DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
Discover. Learn. Empower.

# Experiment 1.1

**Student Name:** Gaurav Monga
**UID:** 23BAI70156
**Branch:** BE-AIT-CSE
**Section/Group:** 23AML-1 (A)
**Semester:** 4th
**Date of Performance:** 8 January, 2025
**Subject Name:** Design & Analysis of Algorithm
**Subject Code:** 23CSH-282

1. **Experiment Name:**
   Binary Search

2. **Objective:**
   To implement the Binary Search using the Divide & Conquer Approach.

3. **Theory:**
   Binary search is a highly efficient algorithm used for finding an element in a sorted array or list by repeatedly dividing the search interval in half. It's working includes:
   - Begin with the middle element of the array.
   - If the middle element is equal to the target value, the search is complete.
   - If the target value is less than the middle element, repeat the search on the left sub-array.
   - If the target value is greater, repeat the search on the right sub-array.

   The array or list must be a sorted array prior to performing a binary search; otherwise, the algorithm will not function correctly

   **Time Complexity:** O(log n)
   **Space Complexity:** O(log n) for the recursive approach and O(1) for the iterative approach.

4. **Algorithm:**

1. Start
2. Initialize low to 0 and high to n-1(n being the number of elements in the array)
3. Calculate mid as the integer division of (low + high) / 2.
4. Check the value of mid
   - If the element at mid is equal to the target, return mid.
   - If the element at mid is greater than the target, update high to mid-1.
   - If the element at mid is smaller than the target, update low to mid+1.
6. Repeat steps 3 and 4 until target is found or low becomes equal to high.
7. Return -1 if the target value is not found in the array.
8. End

5. **Code:**

```
#include<iostream>
using namespace std;
int main()
{
   int a[50],n,low,high,x, i, t, mid, f=0;
   cout << "How many elements do you want in the array\n";
   cin>>n;
   cout << "Enter the elements\n";
   for(i=0;i<n;i++)
   {
      cin>>a[i];
   }
   for(int i=0;i<n-1;i++)
   {
      for(int j=0;j<n-i-1;j++)
      {
         if(a[j+1]<a[j])
         {
            t=a[j+1];
            a[j+1]=a[j];
            a[j]=t;
         }
      }
   }
```

```
    cout << "Sorted array:\n";
    for(i=0;i<n;i++)
    cout << a[i] << "\t";
    cout << "\nEnter the number you want to search: \n";
    cin>>x;
    high=n;
    low=1;
    mid=((high+low)/2);
    while(low <= high && a[i] != x)
    {
       if(x<a[mid])
       high=mid-1;
       else if(x>a[mid])
       low=mid+1;
       mid=((high+low)/2);
       if(x == a[mid])
       {
          i=mid;
          cout << "Location of " << x << " is: " << i+1;
          f=1;
          break;
       }
    }
    if(f == 0)
    cout << "\n Search unsuccessful ";
    return 0;
}
```

6. **Output:**

```
How many elements do you want in the array
5
Enter the elements
5 32 12 34 11
Sorted array:
5       11      12      32      34
Enter the number you want to search:
32
Location of 32 is: 4
```

```
How many elements do you want in the array
4
Enter the elements
1 2 3 4
Sorted array:
1       2       3       4
Enter the number you want to search:
5

 Search unsuccessful
```

7. **Learning Outcomes:**
    - Learned about the implementation of divide and conquer technique.
    - Learned about the practical application of divide and conquer technique.
    - Learned about different searching algorithms.
    - Learned about binary search implementation and its efficiency.