# DATA STRUCTURES

| | |
|---|---|
| **Division** | **CS(AIML) -A** |
| **Batch** | **2** |
| **GR-no** | **12311493** |
| **Roll no** | **54** |
| **Name** | **Atharva Kangralkar** |

## Assignment 1:

## WAP to implement Linear Search and Binary search.

# Code:-

```
#include <stdio.h>

// Function for Linear Search in 1D Array
int LS_1D(int arr[], int size, int key) {
    for (int i = 0; i < size; i++) {
        if (arr[i] == key) {
            return i; // Found, return index
        }
    }
    return -1; // Not found
}

// Function for Linear Search in 2D Array
int LS_2D(int arr[][100], int rows, int cols, int key, int *row, int *col) {
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            if (arr[i][j] == key) {
                *row = i;
                *col = j;
                return 1; // Found
            }
        }
    }
    return 0; // Not found
}

// Function for Binary Search in 1D Array (Requires Sorted Array)
int BS_1D(int arr[], int size, int key) {
```

```
    int left = 0, right = size - 1;
    while (left <= right) {
        int mid = left + (right - left) / 2;
        if (arr[mid] == key) {
            return mid; // Found
        } else if (arr[mid] < key) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }
    return -1; // Not found
}

// Function for Binary Search in 2D Sorted Array
int BS_2D(int arr[][100], int rows, int cols, int key, int *row, int *col) {
    int left = 0, right = rows * cols - 1;
    while (left <= right) {
        int mid = left + (right - left) / 2;
        int i = mid / cols, j = mid % cols;
        if (arr[i][j] == key) {
            *row = i;
            *col = j;
            return 1; // Found
        } else if (arr[i][j] < key) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
```

```
    }
    return 0; // Not found
}

int main() {
    int choice;
    do {
        printf("\nMenu:\n");
        printf("1. Linear Search in 1D Array\n");
        printf("2. Linear Search in 2D Array\n");
        printf("3. Binary Search in 1D Array (Sorted)\n");
        printf("4. Binary Search in 2D Sorted Array\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1: {
                int n, key;
                printf("Enter size of 1D array: ");
                scanf("%d", &n);
                int arr[n];
                printf("Enter elements: ");
                for (int i = 0; i < n; i++) scanf("%d", &arr[i]);
                printf("Enter key to search: ");
                scanf("%d", &key);
                int index = LS_1D(arr, n, key);
                if (index != -1)
                    printf("Element found at index %d\n", index);
```

```
    else
        printf("Element not found!\n");
    break;
}
case 2: {
    int rows, cols, key, row, col;
    printf("Enter rows and columns of 2D array: ");
    scanf("%d %d", &rows, &cols);
    int arr[100][100];
    printf("Enter elements:\n");
    for (int i = 0; i < rows; i++)
        for (int j = 0; j < cols; j++)
            scanf("%d", &arr[i][j]);
    printf("Enter key to search: ");
    scanf("%d", &key);
    if (LS_2D(arr, rows, cols, key, &row, &col))
        printf("Element found at position (%d, %d)\n", row, col);
    else
        printf("Element not found!\n");
    break;
}
case 3: {
    int n, key;
    printf("Enter size of sorted 1D array: ");
    scanf("%d", &n);
    int arr[n];
    printf("Enter elements (sorted order): ");
    for (int i = 0; i < n; i++) scanf("%d", &arr[i]);
    printf("Enter key to search: ");
```

```
        scanf("%d", &key);
        int index = BS_1D(arr, n, key);
        if (index != -1)
            printf("Element found at index %d\n", index);
        else
            printf("Element not found!\n");
        break;
    }
    case 4: {
        int rows, cols, key, row, col;
        printf("Enter rows and columns of sorted 2D array: ");
        scanf("%d %d", &rows, &cols);
        int arr[100][100];
        printf("Enter elements (sorted row-wise and column-wise):\n");
        for (int i = 0; i < rows; i++)
            for (int j = 0; j < cols; j++)
                scanf("%d", &arr[i][j]);
        printf("Enter key to search: ");
        scanf("%d", &key);
        if (BS_2D(arr, rows, cols, key, &row, &col))
            printf("Element found at position (%d, %d)\n", row, col);
        else
            printf("Element not found!\n");
        break;
    }
    case 5:
        printf("Exiting program...\n");
        break;
    default:
```

```
            printf("Invalid choice! Try again.\n");
        }
    } while (choice != 5);

    return 0;
}
```

# Code Screenshot:-

```c
#include <stdio.h>

// Function for Linear Search in 1D Array
Tabnine | Edit | Test | Explain | Document
int LS_1D(int arr[], int size, int key) {
    for (int i = 0; i < size; i++) {
        if (arr[i] == key) {
            return i; // Found, return index
        }
    }
    return -1; // Not found
}

// Function for Linear Search in 2D Array
Tabnine | Edit | Test | Explain | Document
int LS_2D(int arr[][100], int rows, int cols, int key, int *row, int *col) {
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            if (arr[i][j] == key) {
                *row = i;
                *col = j;
                return 1; // Found
            }
        }
    }
    return 0; // Not found
}

// Function for Binary Search in 1D Array (Requires Sorted Array)
Tabnine | Edit | Test | Explain | Document
int BS_1D(int arr[], int size, int key) {
    int left = 0, right = size - 1;
    while (left <= right) {
        int mid = left + (right - left) / 2;
        if (arr[mid] == key) {
            return mid; // Found
        } else if (arr[mid] < key) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }
    return -1; // Not found
}

// Function for Binary Search in 2D Sorted Array
Tabnine | Edit | Test | Explain | Document
int BS_2D(int arr[][100], int rows, int cols, int key, int *row, int *col) {
    int left = 0, right = rows * cols - 1;
    while (left <= right) {
        int mid = left + (right - left) / 2;
        int i = mid / cols, j = mid % cols;
        if (arr[i][j] == key) {
            *row = i;
            *col = j;
            return 1; // Found
        } else if (arr[i][j] < key) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }
    return 0; // Not found
}

Tabnine | Edit | Test | Explain | Document
int main() {
    int choice;
    do {
```

```c
        printf("\nMenu:\n");
        printf("1. Linear Search in 1D Array\n");
        printf("2. Linear Search in 2D Array\n");
        printf("3. Binary Search in 1D Array (Sorted)\n");
        printf("4. Binary Search in 2D Sorted Array\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1: {
                int n, key;
                printf("Enter size of 1D array: ");
                scanf("%d", &n);
                int arr[n];
                printf("Enter elements: ");
                for (int i = 0; i < n; i++) scanf("%d", &arr[i]);
                printf("Enter key to search: ");
                scanf("%d", &key);
                int index = LS_1D(arr, n, key);
                if (index != -1)
                    printf("Element found at index %d\n", index);
                else
                    printf("Element not found!\n");
                break;
            }
            case 2: {
                int rows, cols, key, row, col;
                printf("Enter rows and columns of 2D array: ");
                scanf("%d %d", &rows, &cols);
                int arr[100][100];
                printf("Enter elements:\n");
                for (int i = 0; i < rows; i++)
                    for (int j = 0; j < cols; j++)
                        scanf("%d", &arr[i][j]);
                printf("Enter key to search: ");
                scanf("%d", &key);
                if (LS_2D(arr, rows, cols, key, &row, &col))
                    printf("Element found at position (%d, %d)\n", row, col);
                else
                    printf("Element not found!\n");
                break;
            }
            case 3: {
                int n, key;
                printf("Enter size of sorted 1D array: ");
                scanf("%d", &n);
                int arr[n];
                printf("Enter elements (sorted order): ");
                for (int i = 0; i < n; i++) scanf("%d", &arr[i]);
                printf("Enter key to search: ");
                scanf("%d", &key);
                int index = BS_1D(arr, n, key);
                if (index != -1)
                    printf("Element found at index %d\n", index);
                else
                    printf("Element not found!\n");
                break;
            }
            case 4: {
                int rows, cols, key, row, col;
                printf("Enter rows and columns of sorted 2D array: ");
                scanf("%d %d", &rows, &cols);
                int arr[100][100];
                printf("Enter elements (sorted row-wise and column-wise):\n");
                for (int i = 0; i < rows; i++)
```

```
                    for (int j = 0; j < cols; j++)
                        scanf("%d", &arr[i][j]);
                printf("Enter key to search: ");
                scanf("%d", &key);
                if (BS_2D(arr, rows, cols, key, &row, &col))
                    printf("Element found at position (%d, %d)\n", row, col);
                else
                    printf("Element not found!\n");
                break;
            }
            case 5:
                printf("Exiting program...\n");
                break;
            default:
                printf("Invalid choice! Try again.\n");
        }
    } while (choice != 5);

    return 0;
}

```

# Output:-

## 1)Linear Search in 1-D array:-

```
Menu:
1. Linear Search in 1D Array
2. Linear Search in 2D Array
3. Binary Search in 1D Sorted Array
4. Binary Search in 2D Sorted Array
5. Exit
Enter your choice: 1
Enter size of 1D array: 5
Enter elements: 1
2
3
4
5
Enter key to search: 5
Element found at index 4
```

```
Menu:
1. Linear Search in 1D Array
2. Linear Search in 2D Array
3. Binary Search in 1D Sorted Array
4. Binary Search in 2D Sorted Array
5. Exit
Enter your choice: 1
Enter size of 1D array: 5
Enter elements: 1
2
3
4
5
Enter key to search: 6
Element not found!
```

## 2)Linear search in 2-D array:-

```
Menu:
1. Linear Search in 1D Array
2. Linear Search in 2D Array
3. Binary Search in 1D Sorted Array
4. Binary Search in 2D Sorted Array
5. Exit
Enter your choice: 2
Enter rows and columns of 2D array: 2
2
Enter elements:
1
2
3
4
Enter key to search: 3
Element found at position (1, 0)
```

```
Menu:
1. Linear Search in 1D Array
2. Linear Search in 2D Array
3. Binary Search in 1D Sorted Array
4. Binary Search in 2D Sorted Array
5. Exit
Enter your choice: 2
Enter rows and columns of 2D array: 2
2
Enter elements:
1
2
3
4
Enter key to search: 5
Element not found!
```

**3) Binary Search in 1-D Sorted Array:-**

```
Menu:
1. Linear Search in 1D Array
2. Linear Search in 2D Array
3. Binary Search in 1D Sorted Array
4. Binary Search in 2D Sorted Array
5. Exit
Enter your choice: 3
Enter size of sorted 1D array: 5
Enter elements (sorted order): 1
2
3
4
5
Enter key to search: 4
Element found at index 3
```

```
Menu:
1. Linear Search in 1D Array
2. Linear Search in 2D Array
3. Binary Search in 1D Sorted Array
4. Binary Search in 2D Sorted Array
5. Exit
Enter your choice: 3
Enter size of sorted 1D array: 5
Enter elements (sorted order): 1
2
3
4
5
Enter key to search: 8
Element not found!
```

**4) Binary Search in 2-D Sorted Array:-**

```
Menu:
1. Linear Search in 1D Array
2. Linear Search in 2D Array
3. Binary Search in 1D Sorted Array
4. Binary Search in 2D Sorted Array
5. Exit
Enter your choice: 4
Enter rows and columns of sorted 2D array: 2
2
Enter elements (sorted row-wise and column-wise):
1
2
3
4
Enter key to search: 3
Element found at position (1, 0)
```

```
Menu:
1. Linear Search in 1D Array
2. Linear Search in 2D Array
3. Binary Search in 1D Sorted Array
4. Binary Search in 2D Sorted Array
5. Exit
Enter your choice: 4
Enter rows and columns of sorted 2D array: 2
3
Enter elements (sorted row-wise and column-wise):
1
2
3
4
5
6
Enter key to search: 9
Element not found!
```

**5) Invalid Choice:-**

```
Menu:
1. Linear Search in 1D Array
2. Linear Search in 2D Array
3. Binary Search in 1D Sorted Array
4. Binary Search in 2D Sorted Array
5. Exit
Enter your choice: 6
Invalid choice! Try again.
```