ASSIGNMENT-6

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

ⓐ using Binary search find the elements and the location in the array where the element is asked from user.

ⓑ Ask the user to enter any two locations print the sum and the product of values at those locations in the sorted array.

```
#include <stdio.h>
void sort main(int a[], int n)
{
    int a[100], n, i, j, temp, sum=0, prod=1, ;
    for(i=0; i<n; i++)
    {
        for(j=0; j<n; j++)
        {
            if(a[i]<a[j])
            {
                temp=a[i];
                a(i)=a[j];
                a[j]=temp;
            }
        }
    }
}
```

```
int binary (int a[], int e, int n)
{
    int i=0, j=n-1, mid;
    while (i<=j)
    {
        mid = (i+j)/2;
        if (a[mid]==e)
            return mid+1;
        else
        {
            if (e<a[mid])
                j=mid-1;
            else
                i=mid+1;
        }
    }
    if (i>j)
    {
        return 0;
    }
}
int main()
{
    int n, i, a[20], f, e, m1, m2;
    printf("Enter the NO of elements of array :\n");
```

```
scanf("%d", &n);
printf("Enter the elements in Array:\n");
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 elements to be searched");
scanf("%d", &e);
if(f!=0)
    printf("Element is found at %d position", f);
else
    printf("Element not found\n");
printf("Enter the position of array to find sum and product");
scanf("%d%d", &m1, &m2);
m1--;
m2--;
printf("The sum is %d", a[m1]+a[m2]);
printf("The product is %d", a[m1]*a[m2]);
}
```

output:

```
Enter the No of elements of array : 5

Enter the elements in array :

12
5
6
7
9

12  9  7  6  5

Enter the element to find in array : 6

Element is found at 4 position

Enter the positions of array to find sum and product

1
3

The sum is 19

The product is 84
```

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

```
#include <stdio.h>
void merge (int *a, int s, int e)
{
        int temp[15];
        int i, j, k, mid;
        mid = (e+s)/2;
        i=s;
        j = mid+1;
        k =s;
        while (i<=mid && j<=e)
        {
            if(a[i] < a[j])
            {
                temp[k++] = a[i++];
            }
            else
            {
                temp[k++]= a[j++];
            }
        }
        while (j<=e)
        {
            temp[k++] = a[j++];
        }
```

```
    while (i<=mid)
    {
        temp[k++] = a[i++];
    }
    for(i=s; i<=e; i++)
    {
        a[i] = temp[i];
    }
}
void mergesort (int *a, int s, int e)
{
    int m;
    if (s<e)
    {
        m = (s+e)/2
        mergesort(a,s,m);
        mergesort (a, m+1, e);
        mergesort (a,s,e);
    }
}
```

```
int main()
{
    int n, i, a[20], prod1=0, prod2=1, k, j, c=1;
    printf("Enter the NO of elements of array");
    scanf("%d", &n);
    printf("Enter the elements of array \n");
    for(i=0; i<n; i++)
    {
        scanf("%d", &a[i]);
    }
    printf("Enter the position till which you want to find the product
        from first and Last \n");
    scanf("%d", &k);
    i=0;
    j=n-1;
    while(c<=k)
    {
        prod1 = prod1 * a[i];
        prod2 = prod2 * a[j];
        i++;
        j++;
        c++;
    }
```

```
    printf ("%d %d ", prod1, prod2);
}
```

output :

```
Enter the NO of elements of array: 5
Enter the elements in Array :
12
55
91
7
6
6  7  12  55  91
Enter the position till which you want to find product from
First and Last :
42
5005
```

③ Discuss Insertion sort and selection sort with examples.

Ⓐ Insertion-sort:

Insertion sort is based on the Idea that one element from the Input element is consumed in each iteration to finds its correct position i.e; the position as per ascending order or descending order accordingly.

Algorithm:

① : If it is the first element, it is already sorted. return 1;

② : pick next element

③ : compare with all elements in the sorted sub-list

④ : shift all the elements in the sorted sub-list that is greater than the value to be sorted

⑤ : Insert the value.

⑥ : Repeat until list is sorted

Time complexity: Best-case : $O(n)$

Average : $O(n^2)$

Worst : $O(n^2)$

Example:

a = | 10 | 2 | 12 | 7 | 6 |

$\Rightarrow$ [10, 2, 12, 7, 6] {Representing an Bracket}

① ⇒ [10, 2, 12, 7, 6]

② ⇒ [2, 10, 12, 7, 6]

③ ⇒ [2, 10, 12, 7, 6]

④ ⇒ [2, 7, 10, 12, 6]

⑤ ⇒ [2, 6, 7, 10, 12]

Selection-sort:

selection-sort Algorithm is based on the idea of finding maximum or minimum element in an unsorted array and then putting it in its correct position in a array.

Algorithm:

① : set MIN to location 0.

② : search the minimum element in the list

③ : swap with value at location MIN.

④ : increment MIN to point to next element

⑤ : Repeat until list is sorted.

Time complexity:

Best-case : $O(n^2)$

Average : $O(n^2)$

Worst : $O(n^2)$

Example:

$a =$ | 10 | 2 | 12 | 7 | 6 |

$\Rightarrow$ [10, 2, 12, 7, 6] $\Rightarrow$ {Representing in Brackets}

① $\Rightarrow$ [2, 10, 12, 7, 6]

② $\Rightarrow$ [2, 6, 12, 7, 10]

③ $\Rightarrow$ [2, 6, 7, 12, 10]

④ $\Rightarrow$ [2, 6, 7, 10, 12]

⑤ $\Rightarrow$ [2, 6, 7, 10, 12]

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

```
#include <stdio.h>
void main()
{
    int a[100], i, j, n, temp, sum=0, prod=1, m;
    printf("Enter the Numbers of elements to be used \n");
    scanf("%d", &n);
    printf(" Enter %d Integers \n", n);
    for (i=0; i<n; i++)
    {
        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("\n sorted array is \n");
for(i=0;i<n;i++)
{
    printf("%d ",a[i]);
}
```

(i) In Alternate order

```
printf(" Alternate order : \n");
for(i=0; i<n;i++)
{
    if(i%2 ==0)
    {
        printf(" %d ", a[i]);
    }
}
```

(ii) sum of elements in odd positions and product of elements in even positions.

```
for(i=0;i<n;i++)
{
    if(i%2!=0)
    {
        sumo = sumo+a[i];
    }
}
```

```
for (i=0; i<n; i++)
{
    if (i%2==0)
    {
        prod = prod * a[i];
    }
}
printf (" sum of odd-Index is \n %d ", sumo);
printf (" product of even-Index is %d \n ", prod);
```

(iii) Elements which is divisible by m.

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

output:

```
Enter the Number of elements to be used : 6

Enter 6 Integers:
12
9
1
5
7
3
sorted array is
1 3 5 7 9 12

Alternate order :
1 5 9

sum of odd-Index is : 22

product of Even - Index is : 45

Enter the value of m: 3

3 9 12.
```

⑤ write a recursive program to implement Binary search.

```
#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;
            }
        }
    }
}
int binary (int a[], int e, int i, int j)
{
    int mid = (i+j)/2;
    if (a[mid] == e)
        return mid+1;
```

```
    else
    {
        if (e< a[mid])
            return binary (a, e, i, mid - 1);
        else
            return binary (a, e, mid + 1, j);
    }
}
int main()
{
    int n, i, a[20], f, e;
    printf ("Enter the No: of elements to be used \n");
    scanf ("%d", &n);
    printf (" Enter %d Integers \n", n);
    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 be searched\n");
    scanf("%d", &e);
    f = binary(a, e, 0, n-1);
    if (f!=0)
        printf("Element is found at %d position", f)
    else
        printf("Element Not-found\n");
}
```

output:

```
Enter the No of elements to be used : 6
Enter 6 Integers
17 9 6 45 2 1

1 2 6 9 17 45

Enter the element to be searched :17
Element is found at 5 position.
```