**Rahul Arora & Shaurya Grover**

Consider the following list of integers: [1,2,3,4,5,6,7,8,9,10]

* Sorted using bubble sort

1st pass:

[1,2,3,4,5,6,7,8,9,10] – comparing indices 0 & 1; a[0] = 1 and a[1] = 2; 1 < 2 so there’s no movement

2nd pass:

[1,2,3,4,5,6,7,8,9,10] – comparing indices 1 & 2; a[1] = 2 and a[2] = 3; 2 < 3 so there’s no movement

3rd pass:

[1,2,3,4,5,6,7,8,9,10] – comparing indices 3 & 4; a[2] = 3 and a[3] = 4; 3 < 4 so there’s no movement

4th pass:

[1,2,3,4,5,6,7,8,9,10] – comparing indices 4 & 5; a[3] = 4 and a[4] = 5; 4 < 5 so there’s no movement

5th pass:

[1,2,3,4,5,6,7,8,9,10] – comparing indices 5 & 6; a[4] = 5 and a[5] = 6; 5 < 6 so there’s no movement

6th pass:

[1,2,3,4,5,6,7,8,9,10] – comparing indices 6 & 7; a[5] = 6 and a[6] = 7; 6 < 7 so there’s no movement

7th pass:

[1,2,3,4,5,6,7,8,9,10] – comparing indices 7 & 8; a[6] = 7 and a[7] = 8; 7 < 8 so there’s no movement

8th pass:

[1,2,3,4,5,6,7,8,9,10] – comparing indices 8 & 9; a[7] = 8 and a[8] = 9; 8 < 9 so there’s no movement

9th pass:

[1,2,3,4,5,6,7,8,9,10] – comparing indices 9 & 10; a[8] = 9 and a[9] = 10; 9 < 10 so there’s no movement

* selection sort (bold indicates which index in current iteration of loop is being compared to)

[1,**2**,3,4,5,6,7,8,9,10] – finding min index, it is at index 1

[1,2,**3**,4,5,6,7,8,9,10] – finding min available index, it is at index 2

[1,2,3,**4**,5,6,7,8,9,10] – finding min available index, it is at index 3

[1,2,3,4,**5**,6,7,8,9,10] – finding min available index, it is at index 4

[1,2,3,4,5,**6**,7,8,9,10] – finding min available index, it is at index 5

[1,2,3,4,5,6,**7**,8,9,10] – finding min available index, it is at index 6

[1,2,3,4,5,6,7,**8**,9,10] – finding min available index, it is at index 7

[1,2,3,4,5,6,7,8,**9**,10] – finding min available index, it is at index 8

[1,2,3,4,5,6,7,8,9,**10**] – finding min available index, it is at index 9

[1,2,3,4,5,6,7,8,9,10] –sorted

* insertion sort

[1,**2**,3,4,5,6,7,8,9,10] – comparing a[1] to a[0], since a[1] is larger there is no movement

[1,2,**3**,4,5,6,7,8,9,10] – comparing a[2] to a[1], since a[2] is larger there is no movement

[1,2,3,**4**,5,6,7,8,9,10] – comparing a[3] to a[2], since a[3] is larger there is no movement

[1,2,3,4,**5**,6,7,8,9,10] – comparing a[4] to a[3], since a[4] is larger there is no movement

[1,2,3,4,5,**6**,7,8,9,10] – comparing a[5] to a[4], since a[5] is larger there is no movement

[1,2,3,4,5,6,**7**,8,9,10] – comparing a[6] to a[5], since a[6] is larger there is no movement

[1,2,3,4,5,6,7,**8**,9,10] – comparing a[7] to a[6], since a[7] is larger there is no movement

[1,2,3,4,5,6,7,8,**9**,10] – comparing a[8] to a[7], since a[8] is larger there is no movement

[1,2,3,4,5,6,7,8,9,**10**] – comparing a[9] to a[8], since a[9] is larger there is no movement

* shell sort (gapsize = 3) - comparing bold values

[**1**,2,3,**4**,5,6,**7**,8,9,**10**] - Comparing a[0], a[3], and a[6], and a[9]; sublist #1 is already sorted

[1,**2**,3,4,**5**,6,7,**8**,9,10] - Comparing a[1], a[4], and a[7]; sublist #2 is already sorted

[1,2,**3**,4,5,**6**,7,8,**9**,10]- Comparing a[2], a[5], and a[8]; sublist #2 is already sorted

– change gapsize to 1; array is completely sorted

[1,2,3,4,5,6,7,8,9,10]

* merge sort
* **[1,2,3,4,5,6,7,8,9,10]**
* **[1,2,3,4,5],[6,7,8,9,10]- splitting in half**
* **[1,2],[3,4],[5],[6,7],[8],[9,10] - another split**
* **[1,2,3,4],[5],[6,7,8,9],[10] - rejoining**
* **[1,2,35,4,,6,7,8,9,10] - fully joint**
* quick sort (you decide on the pivot value) – bold = pivot value , blue = left, red = right

[1,2,3,4,**5**,6,7,8,9,10] – 1 is < 5, so keep searching

[1,2,3,4,**5**,6,7,8,9,10] – 2 is < 5, so keep searching

[1,2,3,4,**5**,6,7,8,9,10] – 3 is < 5, so keep searching

[1,2,3,4,**5**,6,7,8,9,10] – 4 is < 5, so keep searching

[1,2,3,4,**5**,6,7,8,9,10] – 5=5, left value has been found

[1,2,3,4,**5**,6,7,8,9,10] – 10 > 5, so keep searching

[1,2,3,4,**5**,6,7,8,9,10] – 9 > 5, so keep searching

[1,2,3,4,**5**,6,7,8,9,10] – 8 > 5, so keep searching

[1,2,3,4,**5**,6,7,8,9,10] – 7 > 5, so keep searching

[1,2,3,4,**5**,6,7,8,9,10] – 6 > 5, so keep searching

[1,2,3,4,**5**,6,7,8,9,10] – 5=5, so right value has been found

[1,2,3,4,**5**,6,7,8,9,10] – list is sorted, right and left value equal each other, no need to repeat recursively

Consider the following list of integers: [10,9,8,7,6,5,4,3,2,1]. With pencil and paper, show how this list is sorted by the following algorithms (Attach your image here when complete):

* bubble sort

1st pass

[9, 10, 8, 7, 6, 5, 4, 3, 2, 1]

[9, 8, 10, 7, 6, 5, 4, 3, 2, 1]

[9, 8, 7, 10, 6, 5, 4, 3, 2, 1]

[9, 8, 7, 6, 10, 5, 4, 3, 2, 1]

[9, 8, 7, 6, 5, 10, 4, 3, 2, 1]

[9, 8, 7, 6, 5, 4, 10, 3, 2, 1]

[9, 8, 7, 6, 5, 4, 3, 10, 2, 1]

[9, 8, 7, 6, 5, 4, 3, 2, 10, 1]

[9, 8, 7, 6, 5, 4, 3, 2, 1, 10]

2nd pass

[8, 9, 7, 6, 5, 4, 3, 2, 1, 10]

[8, 7, 9, 6, 5, 4, 3, 2, 1, 10]

[8, 7, 6, 9, 5, 4, 3, 2, 1, 10]

[8, 7, 6, 5, 9, 4, 3, 2, 1, 10]

[8, 7, 6, 5, 4, 9, 3, 2, 1, 10]

[8, 7, 6, 5, 4, 3, 9, 2, 1, 10]

[8, 7, 6, 5, 4, 3, 2, 9, 1, 10]

[8, 7, 6, 5, 4, 3, 2, 1, 9, 10]

[8, 7, 6, 5, 4, 3, 2, 1, 9, 10]

3rd pass

[7, 8, 6, 5, 4, 3, 2, 1, 9, 10]

[7, 6, 8, 5, 4, 3, 2, 1, 9, 10]

[7, 6, 5, 8, 4, 3, 2, 1, 9, 10]

[7, 6, 5, 4, 8, 3, 2, 1, 9, 10]

[7, 6, 5, 4, 3, 8, 2, 1, 9, 10]

[7, 6, 5, 4, 3, 2, 8, 1, 9, 10]

[7, 6, 5, 4, 3, 2, 1, 8, 9, 10]

[7, 6, 5, 4, 3, 2, 1, 8, 9, 10]

[7, 6, 5, 4, 3, 2, 1, 8, 9, 10]

4th pass

[6, 7, 5, 4, 3, 2, 1, 8, 9, 10]

[6, 5, 7, 4, 3, 2, 1, 8, 9, 10]

[6, 5, 4, 7, 3, 2, 1, 8, 9, 10]

[6, 5, 4, 3, 7, 2, 1, 8, 9, 10]

[6, 5, 4, 3, 2, 7, 1, 8, 9, 10]

[6, 5, 4, 3, 2, 1, 7, 8, 9, 10]

[6, 5, 4, 3, 2, 1, 7, 8, 9, 10]

[6, 5, 4, 3, 2, 1, 7, 8, 9, 10]

[6, 5, 4, 3, 2, 1, 7, 8, 9, 10]

5th pass

[5, 6, 4, 3, 2, 1, 7, 8, 9, 10]

[5, 4, 6, 3, 2, 1, 7, 8, 9, 10]

[5, 4, 3, 6, 2, 1, 7, 8, 9, 10]

[5, 4, 3, 2, 6, 1, 7, 8, 9, 10]

[5, 4, 3, 2, 1, 6, 7, 8, 9, 10]

[5, 4, 3, 2, 1, 6, 7, 8, 9, 10]

[5, 4, 3, 2, 1, 6, 7, 8, 9, 10]

[5, 4, 3, 2, 1, 6, 7, 8, 9, 10]

[5, 4, 3, 2, 1, 6, 7, 8, 9, 10]

6th pass

[4, 5, 3, 2, 1, 6, 7, 8, 9, 10]

[4, 3, 5, 2, 1, 6, 7, 8, 9, 10]

[4, 3, 2, 5, 1, 6, 7, 8, 9, 10]

[4, 3, 2, 1, 5, 6, 7, 8, 9, 10]

[4, 3, 2, 1, 5, 6, 7, 8, 9, 10]

[4, 3, 2, 1, 5, 6, 7, 8, 9, 10]

[4, 3, 2, 1, 5, 6, 7, 8, 9, 10]

[4, 3, 2, 1, 5, 6, 7, 8, 9, 10]

[4, 3, 2, 1, 5, 6, 7, 8, 9, 10]

7th pass

[3, 4, 2, 1, 5, 6, 7, 8, 9, 10]

[3, 2, 4, 1, 5, 6, 7, 8, 9, 10]

[3, 2, 1, 4, 5, 6, 7, 8, 9, 10]

[3, 2, 1, 4, 5, 6, 7, 8, 9, 10]

[3, 2, 1, 4, 5, 6, 7, 8, 9, 10]

[3, 2, 1, 4, 5, 6, 7, 8, 9, 10]

[3, 2, 1, 4, 5, 6, 7, 8, 9, 10]

[3, 2, 1, 4, 5, 6, 7, 8, 9, 10]

[3, 2, 1, 4, 5, 6, 7, 8, 9, 10]

8th pass

[2, 3, 1, 4, 5, 6, 7, 8, 9, 10]

[2, 1, 3, 4, 5, 6, 7, 8, 9, 10]

[2, 1, 3, 4, 5, 6, 7, 8, 9, 10]

[2, 1, 3, 4, 5, 6, 7, 8, 9, 10]

[2, 1, 3, 4, 5, 6, 7, 8, 9, 10]

[2, 1, 3, 4, 5, 6, 7, 8, 9, 10]

[2, 1, 3, 4, 5, 6, 7, 8, 9, 10]

[2, 1, 3, 4, 5, 6, 7, 8, 9, 10]

[2, 1, 3, 4, 5, 6, 7, 8, 9, 10]

9th pass

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

**SORTED!**

* selection sort

[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]

[1, 9, 8, 7, 6, 5, 4, 3, 2, 10]

[1, 9, 8, 7, 6, 5, 4, 3, 2, 10]

[1, 9, 8, 7, 6, 5, 4, 3, 2, 10]

[1, 9, 8, 7, 6, 5, 4, 3, 2, 10]

[1, 9, 8, 7, 6, 5, 4, 3, 2, 10]

[1, 9, 8, 7, 6, 5, 4, 3, 2, 10]

[1, 9, 8, 7, 6, 5, 4, 3, 2, 10]

[1, 2, 8, 7, 6, 5, 4, 3, 9, 10]

[1, 2, 8, 7, 6, 5, 4, 3, 9, 10]

[1, 2, 8, 7, 6, 5, 4, 3, 9, 10]

[1, 2, 8, 7, 6, 5, 4, 3, 9, 10]

[1, 2, 8, 7, 6, 5, 4, 3, 9, 10]

[1, 2, 3, 7, 6, 5, 4, 8, 9, 10]

[1, 2, 3, 7, 6, 5, 4, 8, 9, 10]

[1, 2, 3, 7, 6, 5, 4, 8, 9, 10]

[1, 2, 3, 4, 6, 5, 7, 8, 9, 10]

* shell sort (you decide on the increments)

After each switch:

[7, 6, 5, 4, 9, 8, 10, 3, 2, 1]

[4, 6, 5, 7, 3, 8, 10, 9, 2, 1]

[4, 3, 5, 7, 6, 2, 10, 9, 8, 1]

[4, 3, 2, 7, 6, 5, 1, 9, 8, 10]

[4, 3, 2, 1, 6, 5, 7, 9, 8, 10]

[1, 3, 2, 4, 6, 5, 7, 9, 8, 10]

* quick sort (you decide on the pivot value)

After each increment:

[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]

[1, 9, 8, 7, 6, 5, 4, 3, 2, 10]

[1, 9, 8, 7, 6, 5, 4, 3, 2, 10]

[1, 2, 8, 7, 6, 5, 4, 3, 9, 10]

[1, 2, 8, 7, 6, 5, 4, 3, 9, 10]

[1, 2, 3, 7, 6, 5, 4, 8, 9, 10]

[1, 2, 3, 7, 6, 5, 4, 8, 9, 10]

[1, 2, 3, 4, 6, 5, 7, 8, 9, 10]

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]