        int lineNumber = 1;
        boolean found = false;
        while ((line = fileReader.readLine()) != null) {
            String[] parts = line.split(",");
            int id = Integer.parseInt(parts[0]);
            String name = parts[1];
            if (id == idToSearch) {
                System.out.println("Record found: ID=" + id + ",
Name=" + name);
                found = true;
                break;
            }
            lineNumber++;
        }
        if (!found) {
            System.out.println("Record with ID " + idToSearch +
" not found.");
        }
    } catch (IOException | NumberFormatException e) {
        e.printStackTrace();
    }
}
}

```

Output:

Enter the file name to be created: fileno11

New database created.

Current Database Records:

Menu:

1. Add Record
2. Search Record
3. Display Database
4. Exit

Enter your choice: 1

Enter ID for the new record: 73

Enter name for the new record: vijay

Record added successfully.

Menu:

1. Add Record
2. Search Record
3. Display Database
4. Exit

Enter your choice: 3

Current Database Records:

Record 1: ID=73, Name=vijay

Menu:

1. Add Record

2. Search Record

3. Display Database

4. Exit

Enter your choice: 2

Enter ID to search: 73

Record found: ID=73, Name=vijay

Menu:

1. Add Record

2. Search Record

3. Display Database

4. Exit

Enter your choice: 4