# UCSC Silicon Valley Extension

## Advanced C Programming

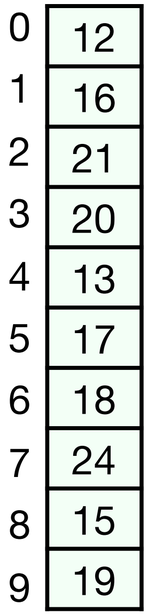
### Comb Sort

Instructor: Radhika Grover

# Overview

- When sorting with bubble sort in ascending order, small values at bottom (called turtles) move up slowly.
- Comb sort moves turtles up quickly.
- Gap between elements being compared equals 1 in bubble sort, but is variable in comb sort.

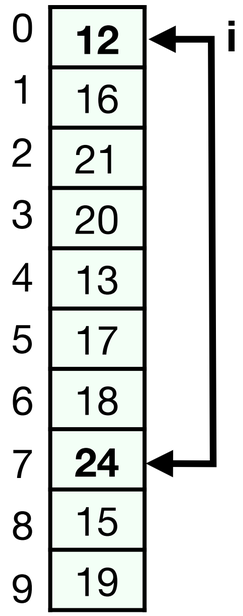
# Comb Sort




gap = n/1.3 = 10/1.3 ≈ 7

# Comb Sort

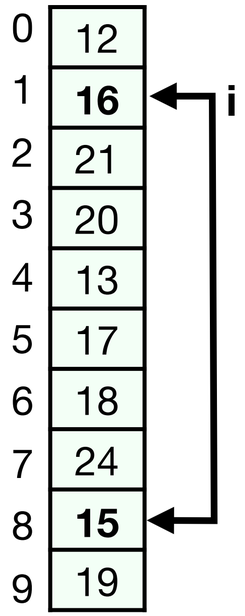
**Iteration 1, gap = 7**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

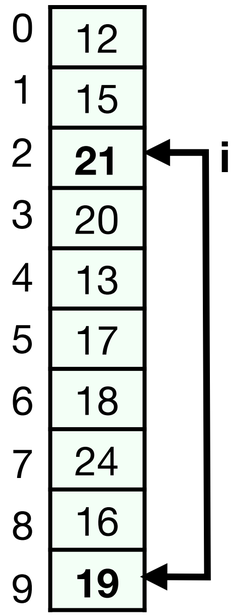
**Iteration 1, gap = 7**




array[i] > array[ i + gap]
swap and i++;

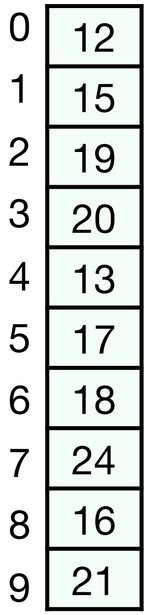
# Comb Sort

**Iteration 1, gap = 7**




array[i] > array[ i + gap]
swap and set new gap;

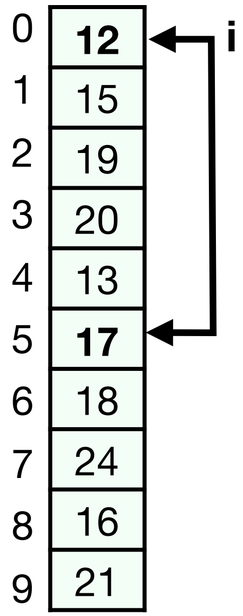
# Comb Sort




gap = n/1.3 = 7/1.3 ≈ 5

# Comb Sort

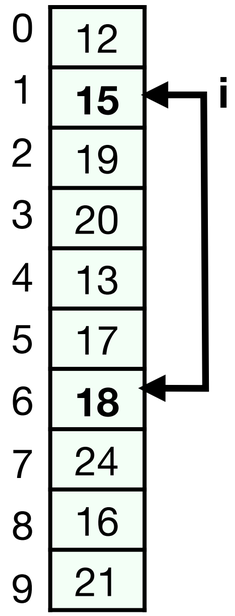
**Iteration 2, gap = 5**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

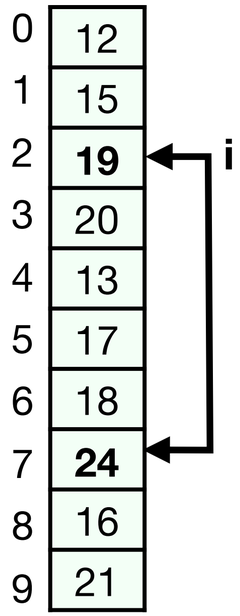
**Iteration 2, gap = 5**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

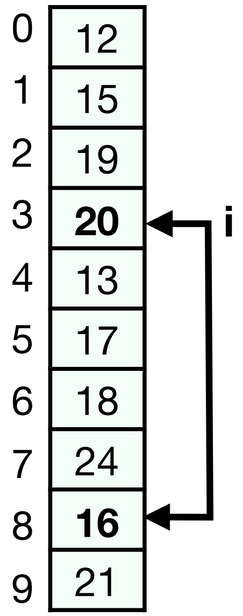
**Iteration 2, gap = 5**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

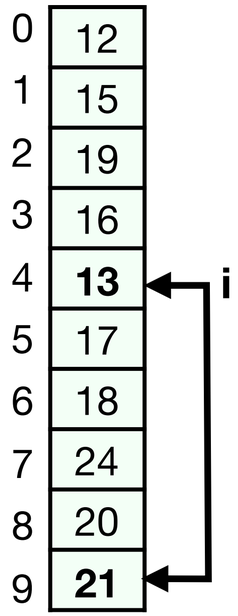
**Iteration 2, gap = 5**




array[i] > array[ i + gap]
swap and i++;

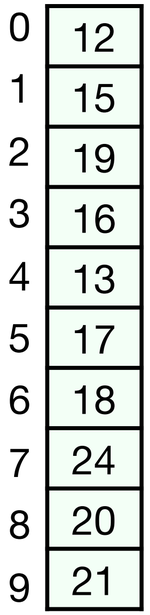
# Comb Sort

**Iteration 2, gap = 5**




array[i] <= array[ i + gap]
set new gap;

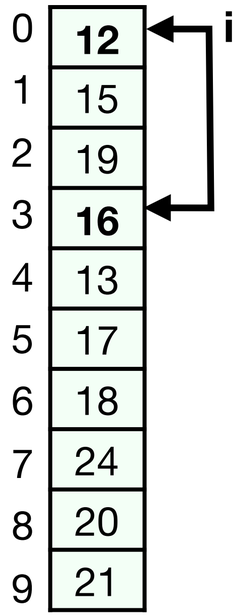
# Comb Sort




gap = n/1.3 = 5/1.3 ≈ 3

# Comb Sort

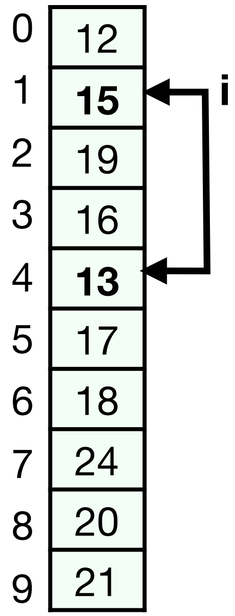
**Iteration 3, gap = 3**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

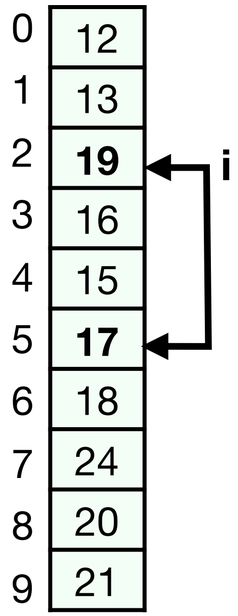
**Iteration 3, gap = 3**




array[i] > array[ i + gap]
swap and i++;

# Comb Sort

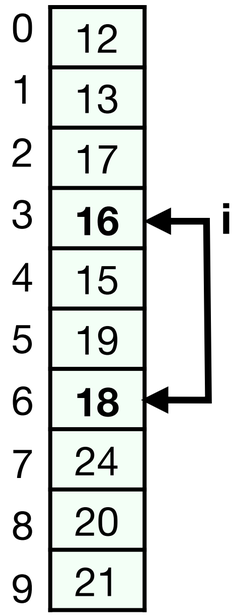
**Iteration 3, gap = 3**




array[i] > array[ i + gap]
swap and i++;

# Comb Sort

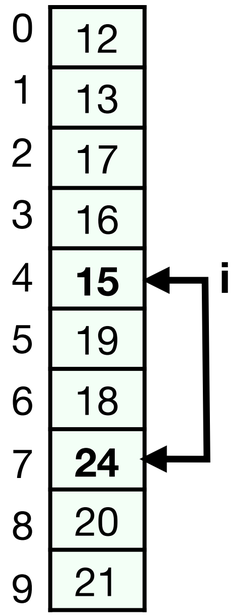
**Iteration 3, gap = 3**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

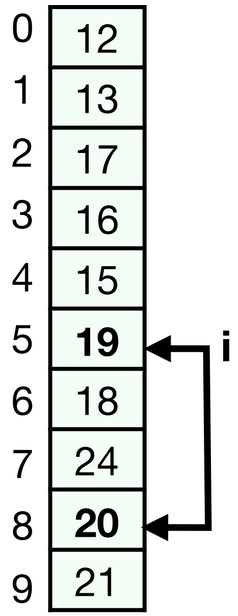
**Iteration 3, gap = 3**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

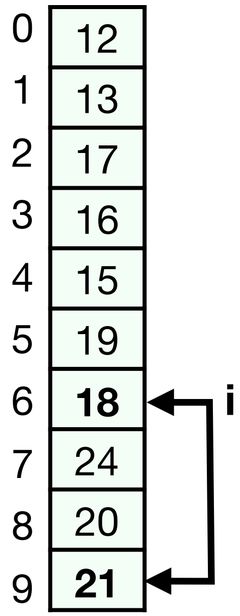
**Iteration 3, gap = 3**




```
array[i] <= array[ i + gap]
        i++;
```

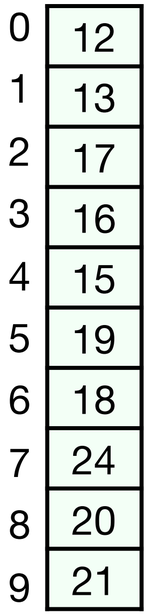
# Comb Sort

**Iteration 3, gap = 3**




array[i] <= array[ i + gap]
set new gap;

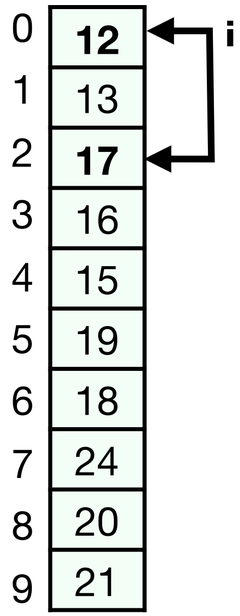
# Comb Sort




gap = n/1.3 = 3/1.3 ≈ 2

# Comb Sort

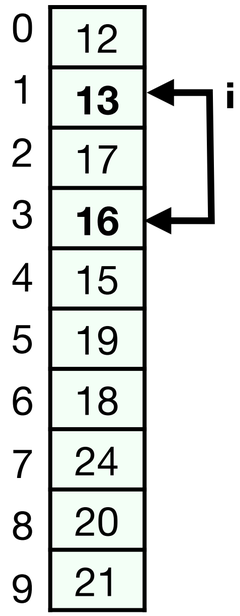
**Iteration 4, gap = 2**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

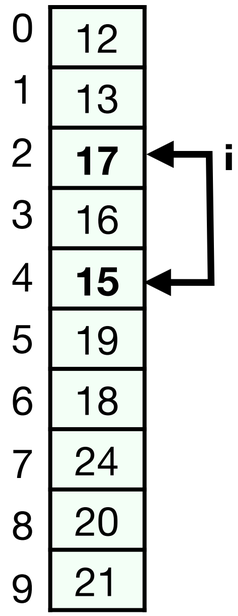
**Iteration 4, gap = 2**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

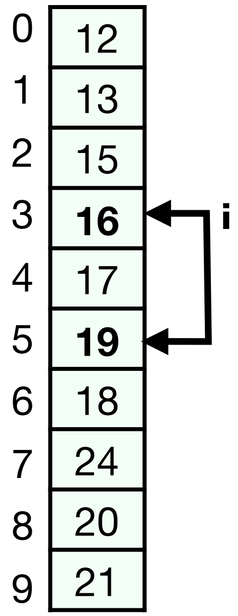
**Iteration 4, gap = 2**




array[i] > array[ i + gap]
swap and i++;

# Comb Sort

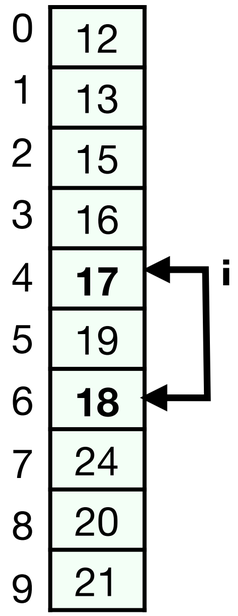
**Iteration 4, gap = 2**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

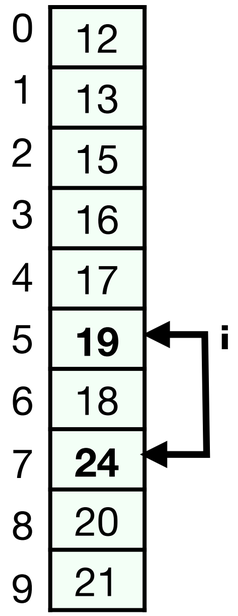
**Iteration 4, gap = 2**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

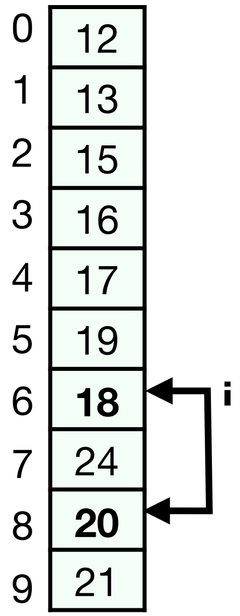
**Iteration 4, gap = 2**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

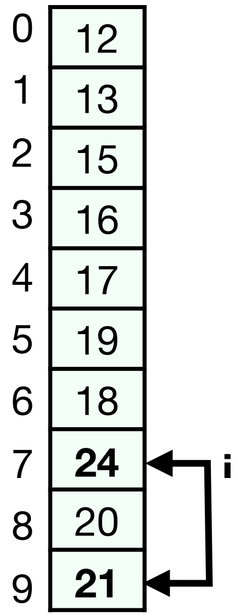
**Iteration 4, gap = 2**




```
array[i] <= array[ i + gap]
        i++;
```

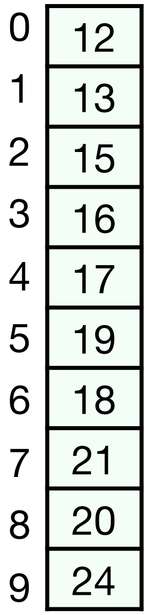
# Comb Sort

**Iteration 4, gap = 2**




array[i] > array[ i + gap]
swap and set new gap;

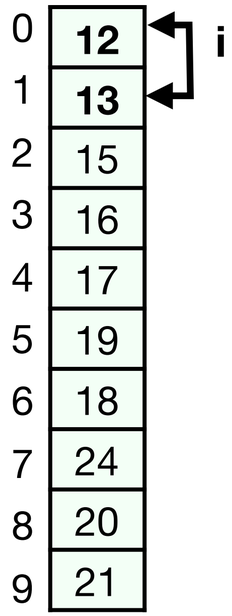
# Comb Sort




gap = n/1.3 = 2/1.3 ≈ 1

# Comb Sort

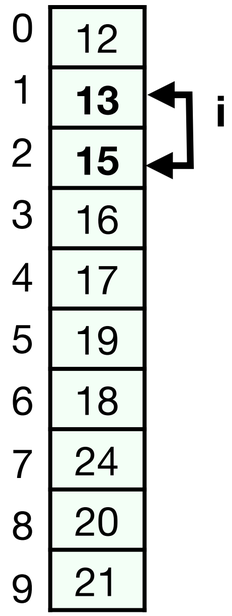
**Iteration 5, gap = 1**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

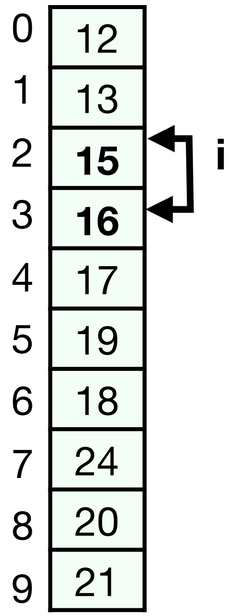
**Iteration 5, gap = 1**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

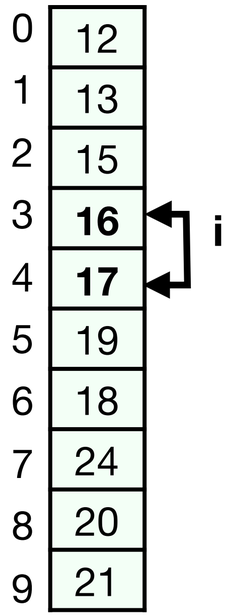
**Iteration 5, gap = 1**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

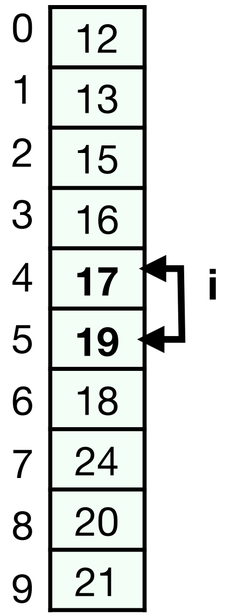
**Iteration 5, gap = 1**




```
array[i] <= array[ i + gap]
i++;
```

# Comb Sort

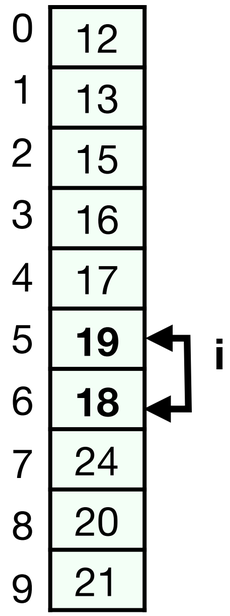
**Iteration 5, gap = 1**




```
array[i] <= array[ i + gap]
i++;
```

# Comb Sort

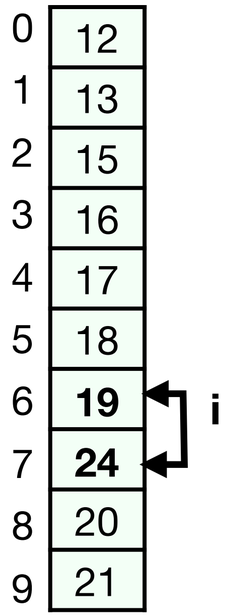
**Iteration 5, gap = 1**




array[i] > array[ i + gap]
swap and i++;

# Comb Sort

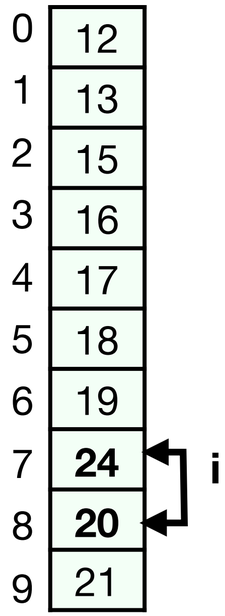
**Iteration 5, gap = 1**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

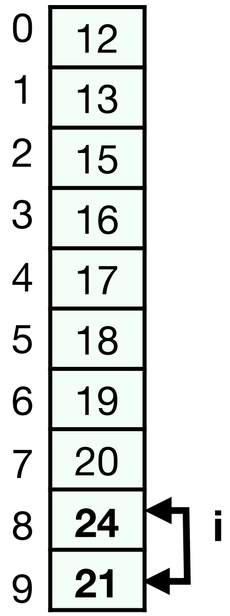
**Iteration 5, gap = 1**




array[i] > array[ i + gap]
swap and i++;

# Comb Sort

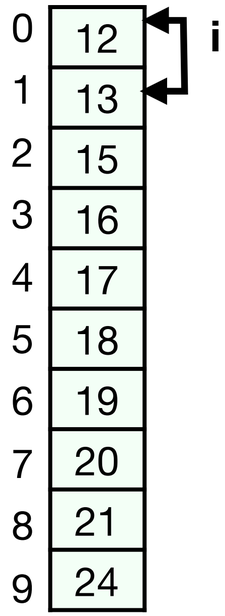
**Iteration 5, gap = 1**




array[i] > array[ i + gap]
swap;

# Comb Sort

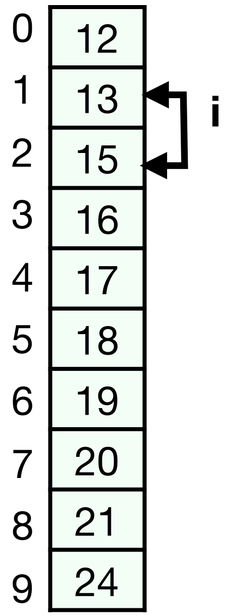
**Iteration 6, gap = 1**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

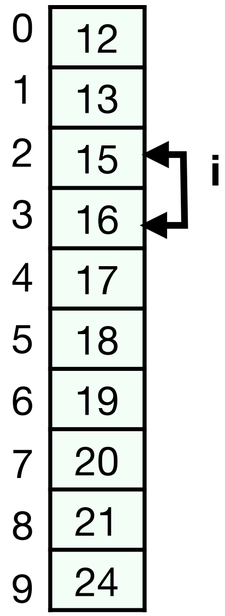
**Iteration 6, gap = 1**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

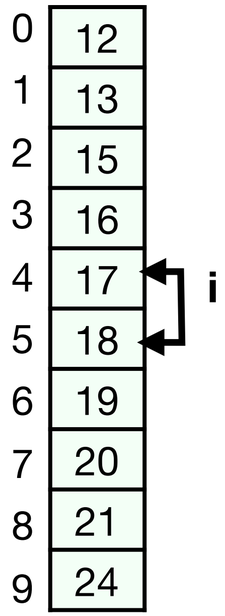
**Iteration 6, gap = 1**




```
array[i] <= array[ i + gap]
i++;
```

# Comb Sort

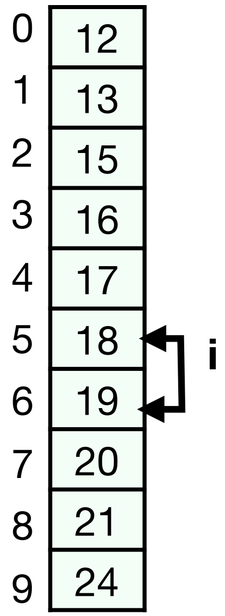
**Iteration 6, gap = 1**




```
array[i] <= array[ i + gap]
i++;
```

# Comb Sort

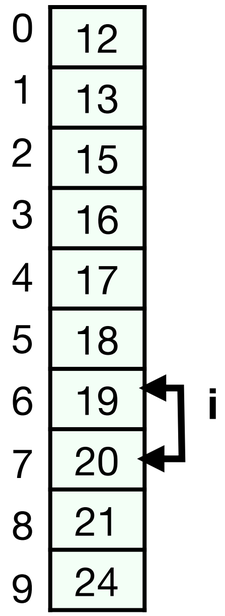
**Iteration 6, gap = 1**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

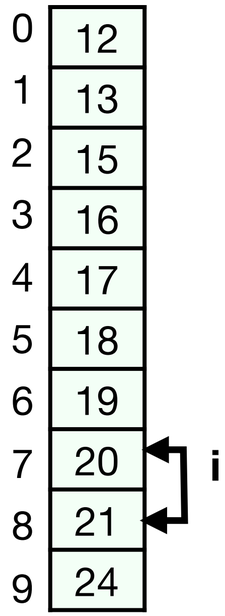
**Iteration 6, gap = 1**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

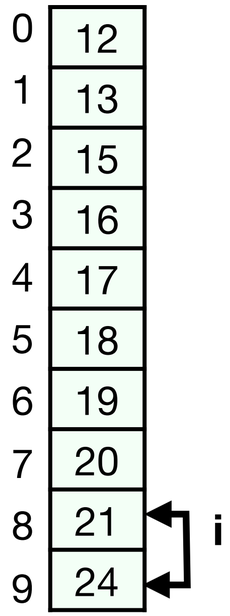
**Iteration 6, gap = 1**




```
array[i] <= array[ i + gap]
        i++;
```

# Comb Sort

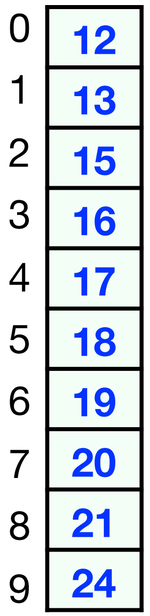
**Iteration 6, gap = 1**




array[i] <= array[ i + gap]
i++;

no more swaps and gap = 1; terminate

# Comb Sort Solution



# Reference

- https://en.wikipedia.org/wiki/Comb_sort