PAVITHARANI G P

2347244

DATA STRUCTURES USING C

LAB EXERCISE – 1

QUESTION :

Implement Matrix manipulation .
Consider the 2D representation for your chosen domain. Perform all data structure operations (insertion, Deletion, linear search) using 2D arrays for any chosen logical data of your domain. Implement any two matrix operations.

PROGRAM :

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

#define PROPS_ROWS 3
#define PROPS_COLS 3

struct propMatrix
{
    char prop_name[25];
    int stock[PROPS_ROWS][PROPS_COLS];
};

struct propMatrix propStore;

void propInsertionDisplay();
int propDeleteElement(int row, int col);
void displayProps();
int propLinearSearch(int value);
void addProps();
void multiplyProps();
int main()
{
    int choice, value, count = PROPS_ROWS * PROPS_COLS, row, col;
    int continueMenu = 1;

    while (continueMenu)
    {
        printf("----------------------Photography_Prop_Store-----------------
-------------\n");
        printf("Enter your Choice (1: Insert, 2: Delete 3: Display, 4:
Searching, 5: Add Props, 6: Multiply Props, 0: Exit): ");
        scanf("%d", &choice);
```

```
        switch (choice)
        {
        case 1:
            printf("Enter Prop Name :");
            scanf("%s", &propStore.prop_name);
            printf("Enter the stock count for the last %d days:\n", count);
            propInsertionDisplay();
            break;
        case 2:
            printf("Enter the Row and Column of the stock : ");
            scanf("%d %d", &row, &col);
            propDeleteElement(row - 1, col - 1);
            break;
        case 3:
            displayProps();
            break;
        case 4:
            printf("Enter the value to be searched :");
            scanf("%d", &value);
            propLinearSearch(value);
            break;
        case 5:
            printf("----------------PROP ADDITION----------------------");
            addProps();
            break;
        case 6:
            printf("----------------PROP MULTIPLICATION----------------------
");
            multiplyProps();
            break;
        case 0:
            continueMenu = 0;
            break;
        default:
            printf("Invalid choice. Please try again.\n");
        }
    }

    return 0;
}

void addProps()
{
    int i, j, sum[3][3];
    for (i = 0; i < PROPS_ROWS; ++i)
        for (j = 0; j < PROPS_COLS; ++j)
        {
            sum[i][j] = propStore.stock[i][j] + propStore.stock[i][j];
```

```
        }

    printf(" ");
    for (int i = 0; i < PROPS_ROWS; i++)
    {
        printf(" ");
        for (int j = 0; j < PROPS_COLS; j++)
        {
            printf("%d     ", sum[i][j]);
        }
        printf("\n");
    }
}

void multiplyProps()
{
    int i, j, mul[3][3];
    for (i = 0; i < PROPS_ROWS; ++i)
        for (j = 0; j < PROPS_COLS; ++j)
        {
            mul[i][j] = propStore.stock[i][j] * propStore.stock[i][j];
        }

    printf(" ");
    for (int i = 0; i < PROPS_ROWS; i++)
    {
        printf(" ");
        for (int j = 0; j < PROPS_COLS; j++)
        {
            printf("%d     ", mul[i][j]);
        }
        printf("\n");
    }
}

void displayProps()
{
    printf("----------------------3x3 Matrix of Prop Stock--------------------
-------\n");
    for (int i = 0; i < PROPS_ROWS; i++)
    {
        printf(" ");
        for (int j = 0; j < PROPS_COLS; j++)
        {
            printf("%d     ", propStore.stock[i][j]);
        }
        printf("\n");
    }
```

```
}

void propInsertionDisplay()
{
    for (int i = 0; i < PROPS_ROWS; i++)
    {
        for (int j = 0; j < PROPS_COLS; j++)
        {
            scanf("%d", &propStore.stock[i][j]);
        }
    }

    printf("----------------------Stock of %s--------------------------\n",
propStore.prop_name);

    for (int i = 0; i < PROPS_ROWS; i++)
    {
        printf(" ");
        for (int j = 0; j < PROPS_COLS; j++)
        {
            printf("%d     ", propStore.stock[i][j]);
        }
        printf("\n");
    }
}

int propDeleteElement(int row, int col)
{
    if (row >= 0 && row < PROPS_ROWS && col >= 0 && col < PROPS_COLS)
    {
        propStore.stock[row][col] = -1;
        return printf("Element deleted\n");
    }
    return printf("Element not found\n");
}

int propLinearSearch(int value)
{
    printf("---------------------------Searching-----------------------------
-----\n");

    for (int i = 0; i < PROPS_ROWS; i++)
    {
        for (int j = 0; j < PROPS_COLS; j++)
        {
            if (propStore.stock[i][j] == value)
            {
                return printf("value %d is found\n", value);
```

```
            }
        }
    }
    return printf("value is not found\n");
}
```

OUTPUT :