Output:

arr[5] = {1, 2, 4, 10, 40}

Case (i) → Key = 4.

Element is present in the array at Index : 02.

Case (ii) → Key = 18

Element is Not present in the array.

Date 11/01/24 Expt. No. 01

Expt. Name Page No. 1

AIM: Implementation of Binary Search using the divide and conquer approach.

Algorithm:

01. Start the binary Search Function with Parameter
- arr : the array to be Searched.
- low : the lowest index of the array.
- High : the highest index of the array.
- Key : the element to be Searched.

02. Check if the Higher index is greater than or equal to the low index.
a. if true, continue to the next step.
b. if false, return -1 to Indicate that the element is not found.

03. Calculate the Mid index using the formula:

Mid = low + (High-low)/2 ;

04. Check if the element at the Mid index of the array is Equal to the Key:
a. If true, return, the Mid index.
b. If false, continue, to the next Step.

05. Check if the Element at the Mid index is greater than the key: a. if true, Recursively

Teacher's Signature:

Date _____ Expt. No. _____

Expt. Name _____ Page No. 2.

Call the binary Search Function with Parameters.
- arr : the array.
- low : the lowest index.
- Mid : -1 the new highest index.
- key : the Element to be stored.

b. If false, recursively call the Binary Search function with parameters :-
- arr : the array
- Mid + 1 : the new lowest index.
- High : the highest index.
- Key : the Element to be Searched.

6. Return the result of the recursive calls.
- In the Main Function() :-
  - Declare the variables, array, cal the size of array by using n = Size of (arr) / Size of (arr[0]).
- Call the Binary Search function with Parameters
- Check if the result is -1 ;
  - if true, print "Element is Not Present in the array".
  - if false, Print "Element is not Present in the array".

SOURCE CODE :

```
#include <stdio.h>
int BinarySearch (int arr[], int low, int High, int
                  Key ) {
    if (High >= low) {
```

Teacher's Signature: _____

Date _____ Expt. No. _____

Expt. Name _____ Page No. 3

```
    int Mid = low + (High - low)/2;
      if (arr[mid] == Key)
       return Mid;

      if (arr[Mid] > Key) {
       return BinarySearch(arr, low, mid-1, key);
       return BinarySearch(arr, Mid+1, High, Key);
    }
      return -1;
}
    int main() {
    int arr[] = {1,2,4,10,40};
    int n = sizeof(arr)/sizeof(arr[0]);
     int key = 4;
    int result = binarySearch(arr, 0, n-1, key);
     if (result == -1)
         printf("Element is not present in the array");
    else {
      printf("Element is Present in the array at
            index %d " result);
     return 0;
    }
```

TIME COMPLEXITY :

O(log n) → Average Case.
O(1) → Best Case.
O(n) → Worst Case.

Teacher's Signature: _____