Documentation of Q4 (Explanations are written with the abbreviation "Exp:")

```
#include <stdio.h>
//#define MAX 50
```

Exp: Libraries are called and a MAX is defined that is the maximum inputs the array can take.

```
int Search(int *ar, int start, int end, int element)
{
   while (start <= end)
   {
      int middle = start + (end- start )/2;
      if (ar[middle] == element)
         return middle;
      if (ar[middle] < element)
         start = middle + 1;
      else
         end = middle - 1;
   }
   return -1;
}

int main(){
/* sorting algorithm is used to sort the elements used in the
following search algorithm*/
   int element,i,j,n,found,t,next,m,k;
   //float ar[j+1];
   int ar[k+1];
   n=10;
                                          // max elements the
program take is 10
   for(k=0;k<n;k++)
                                          // for loop to take
elements from the user
   {
               printf("Enter a number: \n");
               scanf("%d",&ar[k]);
   }

   printf("Enter the element to be found:\n");
        // element to be found is entered
   scanf("%d",&element);
   for(i=0; i<n-1;i+
+)                                                  /* Nested loop to
compare the ith element of the list to all others (from i+1)     */
        {
               m= i;
```

```
                for(j=i+1;j<n;j++)
                {
                        if(ar[m] > ar[j])
                                m=j;
                }
                if(m!=i)
                {
                        next= ar[i];
                        ar[i]= ar[m];
                        ar[m]= next;
                }
                /*printf("The %d intermediate list is: ",i+1);
        // prints the intermediate sorted lists
                for(t=0;t<n;t++)
                {
                        printf("%f",ar[t]);
                }
                printf("\n");
                */
        }
        printf("The sorted list is :");
                        // prints the sorted list
        for(i=0;i<n;i++)
        {
                printf("%d",ar[i]);
        }
        printf("\n");
   /* binary search algorithm is used */
   found = Search(ar, 0, n-1, element);
   if(found == -1 ) {
                                            // if the found is -1
that means while loop was not executes in search function
      printf("Element not found in the array \n");
   }
   else {
      printf("Element found at index : %d\n",found);
   }
   return 0;
}
```

Exp: A search function is declared that carries the arguments--
array,start(or the left most element of the array),end(or the right most element of the array)
and element(which is some random element of the array taken from the user).
Since we are supposed to find an element in O(log2n)time complexity we are using a sorted array and applying a binary search algorithm on it.
First we find out the index of the middle element using the formula

(start+(end-start))/2 and is stored in the variable middle.
Now we check if the middle element (ar[middle]) is equal to the element we are looking for, if not then we check if it is less than the middle element (lying to the left) or more than the element (lying to the right).This increases the efficiency of the search process
as we will only be searching in only one part of the array.
Once we find out the part of the array where the element is located we shift the start to the (middle+1) index if the element is found on the right
else if the element if on the left,start will stay on it's initial position and the end will shift to (middle-1).
The alg is repeated until element to be found is equal to middle element and then it moves out of the while loop.
Value of the middle is stored in variable "found" and printed when "found" is not equal to -1. Since While condition states that start is always less than end in sorted list, when the orientation reverses the loop ends and found is printed to be -1.