Implementation of Shell Sort

The shell sort improves on the insertion sort by breaking the original list into a number of smaller sublists, each of which is sorted using an insertion sort. The unique way that these sublists are chosen is the key to the shell sort. Instead of breaking the list into sublists of contiguous items, the shell sort uses an increment i, sometimes called the gap, to create a sublist by choosing all items that are i items apart.

Resources for Review

Check out the resources below for a review of Shell sort!

- Wikipedia
- Visual Algo
- Sorting Algorithms Animcation with Pseudocode

In [5]:
```
def shell_sort(arr):
    sublistcount = len(arr)//2

    # While we still have sub lists
    while sublistcount > 0:
        for start in range(sublistcount):
            # Use a gap insertion
            gap_insertion_sort(arr,start,sublistcount)



        sublistcount = sublistcount // 2

def gap_insertion_sort(arr,start,gap):
    for i in range(start+gap,len(arr),gap):

        currentvalue = arr[i]
        position = i

        # Using the Gap
        while position>=gap and arr[position-gap]>currentvalue:
            arr[position]=arr[position-gap]
            position = position-gap

        # Set current value
        arr[position]=currentvalue
```

In [6]:
```
arr = [45,67,23,45,21,24,7,2,6,4,90]
shell_sort(arr)
arr
```

Out[6]:
```
[2, 4, 6, 7, 21, 23, 24, 45, 45, 67, 90]
```

Good Job!