

SEARCH ALGORITHM

# INTERPOLATION SEARCH

ALPHA MEWING SIGMA

# INTERPOLATION SEARCH

- This is an improvement over binary search for instances, where the values in a sorted array are uniformly distributed.
- Binary Search always goes to the middle element to check. On the other hand, interpolation search may go to different locations according to the value of the key being searched.
- For example, if the value of the key is closer to the last element, interpolation search is likely to start search toward the end side.

# FORMULA

- $$pos = lo + \frac{(x - arr[lo]) \cdot (hi - lo)}{arr[hi] - arr[lo]}$$
- $arr[]$  ==> *Array where elements need to be searched*
- $x$  ==> *Element to be searched*
- $lo$  ==> *Starting index in  $arr[]$*

# TIME COMPLEXITY

- For Average Case :

$$O(\log_2(\log_2 n))$$

- For Worst Case :

$$O(n)$$

- Auxiliary Space Complexity :

$$O(1)$$

# EXAMPLE

10	13	15	26	28	30	50	56	58
----	----	----	----	----	----	----	----	----

- **Step1:** In a loop, calculate the value of "pos" using the probe position formula.
- **Step2:** If it is a match, return the index of the item, and exit.
- **Step3:** If the item is less than arr[pos], calculate the probe position of the left sub-array. Otherwise, calculate the same in the right sub-array.
- **Step4:** Repeat until a match is found or the sub-array reduces to zero.

# CODE

```
public static int interpolationSearch(int arr[], int lo,
                                    int hi, int x)
{
    int pos;

    // Since array is sorted, an element
    // present in array must be in range
    // defined by corner
    if (lo <= hi && x >= arr[lo] && x <= arr[hi]) {

        // Probing the position with keeping
        // uniform distribution in mind.
        pos = lo
            + (((hi - lo) / (arr[hi] - arr[lo]))
              * (x - arr[lo]));

        // Condition of target found
        if (arr[pos] == x)
            return pos;

        // If x is larger, x is in right sub array
        if (arr[pos] < x)
            return interpolationSearch(arr, pos + 1, hi,
                                     x);

        // If x is smaller, x is in left sub array
        if (arr[pos] > x)
            return interpolationSearch(arr, lo, pos - 1,
                                     x);
    }
    return -1;
}
```

```
public static void main(String[] args)
{

    // Array of items on which search will
    // be conducted.
    int arr[] = { 10, 12, 13, 16, 18, 19, 20, 21,
                  22, 23, 24, 33, 35, 42, 47 };

    int n = arr.length;

    // Element to be searched
    int x = 18;
    int index = interpolationSearch(arr, 0, n - 1, x);

    // If element was found
    if (index != -1)
        System.out.println("Element found at index "
                           + index);
    else
        System.out.println("Element not found.");
}
```

Element found at index 4

# OUR TEAM



**Muhammad Andhika**

NIM. 20250040092



**Regith Rayabi**

NIM.20250040075



**Syifa Nurul Fadilah**

NIM.20250040087



**M. Nizar Wirapradana**

NIM.20250040070

**THANK YOU !**





**ANY QUESTIONS????**

