Sorting Competition

Group 0

Data Representation

Takes in the data as	String[ ]
Looking two at a time like	str = String[?]
Changing them to doubles	numb = Double.parseDouble(str)

From here variables are done and the computation starts.

Algorithm

The basics:

First string 'i' is compared to the other strings 'j' in the array till it finds its spot.

This is done for every string against every other string.

Not really efficient, however it does sort in place.

I was going to switch to a better algorithm when it became almost sorted but never got to it.

Variable - 1	1n	1n
For Loop :	(n-1)	12(n-3) + 2[(n-4) + 4(n-4)] + 2[(n-4) + 1(n-4)] + 1(n-3) + 4(n-2) + 2(n-1)
Variable - 2	2(n-1)	2(n-1)
For Loop :	(n-2)	12(n-3) + 2[(n-4) + 4(n-4)] + 2[(n-4) + 1(n-4)] + 1(n-3) + 4(n-2)
If :	(n-3)	12(n-3) + 2[(n-4) + 4(n-4)] + 2[(n-4) + 1(n-4)] + 1(n-3)
Variable - 6	6(n-3)	6(n-3)
For Loop :	(n-4)	(n-4) + 4(n-4)
Variable - 4	4(n-4)	4(n-4)
For Loop :	(n-4)	
Variable - 4	4(n-4)	
Variable - 6	6(n-3)	
If :	(n-4)	(n-4) + 1(n-4)
Variable - 1	1(n-4)	1(n-4)
If :	(n-4)	
Variable - 1	1(n-4)	
Variable - 1	1(n-3)	1(n-3)
Variable - 4	4(n-2)	4(n-2)

Non-Constant Memory

Sorts array in place.	`String[ ] toSort`
Holds the two strings to be compared.	`String string = toSort[ i ]`
Contains the numbers.	`double numb = Double.parseDouble(string)`
Placeholder for indexes; Increments	`double count = 0`
Gets only the number before decimal	`String[ ] split = string.split("\\.")`