UNIT – III

SEARCHING, SORTING AND HASHING TECHNIQUES

SEARCHING

- Searching means to find whether a particular value is present in an array or not
- If the value is present in an array, the searching is said to be successful and the searching process gives the location of that value in the array

TYPES OF SEARCHING

- LINEAR SEARCH
- BINARY SEARCH

LINEAR SEARCH

- Also called **Sequential search**
- Very simple method of searching
- Comparing the value with every element of the array one by one in a sequence until a match found
- Used for unordered list of elements

Ex:

int a[10] = { 10,5,8,2,4,9,1,3,7,6}

Key = 4	Key = 44
Pos = 5	Element not found

LINEAR SEARCH

LINEAR_SEARCH(A, N, VAL)

```
Step 1: [INITIALIZE] SET POS = -1
Step 2: [INITIALIZE] SET I = 0
Step 3:  Repeat Step 4 while I<N
Step 4:       IF A[I] = VAL, then
                   SET POS = I
                   PRINT POS + 1
                   Go to Step 6
          [END OF IF]
     [END OF LOOP]
Step 5: PRINT "Value Not Present In The Array"
Step 6: EXIT
```

BINARY SEARCH

- Work efficiently with sorted array
- Telephone Directory & Dictionary – Analogy

BEG = lower_bound and END = upper_bound

MID = (BEG + END) / 2

If VAL < A[MID], then VAL will be present in the left segment of the array. So,

the value of END will be changed as, END = MID – 1

If VAL > A[MID], then VAL will be present in the right segment of the array. So,

the value of BEG will be changed as, BEG = MID + 1

BINARY SEARCH

```
BINARY_SEARCH(A, lower_bound, upper_bound, VAL)

Step 1: SET BEG = lower_bound, END = upper_bound
Step 2: Repeat Step 3 and Step 4 while BEG <= END
Step 3:         SET MID = (BEG + END)/2
Step 4:         IF A[MID] = VAL, then
                        PRINT MID + 1
                        Go to Step 6
                ELSE IF A[MID] > VAL then;
                        SET END = MID - 1
                ELSE
                        SET BEG = MID + 1                      [END OF IF]
        [END OF LOOP]
Step 5: PRINT "VAL IS NOT PRESENT IN THE ARRAY"
Step 6: EXIT
```

BINARY SEARCH EXAMPLE

int A[11] = {0, 12, 24, 3, 44, 15, 67, 97, 28, 9, 10};
After sort : int A[11] = {0, 3, 9, 10, 12, 15, 24, 28, 44, 67, 97};

and VAL = 67, the algorithm will proceed in the following manner.

BEG = 0, END = 10, MID = (0 + 10)/2 = 5

Now, VAL = 67 and A[MID] = A[5] = 15

A[5] is less than VAL, therefore, we will now search for the value in the later half of the array. So, we change the values of BEG and MID.

Now, BEG = MID + 1 = 6, END = 10, MID = (6 + 10)/2 =16/2 = 8

Now, VAL = 67 and A[MID] = A[8] = 44

BINARY SEARCH EXAMPLE

A[8] is less than VAL, therefore, we will now search for the value in the later half of the array. So, again we change the values of BEG and MID.

Now, BEG = MID + 1 = 9, END = 10, MID = (9 + 10)/2 = 9

Now VAL = 67 and A[MID] = A[9] = 67

Therefore, the element found in the array of position 10