Assignment

1.

```
#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 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 ");
    Scanf("%d\n", &n);
    printf("Enter the elements of array \n");
    Scanf("%d\n", &a[i]);          for(i=0; i<n; i++)
    sort(a,n);
    for(i=0; i<n; i++)
    {
        printf(%d", a[i]);
    }
    printf("\n Enter the elements to find in array \n");
    Scanf("%d", &e);
    f = binary(a,e,n);
    if (f!=0)
    {
        printf("\n Elements is found at %d position \n", f);
    }
    else
        printf("Element not found \n");
    printf("Enter position of array to find sum & product \n");
    Scanf("%d %d", &m1, &m2);
    m1--; m2--;
    printf("the sum is %d\n", a[m1]+a[m2]);
    printf("The product is %d \n", a[m1]*a[m2]);
}
```

2.

```
#include <stdio.h>
#include <stdlib.h>
void merge(int arr[], int l, int m, int r)
{
    int i, j, k;
    int n1 = m - l + 1;
    int n2 = r - m;
    int L[n1], R[n2];
    for (i = 0; i < n1; i++)
        L[i] = arr[l + i];
    for (j = 0; j < n2; j++)
        R[j] = arr[m + 1 + j];
    i = 0; j = 0; k = l;
    while (i < n1 && j < n2)
    {
        if (L[i] <= R[j])
        {
            arr[k] = L[i];
            i++;
        }
        else
        {
            arr[k] = R[j];
            j++;
        }
        k++;
    }
    while (i < n1)
    {
        arr[k] = L[i];
        i++;
        k++;
    }
```

```
    while (j<n2)
    {
        arr[k] = R[j];
        j++;
        k++;
    }
}
void mergesort(int arr[], int l, int r)
{
    if (l<r)
    {
        int m = l+(r-l)/2;
        mergesort(arr, l, m);
        mergesort(arr, m+1, r);
        merge(arr, l, m, r);
    }
}
void printArray(int A[], int size)
{
    int i;
    for (i=0; i<size; i++)
        printf("%d", A[i])
    printf("\n");
}
int main()
{
    int arr[5];
    int i;
    int arr_size = sizeof(arr)/sizeof(arr[0]);
    for(i=0; i<arr_size; i++)
    {
        printf("Enter the elements");
        scanf("%d", &arr[i]);
    }
    printf("Given array is \n");
    printArray(arr, arr_size);
```

```
mergesort(arr, 0, arr_size - 1);
printf("\n sorted array is \n");
printArray(arr, arr_size);
int k;
printf("Enter the value of k");
scanf("%d", &k);
int fromfirst = arr[k-1];
int fromlast = arr[5-(k)];
printf("%d", fromlast * fromfirst);
return 0;
}
```

3. Insertion sort :-

It is the sorting mechanism where the sorted array is built having one item at a time. The array elements are compared with each other sequentially and then arranged simultaneously in some particular order.

Worst complexity $\rightarrow n^2$ Best complexity $\rightarrow n$

Average $\rightarrow n^2$ Space $\rightarrow 1$

Ex :- The lower part of an array is maintained be be sorted.

12, 11, 13, 5, 6.

Loop for $i = 1$ to 4.

$i = 1$ $11 < 12$ move 11 to 12

11, 12, 13, 5, 6.

$i = 2$ $13 > 12$. No change.

11, 12, 13, 5, 6

$i = 3$ $5 < 13$. 13 move back

$5 < 12$ and 5 goes to first.

$5 < 11$

5, 11, 12, 13, 6.

$i = 4$.

$6 < 11$

$6 < 12$

$6 < 13$

5, 6, 11, 12, 13.

Selection sort:-

The sorting algorithm is an in-place comparison based algorithm in which the list is divided into two parts, the sorted part at the left and the unsorted part at the right end.

arr[] = 84, 69, 13, 24, 9

1) First find min from [1 ... 4]

9, 69, 13, 24, 84

2) second min from [2 ... 4]

9, 24, 13, 69, 84.

3) third min from [3 ... 4)

9 13, 24, 69, 84.

4).

```
#include <stdio.h>
void main()
{
    int a[100], n, i, j, temp, sum=0, prod=1, m;
    printf("Enter number of elements \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[i] = a[i+1];
            a[j+1] = temp;
        }
    }
}
printf("\n Sorted list in ascending order \n");
for(i=0; i<n; i++)
{
    printf("%d\n", a[i]);
}
printf("the alternate order is ");
for (i=0; i<n; i++)
{
    if (i%2 == 0)
    {
        printf("%d", a[i]);
    }
}
for(i=0; i<n; i++)
{
    if (i%2 != 0)
    {
        sumo = sumo + a[i];
    }
}
printf("\n sum of odd index is %d", sumo);
for(i=0; i<n; i++)
{
    if (i%2 == 0)
    {
        prod = prod * a[i];
    }
}
printf("\n product of odd index is %d", prod);
printf("\n Enter the value of n\n");
```

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

5.
```
#include<stdio.h>
#include<stdlib.h>
void Binary search(int arr[], int num, int first, int last)
{
    if (first > last)
    {
        printf("Number is no there");
    }
    else
    {
        int mid;
        mid = (first+last)/2;
        if (arr[mid]==num)
        {
            printf("Element is found at index %d", mid);
            exit(0);
        }
        else if (arr[mid]>num)
        {
            Binary search(arr, num, first, mid-1);
        }
        else
        {
            Binary search(arr, num, mid+1, last);
        }
    }
}
```

```
main()
{
    int arr[] = { 3, 7, 9, 13, 15, 18 };
    int num = 9;
    int first = 0, last =
    last = (size of (arr) / size of (arr[0]) - 1);
    Binary Search (arr, num, first, last);
}.
```