

Animation Demos

<http://www-cse.uta.edu/~holder/courses/cse2320/lectures/applets/sort1/heapsort.html>

<http://cg.scs.carleton.ca/~morin/misc/sortalg/>

Sorting Algorithms

- Selection Sort
- Insertion Sort
- Bubble Sort
- Shaker Sort
- Shell Sort
- Merge Sort
- Heap Sort
- Quick Sort

- Bucket & Radix Sort
- Counting Sort

```
QuickSort(A, p, r)
  if (p < r) then
    q = Partition(A, p, r)
    QuickSort(A, p, q)
    QuickSort(A, q+1, r)
```

```
Partition(A, p, r)
  x = A[r]
  i = p-1
  for j = p to r-1 do
    if (A[j] <= x) then
      i++
      SWAP(A[i], A[j])
  SWAP(A[i+1], A[r])
  return i+1
```

Figure 8.5

Shellsort after each pass if the increment sequence is {1, 3, 5}

ORIGINAL	81	94	11	96	12	35	17	95	28	58	41	75	15
After 5-sort	35	17	11	28	12	41	75	15	96	58	81	94	95
After 3-sort	28	12	11	35	15	41	58	17	94	75	81	96	95
After 1-sort	11	12	15	17	28	35	41	58	75	81	94	95	96

ShellSort

```
public static void shellsort( Comparable [ ] a )
{
    for( int gap = a.length / 2; gap > 0;
        gap = gap == 2 ? 1 : (int) ( gap / 2.2 ) )
        for( int i = gap; i < a.length; i++ )
        {
            Comparable tmp = a[ i ];
            int j = i;

            for( ; j >= gap && tmp.compareTo( a[ j - gap ] ) < 0; j -= gap )
                a[ j ] = a[ j - gap ];
            a[ j ] = tmp;
        }
}
```

Sorting Algorithms

- Selection Sort
- Insertion Sort
- Bubble Sort
- Shaker Sort
- Shell Sort
- Merge Sort
- Heap Sort
- Quick Sort

- Bucket & Radix Sort
- Counting Sort

Bucket Sort

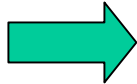
- N values in the range $[a..a+m-1]$
- For e.g., sort a list of 50 scores in the range $[0..9]$.
- **Algorithm**
 - Make m buckets $[a..a+m-1]$
 - As you read elements throw into appropriate bucket
 - Output contents of buckets $[0..m]$ in that order
- **Time $O(N+m)$**

Stable Sort

- A sort is **stable** if equal elements appear in the same order in both the input and the output.
- Which sorts are stable? Homework!

Radix Sort

3 2 9
4 5 7
6 5 7
8 3 9
4 3 6
7 2 0
3 5 5



7 2 0
3 5 5
4 3 6
4 5 7
6 5 7
3 2 9
8 3 9



7 2 0
3 2 9
4 3 6
8 3 9
3 5 5
4 5 7
6 5 7



3 2 9
3 5 5
4 3 6
4 5 7
6 5 7
7 2 0
8 3 9

Algorithm

for $i = 1$ to d do

sort array A on digit i using a stable sort algorithm

Time Complexity: $O((n+k)d)$