```
#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[i];
                      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);
       printf("Enter the elements of 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 element of array");
        scanf("%d",&e);
        f=binary(a,e,n);
        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\n");
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]);
}
```

Expt. Name : 2

```
#include <Stdlib.h>
#include <Stdio.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[i]
      {
        arr[k] = L[i];
        i++;
    }
```

Expt. Name :

```
    else
    {
      arr[k] = R[j];
      j++;
    }
      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);
   }
}
```

Expt. Name :

```
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("Sorted array is");
   printArray(arr, arr_size);
   int k;
   printf("Enter the k");
   scanf("%d", &k);
```

Expt. Name :

```
int fromfirst=arr[k-1];
 int fromlast = arr[5-(k)];
 Printf("%d", fromlast * fromfirst);

 return 0;
}
```

Name : 3

Selection Sort

The selection sort algorithm sorts an array by repeatedly finding the minimum element from unsorted part and putting it at the beginning. The algorithm maintains two subarrays in a given array

Insertion Sort

Insertion sort is a simple sorting algorithm that works the way we sort playing cards in our hands.

Expt. Name : 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++)
{
for (j=0; j<n-i-1; j++)
{
IF (a[j] > a[j+1])
{
temp = a[i];
a[j] = a[j+1];
a[j+1] = temp;
}
}
}
printf("Sorted list in ascending\n")
```

Signature.....

Expt. Name :

```
for(i=0;i<n;i++)
{
printf("%d\n",a[i]);
}
printf("The alternate elements are");
for(i=0;i<n;i++)
{
  if(i%2==0)
  {
    printf("%d",a[i]);
  }
}
for(i=0;i<n;i++)
{
   if(i%2!=0)
   {
     sum0=sum0+a[i];
   }
}
printf("Sum of odd index is %d",sum0);
for(i=0;i<n;i++)
 {
   if(i%2==0)
   {
     prod=prod*a[i];
   }
 }
```

Signature.....

Expt.No : Date : Page No :

Expt. Name :

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

Expt. Name :

```
#include <stdio.h>
int recursiveBinarySearch (int array[],int
start_index,int end_index,int element){
if(end_index >=start_index){
  int middle = start_index+(end_index-start_index)/2;
  if(array[middle]==element)
     return middle;
  if (array[middle]>element)
  return recursiveBinarySearch(array,start_index,middle-1,element);
return recursiveBinarySearch (array,middle+1,end_index,element);
}
return-1;
}
int main(void){
  int array[]={1,4,7,9,16,56,70};
 int n=7;
  int element=9;
   int found_index=recursiveBinarySearch(array,0,n-1,element);
if(found_index==-1){
    printf("Element not found in array");
 }
 else{
    printf("Element found at index:%d",found_index);
 }
 return 0;
}
```

Signature.....