The *Bubble* Sort

```
void Bubble(Itemtype a[], int N )
{
 int i, j;

 for (i = N - 1; i > 0; --i)
  for (j = 1; j <= i; ++j)
   if (a[j-1] > a[j])
    swap(a[j-1],a[j]);
}
```

```
void Bubble1(Itemtype a[], int N )
{

 DidSwap = true;

 for (i = N-1; DidSwap && i > 0; i--)
 {
  DidSwap = false;
  for (j = 1; j <= i; ++j)
   if (a[j-1] > a[j])
   {
     DidSwap = true;
     swap(a[j],a[j-1]);
   }
  }
}
```

The *Insertion* Sort

```
void Insertion(ItemType a[], int N)
{
 int i, j;
 ItemType v;

 for (i = 1; i < N; ++i)
 {
  v = a[i];
  j = i;

  while (j > 0 && a[j-1] > v)
  {
    a[j] = a[j-1];
    j = j-1;
  }

  a[j] = v;
 }
}
```

The *Selection* Sort

```
void selection( ItemType a[], int N)
{
 int i, j, min;

 for (i = 0; i < N-1; ++i)
 {
  //Elements from 0 to i-1 are in the correct place

   min = i;
   for (j = i+1; j < N; ++j)
    if (a[j] < a[min])
       min = j;

   swap(a[i], a[min]);

  //Elements from 0 to i are in the correct place
 }
}
```

The *Merge* Sort

```
void mergesort(ItemType a[]; Itemtype temp[];int left; int right)
{
 int mid;

 if (right > left)
 {
  mid = (right + left)/2;

  mergesort(a, temp, left, mid);
  mergesort(a, temp, mid+1, right);

  merge(a,temp,left,mid+1,right);
 }
}

void merge(ItemType a[]; ItemType temp[]; int left; int mid; int right)
{
 int i = left, j = mid, k = left;

 while (i < mid && j <= right)
 {
  if (a[i] <= a[j])
     temp[k++] = a[i++];
  else
     temp[k++] = a[j++];
 }

 while (i < mid)
    temp[k++] = a[i++];

 while (j <= right)
    temp[k++] = a[j++];

 for (i = left; i <= right; ++i)
  a[i] = temp[i];
}
```

The *QuickSort*

```
void Quicksort(ItemType a[], int left, int right)
{
 int i, j;
 ItemType v;

 if (right > left)
 {
  v = a[right]; //Pivot element is simply the righmost one!
  i = left - 1;
  j = right;

  do
  {
   do
    i = i + 1;
   while ( a[i] < v);

   do
    j = j - 1;
   while (j >= left && a[j] > v);

   if (i < j)
    swap(a[i], a[j]);
  }
  while (j > i);

  swap(a[i], a[right]);

  Quicksort(a, left, i-1);
  Quicksort(a, i+1, right);
 }
}
```

Summary of Sort Algorithm Complexity

Algorithm	Comparisons	Moves
Bubble Sort	$O(n^2)$	$O(n^2)$
Insertion Sort	$O(n^2)$	$O(n^2)$
Binary Insertion Sort	$O(n \log n)$	$O(n^2)$
Selection Sort	$O(n^2)$	$O(n)$
Merge Sort	$O(n \log n)$	$O(n \log n)$
QuickSort (*average)*	$O(n \log n)$	$O(n \log n)$
QuickSort (*worst case)*	$O(n^2)$	$O(n^2)$