Activity 4

B25CE1205 SARVESH MANDE CE3 C

A database is an organized collection of data that allows efficient storage, retrieval, and management. In computer systems, databases act as central repositories where multiple users or applications can access, update, and analyze information in a structured way.

In the case of a Cinema Booking System, a database stores all booking details such as customer names, movie titles, show timings, seat counts, ticket prices, and show dates. This helps cinema owners, multiplexes, and online booking platforms manage their records efficiently without relying on paper-based systems.

Maintaining a database ensures that data is accurate and consistent, allows quick retrieval of specific records, and supports large numbers of bookings. It also enables reporting and analysis, such as tracking popular movies, peak hours, and revenue. Real-world cinema management systems may use relational databases or simple file-based storage to maintain booking information, and they often use indexed structures to make searching and sorting faster.

This approach is similar to other management systems like e-commerce platforms, airline or train ticketing systems, and event registration systems. Overall, a well-maintained database makes cinema booking management efficient, error-free, and accessible, improving both operations and customer satisfaction.

References:
https://www.geeksforgeeks.org/dbms-introduction/
https://www.tutorialspoint.com/dbms/index.htm
https://www.studytonight.com/dbms/introduction-to-database-management-system

ANALYSIS:

The Cinema Booking System is a basic program written in C to manage movie ticket bookings in a structured way. The program stores details like customer name, movie name, show time, number of seats, ticket price, and show date. All this information is grouped together using a structure, which makes it easy to manage multiple bookings as a single unit.

The system allows users to input multiple bookings, and then it sorts these bookings by ticket price using a simple Bubble Sort algorithm. Sorting helps to view bookings from the cheapest to the most expensive, making it easier for cinema managers or customers to compare ticket prices quickly.

After sorting, the program displays all the booking details in a clear format, showing each customer's booking information one by one. The program uses arrays to store multiple bookings, loops to handle repeated tasks like input and display, and simple comparison logic for sorting.

Overall, this project demonstrates important programming concepts such as structures, arrays, loops, input/output, and sorting. It is a practical example of how basic programming knowledge can be used to create a useful application that can manage cinema bookings in an organized and efficient way.

Ideation:

The main idea of this program is to create a simple system for managing cinema bookings. It allows users to enter details like customer name, movie name, show time, number of seats, ticket price, and show date. The program stores these bookings, sorts them by ticket price, and displays all the information in an organized way. This helps cinema managers or customers easily view and compare bookings.

Algorithm

1.Start the program.
2.Ask the user how many cinema bookings to enter.

3.For each booking:
- Read customer name
- Read movie name
- Read show time
- Read number of seats
- Read ticket price
- Read show date
- Store all details in the structure

4.Sort all bookings in ascending order based on ticket price.

5.Display all the sorted bookings in a neatly formatted table.

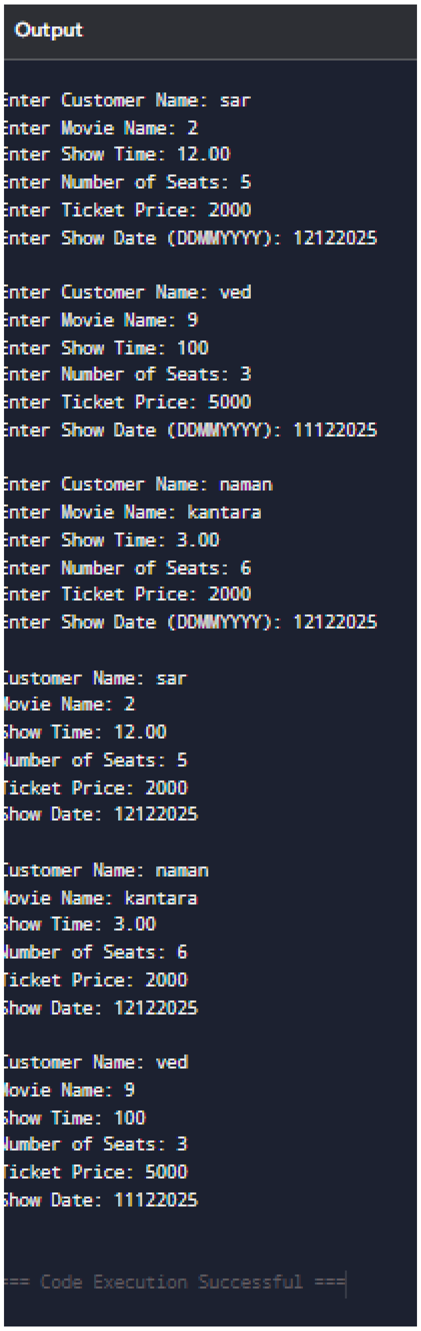
6.End the program.

Build:

```
#include <stdio.h>

#include <string.h>

struct Cinema {

    char customer_name[20];

    char movie_name[20];

    char show_time[15];

    int seats;

    int ticket_price;

    int date; // DDMMYYYY format

};

int main() {

    struct Cinema booking[3], temp; // Fixed size 3 bookings

    int i, j;
```

```
for (i = 0; i < 3; i++) {
    printf("\nEnter Customer Name: ");
    scanf("%s", booking[i].customer_name);
    printf("Enter Movie Name: ");
    scanf("%s", booking[i].movie_name);
    printf("Enter Show Time: ");
    scanf("%s", booking[i].show_time);
    printf("Enter Number of Seats: ");
    scanf("%d", &booking[i].seats);
    printf("Enter Ticket Price: ");
    scanf("%d", &booking[i].ticket_price);
    printf("Enter Show Date (DDMMYYYY): ");
    scanf("%d", &booking[i].date);
}
for (i = 0; i < 3 - 1; i++) {
    for (j = 0; j < 3 - i - 1; j++) {
        if (booking[j].ticket_price > booking[j + 1].ticket_price) {
            temp = booking[j];
            booking[j] = booking[j + 1];
            booking[j + 1] = temp;
        }
        }
    }
    for (i = 0; i < 3; i++) {
        printf("\nCustomer Name: %s", booking[i].customer_name);
        printf("\nMovie Name: %s", booking[i].movie_name);
        printf("\nShow Time: %s", booking[i].show_time);
        printf("\nNumber of Seats: %d", booking[i].seats);
        printf("\nTicket Price: %d", booking[i].ticket_price);
        printf("\nShow Date: %d\n", booking[i].date);
    }

   return 0;
}
```

test:



Output

```
Enter Customer Name: sar
Enter Movie Name: 2
Enter Show Time: 12.00
Enter Number of Seats: 5
Enter Ticket Price: 2000
Enter Show Date (DDMMYYYY): 12122025

Enter Customer Name: ved
Enter Movie Name: 9
Enter Show Time: 100
Enter Number of Seats: 3
Enter Ticket Price: 5000
Enter Show Date (DDMMYYYY): 11122025

Enter Customer Name: naman
Enter Movie Name: kantara
Enter Show Time: 3.00
Enter Number of Seats: 6
Enter Ticket Price: 2000
Enter Show Date (DDMMYYYY): 12122025

Customer Name: sar
Movie Name: 2
Show Time: 12.00
Number of Seats: 5
Ticket Price: 2000
Show Date: 12122025

Customer Name: naman
Movie Name: kantara
Show Time: 3.00
Number of Seats: 6
Ticket Price: 2000
Show Date: 12122025

Customer Name: ved
Movie Name: 9
Show Time: 100
Number of Seats: 3
Ticket Price: 5000
Show Date: 11122025


=== Code Execution Successful ===
```

implementation:https://github.com/sarvesh00719/cinema-database-manager.git