**Practical-5**

**Implement a function of binary search and count the steps executed by function on various inputs for best case and worst case. Also write complexity in each case and draw a comparative chart.**

**Code :**

#include <iostream>

using namespace std;

void bs(int \*arr,int f,int l,int key)

{

if(l<f)

{

printf("element not found");

return;

}

int mid=f+(l-f)/2;

if(key==arr[mid])

{

printf("element %d found at index: %d ",key,mid);

return;

}

else if(key<arr[mid])

{

l=mid-1;

mid=f+(l-f)/2;

bs(arr,f,l,key);

}

else if(key>arr[mid])

{

f=mid+1;

mid=f+(l-f)/2;

bs(arr,f,l,key);

}

}

int main()

{

int arr[]={5,13,23,25,26,27,31,33,35,39,46,48,49,53,54,56,57,62,68,72,79,80,83,84,88,90,92,94,95,96,98,100};

int size=sizeof(arr)/sizeof(arr[0]);

bs(arr,0,size-1,48);

return 0;

}

**Output :**

**Best Case :**

|  |  |
| --- | --- |
| No of Elements | Time |
| 10 | 0.007312 |
| 20 | 0.007324 |
| 30 | 0.007465 |
| 40 | 0.007599 |
| 50 | 0.007897 |

**Worst case :**

|  |  |
| --- | --- |
| No of Elements | Time |
| 10 | 0.008123 |
| 20 | 0.008720 |
| 30 | 0.009786 |
| 40 | 0.010457 |
| 50 | 0.010598 |

**Best Case Vs Worst Case :**

|  |  |  |
| --- | --- | --- |
| No of Elements | **Best Case** | **Worst Case** |
| **10** | 0.007312 | 0.008123 |
| **20** | 0.007324 | 0.008720 |
| **30** | 0.007465 | 0.009786 |
| **40** | 0.007599 | 0.010457 |
| **50** | 0.007897 | 0.010598 |

**Conclusion:**

For Binary search best case will be when key element(element to be searched) is first element of the array and time complexity will be O(1)

And worst case will be key element is last element or not present in array in that case time complexity will be O(logn)