

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);
            }
        }
    } while (choice != 5);
}

```

```

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:-

```
1  #include <stdio.h>
2
3  // Function for Linear Search in 1D Array
4  int LS_1D(int arr[], int size, int key) {
5      for (int i = 0; i < size; i++) {
6          if (arr[i] == key) {
7              return i; // Found, return index
8          }
9      }
10     return -1; // Not found
11 }
12
13 // Function for Linear Search in 2D Array
14 int LS_2D(int arr[][100], int rows, int cols, int key, int *row, int *col) {
15     for (int i = 0; i < rows; i++) {
16         for (int j = 0; j < cols; j++) {
17             if (arr[i][j] == key) {
18                 *row = i;
19                 *col = j;
20                 return 1; // Found
21             }
22         }
23     }
24     return 0; // Not found
25 }
26
27 // Function for Binary Search in 1D Array (Requires Sorted Array)
28 int BS_1D(int arr[], int size, int key) {
29     int left = 0, right = size - 1;
30     while (left <= right) {
31         int mid = left + (right - left) / 2;
32         if (arr[mid] == key) {
33             return mid; // Found
34         } else if (arr[mid] < key) {
35             left = mid + 1;
36         } else {
37             right = mid - 1;
38         }
39     }
40     return -1; // Not found
41 }
42
43 // Function for Binary Search in 2D Sorted Array
44 int BS_2D(int arr[][100], int rows, int cols, int key, int *row, int *col) {
45     int left = 0, right = rows * cols - 1;
46     while (left <= right) {
47         int mid = left + (right - left) / 2;
48         int i = mid / cols, j = mid % cols;
49         if (arr[i][j] == key) {
50             *row = i;
51             *col = j;
52             return 1; // Found
53         } else if (arr[i][j] < key) {
54             left = mid + 1;
55         } else {
56             right = mid - 1;
57         }
58     }
59     return 0; // Not found
60 }
61
62 int main() {
63     int choice;
64     do {
```



```

65     printf("\nMenu:\n");
66     printf("1. Linear Search in 1D Array\n");
67     printf("2. Linear Search in 2D Array\n");
68     printf("3. Binary Search in 1D Array (Sorted)\n");
69     printf("4. Binary Search in 2D Sorted Array\n");
70     printf("5. Exit\n");
71     printf("Enter your choice: ");
72     scanf("%d", &choice);
73
74     switch (choice) {
75     case 1: {
76         int n, key;
77         printf("Enter size of 1D array: ");
78         scanf("%d", &n);
79         int arr[n];
80         printf("Enter elements: ");
81         for (int i = 0; i < n; i++) scanf("%d", &arr[i]);
82         printf("Enter key to search: ");
83         scanf("%d", &key);
84         int index = LS_1D(arr, n, key);
85         if (index != -1)
86             printf("Element found at index %d\n", index);
87         else
88             printf("Element not found!\n");
89         break;
90     }
91     case 2: {
92         int rows, cols, key, row, col;
93         printf("Enter rows and columns of 2D array: ");
94         scanf("%d %d", &rows, &cols);
95         int arr[100][100];
96         printf("Enter elements:\n");
97         for (int i = 0; i < rows; i++)
98             for (int j = 0; j < cols; j++)
99                 scanf("%d", &arr[i][j]);
100        printf("Enter key to search: ");
101        scanf("%d", &key);
102        if (LS_2D(arr, rows, cols, key, &row, &col))
103            printf("Element found at position (%d, %d)\n", row, col);
104        else
105            printf("Element not found!\n");
106        break;
107    }
108    case 3: {
109        int n, key;
110        printf("Enter size of sorted 1D array: ");
111        scanf("%d", &n);
112        int arr[n];
113        printf("Enter elements (sorted order): ");
114        for (int i = 0; i < n; i++) scanf("%d", &arr[i]);
115        printf("Enter key to search: ");
116        scanf("%d", &key);
117        int index = BS_1D(arr, n, key);
118        if (index != -1)
119            printf("Element found at index %d\n", index);
120        else
121            printf("Element not found!\n");
122        break;
123    }
124    case 4: {
125        int rows, cols, key, row, col;
126        printf("Enter rows and columns of sorted 2D array: ");
127        scanf("%d %d", &rows, &cols);
128        int arr[100][100];
129        printf("Enter elements (sorted row-wise and column-wise):\n");
130        for (int i = 0; i < rows; i++)

```

```

131         for (int j = 0; j < cols; j++)
132             scanf("%d", &arr[i][j]);
133         printf("Enter key to search: ");
134         scanf("%d", &key);
135         if (BS_2D(arr, rows, cols, key, &row, &col))
136             printf("Element found at position (%d, %d)\n", row, col);
137         else
138             printf("Element not found!\n");
139         break;
140     }
141     case 5:
142         printf("Exiting program...\n");
143         break;
144     default:
145         printf("Invalid choice! Try again.\n");
146     }
147 } while (choice != 5);
148
149 return 0;
150 }
151

```

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.