ASSIGNMENT-6

Searching and Sorting.

1) Take the elements from the user and sort them in descending order and do the following.

a) Using Binary Search find the element and the location in the array where the element is asked from user.

b) Ask the user to enter any two locations print the sum and product & values at these locations in the sorted array.

Ans:-

```
# include <stdio.h>
void sort (int a[ ], int n)
{
    int i, j, temp;
     For(i=0 ; i<n ; i++) {

         for (j=i+1 ; j<n ; j++){
            if (a[i] <a[j]){

               temp = a[i]
               a[i] = a[j]
               a[j] = temp;

            }
         }
  }
```

64MP AI QUAD CAMERA
Shot on realme XT
2020/05/05 20:50

Scanned with CamScanner

```
int binary (int a[] ,int b, int n)
{
    int i=0, j=n-1 , mid ;
    while (i<=j) {
          mid = (i+j)/2 ;
          if (a[mid] == b)
              return mid +1;
          else {
             if (b <a[mid])
                 j =mid -1 ;
             else
                 i= mid +1 ;
          }
    }
    if (i>j) {
          return0;
    }
  }
  int main (){
        int n,i, a[20], c, b, S1 , S2 ;
        Printf("enter the number of elements of array");
        Scanf("%d", &n);
        Print f("enter the element of array");
```

64MP AI QUAD CAMERA
Shot on realme XT
2020/05/05 20:50

Scanned with CamScanner

```
For (i=0 ; i<n ; i++)
    scanf("%d", &a[i]);
Sort (a,n);
for (i=0 ; i<n ; i++)
     Printf("%d", a[i]);
Printf ("enter the element to find in array");
scanf ("%d", &b);
c= binary (a,b,n);
if (c! =0){
      Printf( "element is found at Position %d", c);
}
else {
     Printf ("element not found \n");
}
Printf ("enter the position of array to find sum and
      Product \n");
scanf ("%d %d", & s1 & s2);
s1 --;
s2 --;
Printf(" the sum is %d ", a[s1] + a[s2]);
Printf (" the Product is %d", a[s1] * a[s2]);
}
```

64MP AI QUAD CAMERA
Shot on realme XT
2020/05/05 20:50

Scanned with CamScanner

2) Sort the array using merge sort where elements are taken from user and find the product of k^th elements from the first and last where k is taken from the user.

```
# include <stdio.h>

# include <stdio.h>

void merge (int arr[], int1, intc, inte)

{

int i, j, k;

int n1 = c-1+1 ;

int n2 = e-c;

int L[n1] , R[n2] ;

for (i=0 ; i<n1 ; i++ )

L[i] = arr [i+1];

For (j=0 ; j<n2 ; j++)

R[j] = arr [c+1 +j] ;

i=0;

j=0;

k=1;

while (i<n1 && j<n2){

if (L[i] <= R[j]){

arr {k] = L[i] ;

i++ ;

}

else{

arr [k] = R[j] ;

j++;

}

k++;

}
```

64MP AI QUAD CAMERA
Shot on realme XT
2020/05/05 20:50

Scanned with CamScanner

```
while (j<n2){

arr[k] = R[j];

j++;

}
}

void merge sort (int arr[] ,int1, inte){

if (r>1){

int m = 1+ (e-1)/2 ;

merge sort (arr,1,c);

merge sort (arr,<+1,e);

merge (arr, 1,c, e);

}
}
void print array (int A[] ,int size){

int i;

For(i=0; i<size ;i++)

printf("%d", A[i]);

printf(" \n");

}
int main()
{
int arr[7];
int i;
int arr_size = size of (arr)/size of (arr[0]);

For(i=0; i<arr-size ;i++){

printf("enter the elements:");

scanf("%d", &arr[i]);

}
printf("given array is \n");
print (arr, arr-size);
merge sort (arr, 0, arr-size-1);
```

64MP AI QUAD CAMERA
Shot on realme XT
2020/05/05 20:51

Scanned with CamScanner

```
Printf ("\n Sorted array is \n");

PrintfArray (arr, arr-size);

int k;

Printf(" enter the value of k :");

scanf("%d", &k);

int From first = arr[k-1];

int from last = arr[7-(k)];

Printf("%d", From last * from first);

return 0;

}
```

Output:-

```
Enter the elements : 65

Enter the elements : 98

enter the elements : 32

Enter the elements : 25

Enter the elements : 15

Enter the elements : 46

Enter the elements : 74

Given array is

65  98  32  25  15  46  74

Sorted array is

15  25  32  46  65  74  98

Eenter the vaue of k: 5

2080
```

P AI QUAD CAMERA
on realme XT
2020/05/05 20:51
Scanned with CamScanner

3) Discuss Insertion sort and selection sort with examples.

Ans:

Insertion sort is a simple sorting algorithm that builds the final sorted array one item at a time. It is ~~mean~~ much less efficient on large lists than more advanced algorithm such as heapsort,

merge sort. The time complexity for insertion sort for worst case and best case are $O(n^2)$ and $O(n)$ respectively.

Algorithm:

Sort & arr[ ] of size n insertion sort (arr, n)

loop from i = 1 to n-1.

Pick the element arr[i] and insert it into sorted sequence arr[0.....i-1].

Example:

	5	6	9	2	1
i=	0	1	2	3	4

let us loop for i=1 to i=4.

i=1 since 5<6, so 5 will as it is.

5	6	9	2	1

i=2, since 9<6, so 6 will Remain in its Position:

5	6	9	2	1

QUAD CAMERA
alme XT
2020/05/05 20:51

Scanned with CamScanner

i=3, 2 will move to the begnining and other elements 5 to 9 will move one position a head && their current position.

i.e

2	5	6	9	1

i=4, 1 will move to the begning and other elements from 2 to 9 will move one position a head && their current position.

i.e

1	2	5	6	9
0	1	2	3	4

Selection Sort:-

selection sort is the sorting algorithm && an array to finding the minimum element repeatedly from unsorted path and putting in the beginning. the time complexity for selection sort for worth and best case is $O(n^2)$ and $O(n^2)$ resepectively.

Example:

13	22	11	5	9
0	1	2	3	4

Find the minimum element in arr[0...4] and place it at beggining and displace the position of preveious element to the new element.

i.e

5	[illegible]	[illegible]	13	9

64MP AI QUAD CAMERA
Shot on realme XT
2020/05/05 20:5

Scanned with CamScanner

Find the minimum element in arr [1... 4] and place it at beginning arr [i.... 4].

Find the minimum element in arr [2.. 4] and Place it at begining ~~of~~ arr [2.. 4]

5	9	11	13	22

Now all the elements has settled down in desending order. Repeat the same Process for Further rearrangment.

4) Sort the array using bubble sort where elements are taken from the user and display the elements.

i. in alternate order.

ii. Sum of elements in odd positions and product of elements in even positions

iii. elements which are divisible by m where m is taken from the user.

Ans:-

```
# inlude <stdio.h>
void main (){
int a[100], n, i, j, temp, sum=0, prod=1, m;
 Printf ("enter the elements");
  scanf ("%d", &n);
  [illegible] integers %d\n", n);
 for (i=0; i<n ; i++) {
```

64MP AI QUAD CAMERA
Shot on realme XT
2020/05/05 20:51
Scanned with CamScanner

```
        scanf("%d ; &a[i]);
    }
    for (i=0 ; i<n-1 ; i++){
        for (j=0 ; j<n-i-1; j++){
            if (a[j]>a[j+1]){
                temp = a[j];
                a[j] = a[j+1];
                a[j+1] = temp;
            }
        }
    }
    printf("sorted list in ascending order");
    for(i=0;  i<n;  i++){
        printf("%d\n", a[i]);
    }

i. printf("the alternate orden is");
    for (i=0; i<n; i++) {
        if (i%2 = =0){
            printf("%d", a[i]);
        }
    }
```

64MP AI QUAD CAMERA
Shot on realme XT
2020/05/05 20:51
Scanned with CamScanner

```
ii.  For (i=0 ; i<n ; i++){
         if (i%2!=0){
              sum = sum + a[i];
         }
     }
     printf("sum of add index is %d",sum);
     for(i=0 ; i<n ; i++){
          if (i%2 ==0){
               prod = prod * a[i];
          }
     }
     printf("product of even position is %d", prod);

iii. printf ("Enter the value of m");
     scanf ("%d", &m);
     for(i=0 ; i<n ; i++){
          if (a[i] % m ==0){
               printf("%d",a[i]);
          }
     }
```

64MP AI QUAD CAMERA
Shot on realme XT
2020/05/05 20:52
Scanned with CamScanner

5) write a recursive program to implement binary search ?

```
#include <Stdio.h>
int recursive Binary search (int arr[] , int
        Start_index, int end_index, int element ){
    if (end_index >= start_index){
        int middle = Start_index + (end_index - sort_index)\2 ;
      if (array [middle] == element)
          return middle ;
      if (array [middle] > element)
            return recursive Binary Search\array ,
            start_index, middle_1 , element);
            return recursive Binary search (array,
             middle +1 , end_index , element);
      }
      return -1 ;
   }
   int main (void){
       int array [] = {3, 13, 17, 5, 23, 57, 73};
       int n=7 ;
       int element = 23 ;
       int found_index = recursive Binary search
       (array , 0 , n-1 , element);
       if (found_index = = -1 ){
              Printf("element not found in the array"];
      }
      else{
          printf("element ~~not~~ found at index : %d",
                found_index);
      }
```

64MP AI QUAD CAMERA
Shot on realme XT
2020/05/05 20:52
Scanned with CamScanner

```
return 0;
}
```

Output :

```
Element found at Index : 4
```

64MP AI QUAD CAMERA
Shot on realme XT
2020/05/05 20:52

Scanned with CamScanner