7.1

① 3,1,4,1,5,9,2,6,5

⬇

② 1,3,4,1,5,9,2,6,5

⬇

③ 1,3,1,4,5,9,2,6,5

⬇

④ 1,1,3,4,5,9,2,6,5

⬇

⑤ 1,1,3,4,5,2,9,6,5

⬇

⑥ 1,1,3,4,2,5,9,6,5

⬇

⑦ 1,1,3,2,4,5,9,6,5

⬇

⑧ 1,1,2,3,4,5,9,6,5

⬇

⑨ 1,1,2,3,4,5,6,9,5

⬇

⑩ 1,1,2,3,4,5,6,5,9

⬇

⑪ 1,1,2,3,4,5,5,6,9

7.2.

when all elements are equal, the running time will be $O(N)$

7.4 9,8,7,6,5,4,3,2,1

    0 1 2 3 4 5 6 7 8 $(i+7)$

7: 2,1,7,6,5,4,3,9,8

⬇ $(i+3)$

3: 2,1,4,6,5,7,3,9,8

⬇

2,1,4,3,5,7,6,9,8

⬇ $(i+1)$

1: 1,2,3,4,5,6,7,8,9

7.5

a. running time will be $O(N^2)$ for shellsort for all increments

7.9

a. sorted will be best time $O(n \log n)$
b. ~~sorted~~ reverse sorted means worst time $O(N^2)$

7.10

a. subtracting one will not affect the worst case running time
b. adding one will also not affect the run time.

7.11

①   142

②   142   =>   543
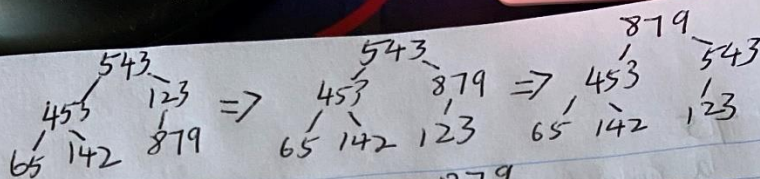   /   //
      543   142

③   543
       /
    142   123

④   543
       /
    142   123
   /
  65

⑤   543   =>   543
       /   /
   142   123   453   123
  /
65 453   65 142

⑥
```
      543              543    879 =>    879   543
453    123   =>   453    879          453    123
65 142  879       65 142 123       65 142  123
```

⑦
```
   879                   879    572
453    543    =>    453        543
65 142 123 572     65 142  123
```

⑧
```
   879                    879    572
453    572    =>     453        542
65 142 123  542     434 142 123
434                 65
```

⑨
```
   879
453       572
434 142 123    543
65 111
```

⑩
```
   879                    879    572
453    572    =>     453        543
434 142 123 543     434 242 123
65 111 242          65 111 142
```

⑪
```
   879                    879    572
453    572    =>     453        543
434 242 123 543     434 841 123
65 111 142 811      65 111 242 142
```

```
        879
811        572
434 453 123  543
65 111   242
       142
```

max heap tree.

```
        879
      811   572
   434  453  123  543
  65 111 142 242 102
```
⑫
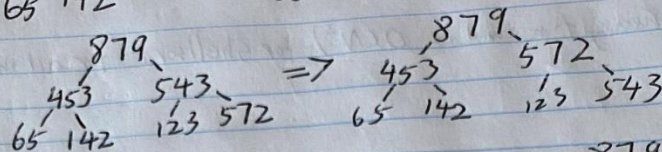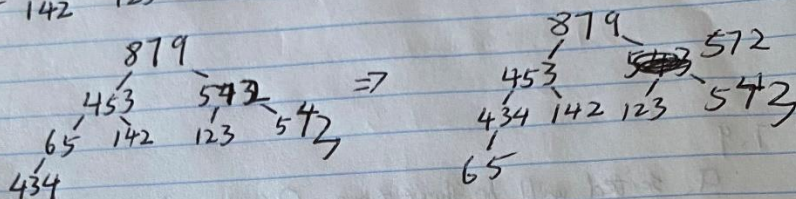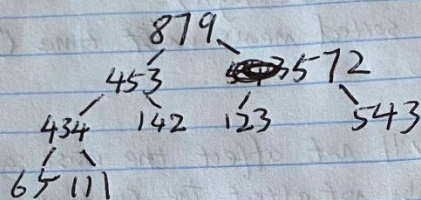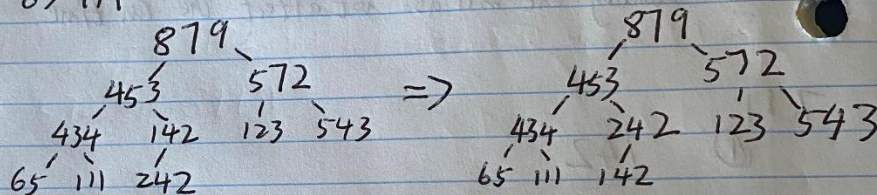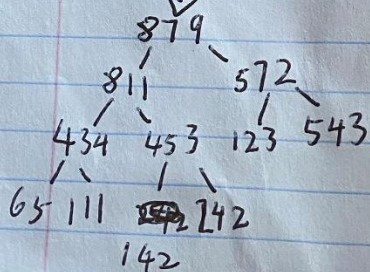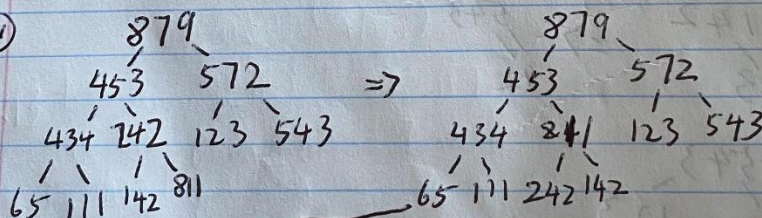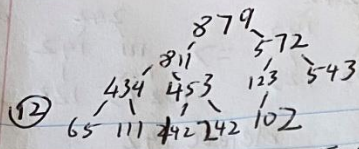
Array of Heap:

[879, 811, 572, 434, 453, 123, 543, 65,
111, 242, 142, 102]

⑬ Delete root to form new sorted array:

```
        102
     811    572
  434  453  123  543
 65 111 142 242
```
=>
```
        811
    102    572
  434 453 123 543
 65 111 142 242
```
=>
```
        811
     453    572
  434 102 123 543
 65 111 142 242
```

```
        811
     453   572
  434 242 123 543
 65 111 142 102
```

⑭
```
        102
     453    572
  434 242 123 543
 65 111 142
```
=>
```
        572
     453    102
  434 242 123 543
 65 111 142
```
=>
```
        572
     453    543
  434 242 123 102
 65 111 142
```

⑮
```
        142
     453    543
  434 242 123 102
 65 111
```
=>
```
        543
     453   142
  434 242 123 102
 65 111
```
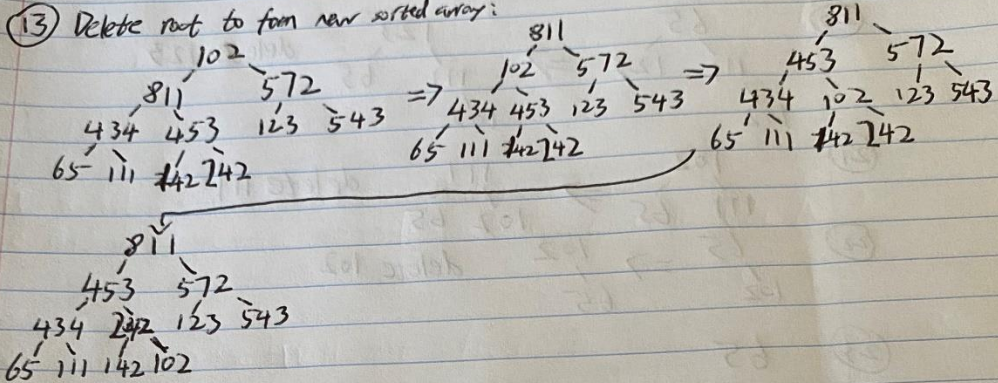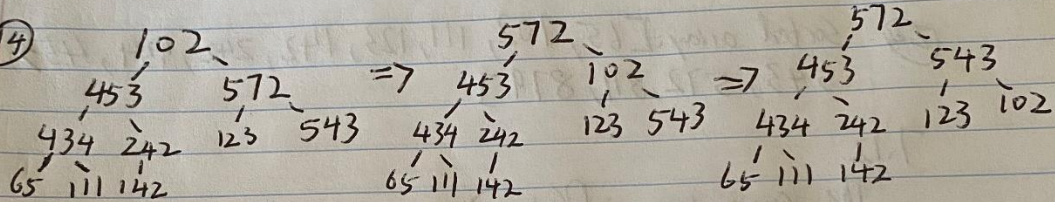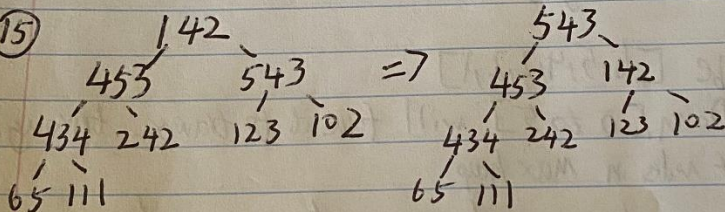
Current array = [, 572, 811, 879

⑯
```
        111
     453  142
  434 242 123 102
 65
```
=>
```
        453
     111    142
  434 242 123 102
 65
```
=>
```
        453
     434    142
  111 242 123 102
 65
```

⑰
```
        65
     434  142
 111 242 123 102
```
=>
```
        434
     65  142
 111 242 123 102
```
=>
```
        434
     242  142
 111 65 123 102
```

(18)
```
        102                      242                    242    delete
      /    \        =>        /      \     =>      /   \      242
   242     142            102      142        111     142
   /  \      \            / \       \          |        \
 111  65    123        111 65      123       102 65     123
```

(19)
```
      123                      142      delete 142
    /    \       =>          /    \
  111    142             111     123
  /  \                    /  \
102  65                 102  65
```

(20)
```
      65                    123       delete 123
    /    \      =>        /    \
  111    123           111     65
  /                      |
102                    102
```

(21)
```
     102                  111      delete 111
    /   \      =>        /   \
  111   65            102    65
```

(22)
```
    65       =>      102       delete 102
   /               /
 102             65
```

(23) 65

(24) Sorted array: [65, 102, 111, 123, 142, 242, 434, 455,
      543, 572, 811, 879]

7.12
    The running time is $O(n \log n)$

7.13 let array be [6,5,4,3,2,1]
      percolateDown [0 to 5] will force it to traverse through
      to the leaf nodes in max heap

```
  6         6         6           6            6             6
           /         / \         / \          / \           / \
          5        5   4       5   4        5   4         5   4
                          /             /   \        / \    /
                         3             3    2      3   2   1
```