

Bubble Sort Notes

Bubble Sort using recursion

1) $\text{arr} = [4, 3, 2, 1]$

parameters $\text{bubble}(\text{int}[] \text{arr}, \text{int } r, \text{int } c)$

$r = \text{arr.length} - 1 = 3$ } initial values
 $c = 0$

2) $\text{if } (r == 0) \{$
 $\text{return};$
}

The will only hit when the length of the array is 0

3) $\text{if } (c < r) \rightarrow$ This will run until until c is less than $\text{arr.length} - 1$ (3)

4) $\text{if } (\text{arr}[c] > \text{arr}[c+1]) \{$
 It will only run when the current item is ~~smaller~~ greater than the next item and it will swap the item :
 $\text{int temp} = \text{arr}[c]$
 $\text{arr}[c] = \text{arr}[c+1]$
 $\text{arr}[c+1] = \text{temp}$

after swapping there is a
~~recursive call~~

~~bubble(arr, r, c+1)~~

our array will look like this

arr [3, 4, 2, 1]

we swapped the position of 4 and 3
because 4 is greater than 3

Now there is a recursive call
bubble(arr, r, c+1)

arr, is same, r is also same

but we changed $c(0)$ to $c+1 = 0+1 = 1$

Now the process will run until
 $c > c+1 / \text{array of } c$

In our case of the recursive call
will run till we get an array

like: c c+1

[3, 4, 2, 1]

4 is greater than 2 swap!
and $c+1 = 1+1 = 2$

[3, 2, 4, 1]

4 is greater than 1 swap!
 $c+1 = 2+1 = 3$

[3, 2, 1, 4] we get this

Now, notice one thing our current value of $c = 3$

And look at the 3rd position where there is a condition

if ($c < r$)

Not this condition will be false and our 2nd & 4th point will not execute

→ if ($3 < 3$) which is false

Our current arr = [3, 2, 1, 4]

this is not sorted yet but notice one thing our last element is sorted (4)

Now there is a role of else part where we change some values

else {

bubble(arr, r-1, 0);

}

Now our array is same but we

change the value of our r which is referring to our length of the array

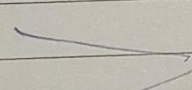
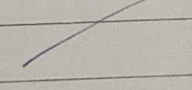
length of the array when the function is started

length = 3 and we changed the value of r which pointing to length

length = 3 , ~~as~~ $r = r - 1$

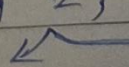
Now the of length will also change why?

2 array see na! It pointing to same object

r  [4, 3, 2, 1]
length 

Now our C which is 3 after one pass to It will become 0

and recursive call will happen ~~when~~

with parameter : `bubble(arr, 2, 0)`
It will run the function again 

Now you will have 1 doubt why we reduced the length of the array by -1

~~Ans~~ because we saw our last element is sorted.

————— 4 ————— 4 ————— 4 —————

$c \leftarrow c+1$
 $arr = [4, 3, 2, 1]$
 $r = 4 - 1 = 3$
 $r = arr.length - 1$
 $c = 0 = arr[c]$

If greater than swap

$arr = [3, 4, 2, 1]$

$arr = [3, 2, 4, 1]$

$arr = [3, 2, 4, 1]$
 $r = 3 - 1 = 2$
 sorted the last index

Now we'll reduce the r according to code

arr = [3, 2, 1, 3, 4]



arr = [2, 3, 1, 3, 4]



arr = [2, 1, 3, 4] Sorted

$$r = 2 - 1 = 1$$

$$r = r - 1$$

arr = ~~[2, 1, 3, 4]~~ [2, 1, 3, 4]



arr = [1, 2, 3, 4] Sorted

$$r = 1 - 1 = 0$$

Now we remain with only 1 element which is 1, now our r is equal to 0

now base condition will hit which is if ($r == 0$) {
return;
}

It will return to the main from where it was called

Initially it was called from main function.

Code

```
package com.nitin;
import java.util.Arrays;

public class bubble {
    public static void main(String[] args) {
        int[] arr = {4, 3, 2, 1};
        bubble(arr, arr.length - 1, 0);
        System.out.println(Arrays.toString(arr));
    }

    static void bubble(int arr[], int s, int c) {
        if (s == 0) {
            return;
        }
        if (c < s) {
            if (arr[c] > arr[c + 1]) {
                // swap
                int temp = arr[c];
                arr[c] = arr[c + 1];
                arr[c + 1] = temp;
            }
            bubble(arr, s, c + 1);
        }
        else {
            bubble(arr, s - 1, 0);
        }
    }
}
```

Keep Learning Keep Practicing 😊