HeapSort

First , we turn array to a max heap then we sort it

At the beginning we accept first element is a heap an insert other elements one by one.

After turn array to a max heap, we start beginning and use heap remove but we dont decrement size of array we act like we do

Q1:

*** A = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}**

*** B = { 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 }**

*** C = {5, 2, 13, 9, 1, 7, 6, 8, 1, 15, 4, 11}**

*** D = {'S', 'B', 'I', 'M', 'H', 'Q', 'C', 'L', 'R', 'E', 'P', 'K'}**

A) = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

Heap Sort

```
cursor: 1
 Before comparision and swap, array(try to turn it max heap) :  1 2 3 4 5 6 7 8 9 10
 compare parent: 1 and child: 2 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) :  2 1 3 4 5 6 7 8 9 10

 cursor: 2
 Before comparision and swap, array(try to turn it max heap) :  2 1 3 4 5 6 7 8 9 10
 compare parent: 2 and child: 3 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) :  3 1 2 4 5 6 7 8 9 10

 cursor: 3
 Before comparision and swap, array(try to turn it max heap) :  3 1 2 4 5 6 7 8 9 10
 compare parent: 1 and child: 4 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) :  4 3 2 1 5 6 7 8 9 10

 cursor: 4
 Before comparision and swap, array(try to turn it max heap) :  4 3 2 1 5 6 7 8 9 10
 compare parent: 3 and child: 5 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) :  5 4 2 1 3 6 7 8 9 10

 cursor: 5
 Before comparision and swap, array(try to turn it max heap) :  5 4 2 1 3 6 7 8 9 10
 compare parent: 2 and child: 6 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) :  6 4 5 1 3 2 7 8 9 10

 cursor: 6
 Before comparision and swap, array(try to turn it max heap) :  6 4 5 1 3 2 7 8 9 10
 compare parent: 5 and child: 7 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) :  7 4 6 1 3 2 5 8 9 10

 cursor: 7
 Before comparision and swap, array(try to turn it max heap) :  7 4 6 1 3 2 5 8 9 10
 compare parent: 1 and child: 8 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) :  8 7 6 4 3 2 5 1 9 10
```

```
cursor: 8
Before comparision and swap, array(try to turn it max heap) :  8 7 6 4 3 2 5 1 9 10
compare parent: 4 and child: 9 , swap them if parent<child
After comparision and swap(if it is necessary), array(try to turn it max heap) :  9 8 6 7 3 2 5 1 4 10

cursor: 9
Before comparision and swap, array(try to turn it max heap) :  9 8 6 7 3 2 5 1 4 10
compare parent: 3 and child: 10 , swap them if parent<child
After comparision and swap(if it is necessary), array(try to turn it max heap) :  10 9 6 7 8 2 5 1 4 3

After turn array to max heap :  10 9 6 7 8 2 5 1 4 3

cursor: 9
swap(heap remove at beginning) max: 10 with end of heap: 3 , decrement size of heap(cursor) by 1
After swap(heap remove) :  3 9 6 7 8 2 5 1 4 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 3 and max child:  9
After swap(parent and max child) :  9 3 6 7 8 2 5 1 4 10

swap Parent: 3 and max child:  8
After swap(parent and max child) :  9 8 6 7 3 2 5 1 4 10

cursor: 8
swap(heap remove at beginning) max: 9 with end of heap: 4 , decrement size of heap(cursor) by 1
After swap(heap remove) :  4 8 6 7 3 2 5 1 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 4 and max child:  8
After swap(parent and max child) :  8 4 6 7 3 2 5 1 9 10

swap Parent: 4 and max child:  7
After swap(parent and max child) :  8 7 6 4 3 2 5 1 9 10

cursor: 7
swap(heap remove at beginning) max: 8 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 7 6 4 3 2 5 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 1 and max child:  7
After swap(parent and max child) :  7 1 6 4 3 2 5 8 9 10

swap Parent: 1 and max child:  4
After swap(parent and max child) :  7 4 6 1 3 2 5 8 9 10

cursor: 6
swap(heap remove at beginning) max: 7 with end of heap: 5 , decrement size of heap(cursor) by 1
After swap(heap remove) :  5 4 6 1 3 2 7 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 5 and max child:  6
After swap(parent and max child) :  6 4 5 1 3 2 7 8 9 10

cursor: 5
swap(heap remove at beginning) max: 6 with end of heap: 2 , decrement size of heap(cursor) by 1
After swap(heap remove) :  2 4 5 1 3 6 7 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 2 and max child:  5
After swap(parent and max child) :  5 4 2 1 3 6 7 8 9 10

cursor: 4
swap(heap remove at beginning) max: 5 with end of heap: 3 , decrement size of heap(cursor) by 1
After swap(heap remove) :  3 4 2 1 5 6 7 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child
```

```
swap Parent: 3 and max child:  4
After swap(parent and max child) :  4 3 2 1 5 6 7 8 9 10

cursor: 3
swap(heap remove at beginning) max: 4 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 3 2 4 5 6 7 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 1 and max child:  3
After swap(parent and max child) :  3 1 2 4 5 6 7 8 9 10

cursor: 2
swap(heap remove at beginning) max: 3 with end of heap: 2 , decrement size of heap(cursor) by 1
After swap(heap remove) :  2 1 3 4 5 6 7 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

cursor: 1
swap(heap remove at beginning) max: 2 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 2 3 4 5 6 7 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

cursor: 0
swap(heap remove at beginning) max: 1 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 2 3 4 5 6 7 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

After sorted array :  1 2 3 4 5 6 7 8 9 10
```

B) = { 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 }

Heap Sort

```
cursor: 1
 Before comparision and swap, array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1
 compare parent: 10 and child: 9 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1

 cursor: 2
 Before comparision and swap, array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1
 compare parent: 10 and child: 8 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1

 cursor: 3
 Before comparision and swap, array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1
 compare parent: 9 and child: 7 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1

 cursor: 4
 Before comparision and swap, array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1
 compare parent: 9 and child: 6 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1

 cursor: 5
 Before comparision and swap, array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1
 compare parent: 8 and child: 5 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1

 cursor: 6
 Before comparision and swap, array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1
 compare parent: 8 and child: 4 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1

 cursor: 7
 Before comparision and swap, array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1
 compare parent: 7 and child: 3 , swap them if parent<child
```

```
After comparision and swap(if it is necessary), array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1

cursor: 8
Before comparision and swap, array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1
compare parent: 7 and child: 2 , swap them if parent<child
After comparision and swap(if it is necessary), array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1

cursor: 9
Before comparision and swap, array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1
compare parent: 6 and child: 1 , swap them if parent<child
After comparision and swap(if it is necessary), array(try to turn it max heap) :  10 9 8 7 6 5 4 3 2 1

After turn array to max heap :  10 9 8 7 6 5 4 3 2 1

cursor: 9
swap(heap remove at beginning) max: 10 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 9 8 7 6 5 4 3 2 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 1 and max child:  9
After swap(parent and max child) :  9 1 8 7 6 5 4 3 2 10

swap Parent: 1 and max child:  7
After swap(parent and max child) :  9 7 8 1 6 5 4 3 2 10

swap Parent: 1 and max child:  3
After swap(parent and max child) :  9 7 8 3 6 5 4 1 2 10

cursor: 8
swap(heap remove at beginning) max: 9 with end of heap: 2 , decrement size of heap(cursor) by 1
After swap(heap remove) :  2 7 8 3 6 5 4 1 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 2 and max child:  8
After swap(parent and max child) :  8 7 2 3 6 5 4 1 9 10

swap Parent: 2 and max child:  5
After swap(parent and max child) :  8 7 5 3 6 2 4 1 9 10

cursor: 7
swap(heap remove at beginning) max: 8 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 7 5 3 6 2 4 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 1 and max child:  7
After swap(parent and max child) :  7 1 5 3 6 2 4 8 9 10

swap Parent: 1 and max child:  6
After swap(parent and max child) :  7 6 5 3 1 2 4 8 9 10

cursor: 6
swap(heap remove at beginning) max: 7 with end of heap: 4 , decrement size of heap(cursor) by 1
After swap(heap remove) :  4 6 5 3 1 2 7 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 4 and max child:  6
After swap(parent and max child) :  6 4 5 3 1 2 7 8 9 10

cursor: 5
swap(heap remove at beginning) max: 6 with end of heap: 2 , decrement size of heap(cursor) by 1
After swap(heap remove) :  2 4 5 3 1 6 7 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 2 and max child:  5
After swap(parent and max child) :  5 4 2 3 1 6 7 8 9 10

cursor: 4
swap(heap remove at beginning) max: 5 with end of heap: 1 , decrement size of heap(cursor) by 1
```

```
After swap(heap remove) :  1 4 2 3 5 6 7 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 1 and max child:  4
After swap(parent and max child) :  4 1 2 3 5 6 7 8 9 10

swap Parent: 1 and max child:  3
After swap(parent and max child) :  4 3 2 1 5 6 7 8 9 10

cursor: 3
swap(heap remove at beginning) max: 4 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 3 2 4 5 6 7 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 1 and max child:  3
After swap(parent and max child) :  3 1 2 4 5 6 7 8 9 10

cursor: 2
swap(heap remove at beginning) max: 3 with end of heap: 2 , decrement size of heap(cursor) by 1
After swap(heap remove) :  2 1 3 4 5 6 7 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

cursor: 1
swap(heap remove at beginning) max: 2 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 2 3 4 5 6 7 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

cursor: 0
swap(heap remove at beginning) max: 1 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 2 3 4 5 6 7 8 9 10

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

After sorted array :  1 2 3 4 5 6 7 8 9 10
```

C) = {5, 2, 13, 9, 1, 7, 6, 8, 1, 15, 4, 11}

Heap Sort

```
cursor: 1
 Before comparision and swap, array(try to turn it max heap) : 5 2 13 9 1 7 6 8 1 15 4 11
 compare parent: 5 and child: 2 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap): 5 2 13 9 1 7 6 8 1 15 4 11

 cursor: 2
 Before comparision and swap, array(try to turn it max heap) : 5 2 13 9 1 7 6 8 1 15 4 11
 compare parent: 5 and child: 13 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap): 13 2 5 9 1 7 6 8 1 15 4 11

 cursor: 3
 Before comparision and swap, array(try to turn it max heap) : 13 2 5 9 1 7 6 8 1 15 4 11
 compare parent: 2 and child: 9 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap): 13 9 5 2 1 7 6 8 1 15 4 11

 cursor: 4
 Before comparision and swap, array(try to turn it max heap) : 13 9 5 2 1 7 6 8 1 15 4 11
 compare parent: 9 and child: 1 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap): 13 9 5 2 1 7 6 8 1 15 4 11

 cursor: 5
 Before comparision and swap, array(try to turn it max heap) : 13 9 5 2 1 7 6 8 1 15 4 11
 compare parent: 5 and child: 7 , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap): 13 9 7 2 1 5 6 8 1 15 4 11
```

```
cursor: 6
Before comparision and swap, array(try to turn it max heap) : 13 9 7 2 1 5 6 8 1 15 4 11
compare parent: 7 and child: 6 , swap them if parent<child
After comparision and swap(if it is necessary), array(try to turn it max heap): 13 9 7 2 1 5 6 8 1 15 4 11

cursor: 7
Before comparision and swap, array(try to turn it max heap) : 13 9 7 2 1 5 6 8 1 15 4 11
compare parent: 2 and child: 8 , swap them if parent<child
After comparision and swap(if it is necessary), array(try to turn it max heap): 13 9 7 8 1 5 6 2 1 15 4 11

cursor: 8
Before comparision and swap, array(try to turn it max heap) : 13 9 7 8 1 5 6 2 1 15 4 11
compare parent: 8 and child: 1 , swap them if parent<child
After comparision and swap(if it is necessary), array(try to turn it max heap): 13 9 7 8 1 5 6 2 1 15 4 11

cursor: 9
Before comparision and swap, array(try to turn it max heap) : 13 9 7 8 1 5 6 2 1 15 4 11
compare parent: 1 and child: 15 , swap them if parent<child
After comparision and swap(if it is necessary), array(try to turn it max heap): 15 13 7 8 9 5 6 2 1 1 4 11

cursor: 10
Before comparision and swap, array(try to turn it max heap) : 15 13 7 8 9 5 6 2 1 1 4 11
compare parent: 9 and child: 4 , swap them if parent<child
After comparision and swap(if it is necessary), array(try to turn it max heap): 15 13 7 8 9 5 6 2 1 1 4 11

cursor: 11
Before comparision and swap, array(try to turn it max heap) : 15 13 7 8 9 5 6 2 1 1 4 11
compare parent: 5 and child: 11 , swap them if parent<child
After comparision and swap(if it is necessary), array(try to turn it max heap): 15 13 11 8 9 7 6 2 1 1 4 5

After turn array to max heap :  15 13 11 8 9 7 6 2 1 1 4 5

cursor: 11
swap(heap remove at beginning) max: 15 with end of heap: 5 , decrement size of heap(cursor) by 1
After swap(heap remove) :  5 13 11 8 9 7 6 2 1 1 4 15

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 5 and max child:  13
After swap(parent and max child) :  13 5 11 8 9 7 6 2 1 1 4 15

swap Parent: 5 and max child:  9
After swap(parent and max child) :  13 9 11 8 5 7 6 2 1 1 4 15

cursor: 10
swap(heap remove at beginning) max: 13 with end of heap: 4 , decrement size of heap(cursor) by 1
After swap(heap remove) :  4 9 11 8 5 7 6 2 1 1 13 15

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 4 and max child:  11
After swap(parent and max child) :  11 9 4 8 5 7 6 2 1 1 13 15

swap Parent: 4 and max child:  7
After swap(parent and max child) :  11 9 7 8 5 4 6 2 1 1 13 15

cursor: 9
swap(heap remove at beginning) max: 11 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 9 7 8 5 4 6 2 1 11 13 15

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 1 and max child:  9
After swap(parent and max child) :  9 1 7 8 5 4 6 2 1 11 13 15

swap Parent: 1 and max child:  8
After swap(parent and max child) :  9 8 7 1 5 4 6 2 1 11 13 15

swap Parent: 1 and max child:  2
After swap(parent and max child) :  9 8 7 2 5 4 6 1 1 11 13 15

cursor: 8
swap(heap remove at beginning) max: 9 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 8 7 2 5 4 6 1 9 11 13 15

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child
```

```
swap Parent: 1 and max child:  8
After swap(parent and max child) :  8 1 7 2 5 4 6 1 9 11 13 15

swap Parent: 1 and max child:  5
After swap(parent and max child) :  8 5 7 2 1 4 6 1 9 11 13 15

cursor: 7
swap(heap remove at beginning) max: 8 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 5 7 2 1 4 6 8 9 11 13 15

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 1 and max child:  7
After swap(parent and max child) :  7 5 1 2 1 4 6 8 9 11 13 15

swap Parent: 1 and max child:  6
After swap(parent and max child) :  7 5 6 2 1 4 1 8 9 11 13 15

cursor: 6
swap(heap remove at beginning) max: 7 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 5 6 2 1 4 7 8 9 11 13 15

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 1 and max child:  6
After swap(parent and max child) :  6 5 1 2 1 4 7 8 9 11 13 15

swap Parent: 1 and max child:  4
After swap(parent and max child) :  6 5 4 2 1 1 7 8 9 11 13 15

cursor: 5
swap(heap remove at beginning) max: 6 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 5 4 2 1 6 7 8 9 11 13 15

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 1 and max child:  5
After swap(parent and max child) :  5 1 4 2 1 6 7 8 9 11 13 15

swap Parent: 1 and max child:  2
After swap(parent and max child) :  5 2 4 1 1 6 7 8 9 11 13 15

cursor: 4
swap(heap remove at beginning) max: 5 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 2 4 1 5 6 7 8 9 11 13 15

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 1 and max child:  4
After swap(parent and max child) :  4 2 1 1 5 6 7 8 9 11 13 15

cursor: 3
swap(heap remove at beginning) max: 4 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 2 1 4 5 6 7 8 9 11 13 15

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: 1 and max child:  2
After swap(parent and max child) :  2 1 1 4 5 6 7 8 9 11 13 15

cursor: 2
swap(heap remove at beginning) max: 2 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 1 2 4 5 6 7 8 9 11 13 15

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

cursor: 1
swap(heap remove at beginning) max: 1 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 1 2 4 5 6 7 8 9 11 13 15

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

cursor: 0
swap(heap remove at beginning) max: 1 with end of heap: 1 , decrement size of heap(cursor) by 1
After swap(heap remove) :  1 1 2 4 5 6 7 8 9 11 13 15
```

```
To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

After sorted array :  1 1 2 4 5 6 7 8 9 11 13 15
```

D) = {‘S’, ‘B’, ‘I’, ‘M’, ‘H’, ‘Q’, ‘C’, ‘L’, ‘R’, ‘E’, ‘P’, ‘K’}

Heap Sort

```
cursor: 1
 Before comparision and swap, array(try to turn it max heap) : S B I M H Q C L R E P K
 compare parent: S and child: B , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) : S B I M H Q C L R E P K

 cursor: 2
 Before comparision and swap, array(try to turn it max heap) : S B I M H Q C L R E P K
 compare parent: S and child: I , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) : S B I M H Q C L R E P K

 cursor: 3
 Before comparision and swap, array(try to turn it max heap) : S B I M H Q C L R E P K
 compare parent: B and child: M , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) : S M I B H Q C L R E P K

 cursor: 4
 Before comparision and swap, array(try to turn it max heap) : S M I B H Q C L R E P K
 compare parent: M and child: H , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) : S M I B H Q C L R E P K

 cursor: 5
 Before comparision and swap, array(try to turn it max heap) : S M I B H Q C L R E P K
 compare parent: I and child: Q , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) : S M Q B H I C L R E P K

 cursor: 6
 Before comparision and swap, array(try to turn it max heap) : S M Q B H I C L R E P K
 compare parent: Q and child: C , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) : S M Q B H I C L R E P K

 cursor: 7
 Before comparision and swap, array(try to turn it max heap) : S M Q B H I C L R E P K
 compare parent: B and child: L , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) : S M Q L H I C B R E P K

 cursor: 8
 Before comparision and swap, array(try to turn it max heap) : S M Q L H I C B R E P K
 compare parent: L and child: R , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) : S R Q M H I C B L E P K

 cursor: 9
 Before comparision and swap, array(try to turn it max heap) : S R Q M H I C B L E P K
 compare parent: H and child: E , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) : S R Q M H I C B L E P K

 cursor: 10
 Before comparision and swap, array(try to turn it max heap) : S R Q M H I C B L E P K
 compare parent: H and child: P , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) : S R Q M P I C B L E H K

 cursor: 11
 Before comparision and swap, array(try to turn it max heap) : S R Q M P I C B L E H K
 compare parent: I and child: K , swap them if parent<child
 After comparision and swap(if it is necessary), array(try to turn it max heap) : S R Q M P K C B L E H I

 After turn array to max heap :  S R Q M P K C B L E H I

 cursor: 11
 swap(heap remove at beginning) max: S with end of heap: I , decrement size of heap(cursor) by 1
```

```
After swap(heap remove) :  I R Q M P K C B L E H S

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: I and max child:  R
After swap(parent and max child) :  R I Q M P K C B L E H S

swap Parent: I and max child:  P
After swap(parent and max child) :  R P Q M I K C B L E H S

cursor: 10
swap(heap remove at beginning) max: R with end of heap: H , decrement size of heap(cursor) by 1
After swap(heap remove) :  H P Q M I K C B L E R S

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: H and max child:  Q
After swap(parent and max child) :  Q P H M I K C B L E R S

swap Parent: H and max child:  K
After swap(parent and max child) :  Q P K M I H C B L E R S

cursor: 9
swap(heap remove at beginning) max: Q with end of heap: E , decrement size of heap(cursor) by 1
After swap(heap remove) :  E P K M I H C B L Q R S

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: E and max child:  P
After swap(parent and max child) :  P E K M I H C B L Q R S

swap Parent: E and max child:  M
After swap(parent and max child) :  P M K E I H C B L Q R S

swap Parent: E and max child:  L
After swap(parent and max child) :  P M K L I H C B E Q R S

cursor: 8
swap(heap remove at beginning) max: P with end of heap: E , decrement size of heap(cursor) by 1
After swap(heap remove) :  E M K L I H C B P Q R S

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: E and max child:  M
After swap(parent and max child) :  M E K L I H C B P Q R S

swap Parent: E and max child:  L
After swap(parent and max child) :  M L K E I H C B P Q R S

cursor: 7
swap(heap remove at beginning) max: M with end of heap: B , decrement size of heap(cursor) by 1
After swap(heap remove) :  B L K E I H C M P Q R S

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: B and max child:  L
After swap(parent and max child) :  L B K E I H C M P Q R S

swap Parent: B and max child:  I
After swap(parent and max child) :  L I K E B H C M P Q R S

cursor: 6
swap(heap remove at beginning) max: L with end of heap: C , decrement size of heap(cursor) by 1
After swap(heap remove) :  C I K E B H L M P Q R S

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: C and max child:  K
After swap(parent and max child) :  K I C E B H L M P Q R S

swap Parent: C and max child:  H
After swap(parent and max child) :  K I H E B C L M P Q R S
```

```
cursor: 5
swap(heap remove at beginning) max: K with end of heap: C , decrement size of heap(cursor) by 1
After swap(heap remove) :  C I H E B K L M P Q R S

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: C and max child:  I
After swap(parent and max child) :  I C H E B K L M P Q R S

swap Parent: C and max child:  E
After swap(parent and max child) :  I E H C B K L M P Q R S

cursor: 4
swap(heap remove at beginning) max: I with end of heap: B , decrement size of heap(cursor) by 1
After swap(heap remove) :  B E H C I K L M P Q R S

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: B and max child:  H
After swap(parent and max child) :  H E B C I K L M P Q R S

cursor: 3
swap(heap remove at beginning) max: H with end of heap: C , decrement size of heap(cursor) by 1
After swap(heap remove) :  C E B H I K L M P Q R S

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: C and max child:  E
After swap(parent and max child) :  E C B H I K L M P Q R S

cursor: 2
swap(heap remove at beginning) max: E with end of heap: B , decrement size of heap(cursor) by 1
After swap(heap remove) :  B C E H I K L M P Q R S

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

swap Parent: B and max child:  C
After swap(parent and max child) :  C B E H I K L M P Q R S

cursor: 1
swap(heap remove at beginning) max: C with end of heap: B , decrement size of heap(cursor) by 1
After swap(heap remove) :  B C E H I K L M P Q R S

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

cursor: 0
swap(heap remove at beginning) max: B with end of heap: B , decrement size of heap(cursor) by 1
After swap(heap remove) :  B C E H I K L M P Q R S

To keep max heap ordered, swap head of heap until it is bigger than both child,  with max child

After sorted array :  B C E H I K L M P Q R S
```