Bubble sort:

```
#include<iostream>
using namespace std;
int main(){
    int n,i,j,temp;
    cout<<"Enter the array size:"<<endl;
    cin>>n;
    int A[n];
    cout<<"Enter the array element:"<<endl;
    for(i=0;i<n;i++){
        cin>>A[i];
    }
    for(i=1;i<n;i++){
        for(j=0;j<n-i;j++){
            if(A[j]>A[j+1]){
                temp=A[j];
                A[j]=A[j+1];
                A[j+1]=temp;
            }
        }
    }
    for(i=0;i<n;i++){
        cout<<A[i]<<" ";
    }
    return 0;

}
```

Insertion sort

```
#include<iostream>
using namespace std;
int main(){
    int n,i,j,temp;
    cout<<"Enter the array size:"<<endl;
    cin>>n;
    int A[n];
    cout<<"Enter the array element:"<<endl;
    for(i=0;i<n;i++){
        cin>>A[i];
    }
    for(int i=1;i<n;i++)
    {
        int temp=A[i];
        int j=i-1;
        while(j>=0 && A[j]>temp)
        {
            A[j+1]=A[j];
            j--;
        }
        A[j+1]=temp;

    }
    cout<<"Sorted Array: ";
    for(int i=0;i<n;i++)
    {
        cout<<A[i]<<" ";
    }
    return 0;

}
```

Selection sort

```
#include<iostream>
using namespace std;
int main()
{
    int n;
    cout<<"Enter array size: "<<endl;
    cin>>n;
    int A[n];
    cout<<"Enter array elements: ";
    for(int i=0;i<n;i++)
    {
        cin>>A[i];
    }
    for(int i=0;i<n-1;i++){
        int Min=i;

    for(int j=i;j<n;j++)
{
    if(A[Min]>A[j])
    {
       Min=j;
    }
}
if(Min!=i){
 int temp=A[i];
        A[i]=A[Min];
        A[Min]=temp;
    }

    }

    cout<<"Sorted array: ";
     for(int i=0;i<n;i++)
    {
        cout<<A[i]<<" ";
    }
    return 0;
}
```

Binary search

```
    int Big=0, End=n-1, Mid, item, loc=-1;
    cout<<"Enter the target value:
"<<endl;
    cin>>item;
    while(Big<=End){
        Mid=(Big+End)/2;
        if(item>=A[Mid])
        {
            Big=Mid+1;
            loc=Mid;
        }
       else
            End=Mid-1;
    }
    cout<<loc<<endl;
    return 0;
}
```

Linear search

```
int Loc=0,item;
    cout<<"Enter the target value: "<<endl;
    cin>>item;
    for(int k=1;k<n;k++){
        if(item==A[k]){
            Loc=k;
        }

    }
    if(Loc==0)
        cout<<"Item is not found in the
data."<<endl;
    else
        cout<<Loc<<" is the location of
item."<<endl;
```

Counting sort

```
#include<iostream>
using namespace std;
int main(){
int size;
cout<<"Enter the array size:"<<endl;
cin>>size;
int arr[size];
cout<<"Enter the array element:"<<endl;
for(int i=0;i<size;i++){
    cin>>arr[i];
}
int output[size];
int max=arr[0];

for(i=1;i<size;i++){
    if(arr[i]>max){
        max=arr[i];
    }
}
int count[max +1];

for(i=0;i<max + 1;i++){
    count[i]=0;
}
for(i=0;i<size;i++){
    count[arr[i]]++;

}
for(i=1;i<=max;i++){
    cont[i]+=count[i-1];
}
for(int i = Size - 1; i >= 0; i--) {

  Output[Count[arr[i]] - 1] = arr[i];

  Count[arr[i]]--;

cout<<"The Sorted Array:"<<endl;
for (int i = 0; i <Size; i++){
cout<<Output[i]<<" "<<endl;
}
return 0;
}
```

Radix sort

```
#include <iostream>
using namespace std;
void countingSort(int A[], int size, int place) {
  const int max = 10;
  int output[size];
  int count[max];

 for (int i = 0; i < max; ++i)
    count[i] = 0;

  for (int i = 0; i < size; i++)
    count[(A[i] / place) % 10]++;

  for (int i = 1; i < max; i++)
    count[i] += count[i - 1];

   for (int i = size - 1; i >= 0; i--) {
    output[count[(A[i] / place) % 10] - 1] = A[i];
    count[(A[i] / place) % 10]--;
  }
 for (int i = 0; i < size; i++)
    A[i] = output[i];
}
void radixsort(int A[], int size) {
   int max = A[0];
  for (int i = 1; i < size; i++){
    if (A[i] > max)
      max = A[i];}
for (int place = 1; max / place > 0; place *= 10)
    countingSort(A, size, place);
}
int main() {
    int n;
    cin>>n;
    int A[n];
for(int i=0;i<n;i++)
{
    cin>>A[i];
}
  radixsort(A, n);
   for (int i = 0; i < n; i++)
    cout << A[i] << " "<< endl;
return 0;
}
```

Mergr sort

```
#include <iostream>
#include <vector>
using namespace std;

void merge(int arr[], int p, int q, int r) {


  int n1 = q - p + 1;
  int n2 = r - q;


  vector<int> L(n1);
  vector<int> M(n2);

  for (int i = 0; i < n1; i++)
    L[i] = arr[p + i];

  for (int j = 0; j < n2; j++)
    M[j] = arr[q + 1 + j];


  int i = 0, j = 0, k = p;


  while (i < n1 && j < n2) {
    if (L[i] <= M[j]) {
      arr[k] = L[i];
      i++;
    } else {
      arr[k] = M[j];
      j++;
    }
    k++;
  }


  while (i < n1) {
    arr[k] = L[i];
    i++;
    k++;
  }

  while (j < n2) {
    arr[k] = M[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 arr[], int size) {
  for (int i = 0; i < size; i++)
    cout << arr[i] << " ";
  cout << endl;
}

int main() {
  int arr[] = {6, 5, 12, 10, 9, 1};
  int size = sizeof(arr) / sizeof(arr[0]);
  mergeSort(arr, 0, size - 1);
  cout << "Sorted array: \n";
  printArray(arr, size);
  return 0;
}
```