Ruby sort

To sort an array we have to compare the first two elements of the array, swap if needed and then move to the next two element. Repeat it until we have no more elements to swap

This is the solution with recursion

```
2.7.0 :090 > def bubble_sort(array, cloned_array = nil)
2.7.0 :091 >   return array if array.size <= 1
2.7.0 :092 >   cloned_array = cloned_array || array.clone
2.7.0 :093 >
2.7.0 :094 >   changed = false
2.7.0 :095 >   (cloned_array.size-1).times do |i|
2.7.0 :096 >     if cloned_array[i] > cloned_array[i+1]
2.7.0 :097 >       cloned_array[i], cloned_array[i+1] = cloned_array[i+1],
cloned_array [i]
2.7.0 :098 >       changed = true
2.7.0 :099 >     end
2.7.0 :100 >   end
2.7.0 :101 >   return cloned_array if changed == false
2.7.0 :102 >   bubble_sort(array, cloned_array)
2.7.0 :103 > end
 => :bubble_sort
2.7.0 :104 > array = [3,6,1,9,2]
2.7.0 :105 > bubble_sort(array)
 => [1, 2, 3, 6, 9]
2.7.0 :106 > array
 => [3, 6, 1, 9, 2]
```

This is the solution with a loop in the function

```
2.7.0 :107 > def bubble_sort(array)
2.7.0 :108 >   return array if array.size <= 1
2.7.0 :109 >   new_array = array.clone
2.7.0 :110 >
2.7.0 :111 >   loop do
2.7.0 :112 >     changed = false
2.7.0 :113 >     (new_array.size-1).times do |i|
2.7.0 :114 >       if new_array[i] > new_array[i+1]
2.7.0 :115 >         new_array[i], new_array[i+1] = new_array[i+1],
new_array[i]
2.7.0 :116 >         changed = true
2.7.0 :117 >       end
2.7.0 :118 >     end
2.7.0 :119 >     break if changed == false
2.7.0 :120 >   end
2.7.0 :121 >   new_array
2.7.0 :122 > end
```

```
 => :bubble_sort
2.7.0 :123 > array = [9,3,6,1,2,7]
2.7.0 :124 > bubble_sort(array)
 => [1, 2, 3, 6, 7, 9]
2.7.0 :125 > array
 => [9, 3, 6, 1, 2, 7]
```