Q1: i. Selection Sort

for i = 0 to i < N-1 do

if A[i]!=A[i-1] or A[i]!=(A[i-1)+1

min = i

for j = i+1 to j < N do

if A[j] < A[min] then

min = j

End for

temp = A[min]

A[min] = A[i]

A[i] = temp

End for

This modification will detect if the current value is already in its correct position by comparing it to the previous element, assuming the array contains only whole numbers.

For 1,2,4,3, the improved algorithm will not need to look through the rest of the array for 2, since it will now recognise that it is already placed correctly. This means that less instructions need to be executed.

ii. Bubble Sort NOT FINISHED

for i = 0 to i < N-1 do

for j = 0 to j < N-1 do

if A[j] > A[j+1]

temp = A[j]

A[j] = A[j+1]

A[j+1] = temp

End if

End for

End for

Q2: Insertion Sort

Insertion Sort works by using an area of sorted elements in the array which becomes one element larger with each iteration. Each time an iteration executes, the next unsorted value is taken and put into the correct place in the sorted area, relative to what is in the sorted area already. The existing sorted elements after the new addition are moved forward one place each to accommodate, thereby making the sorted area one element bigger.

The Big O of this algorithm is the values being completely reversed, since this will require every previous sorted element be moved each time the next element is sorted because the newest element will always be the first in the sorted list.

Source: [www.sorting-algorithms.com](http://www.sorting-algorithms.com)

Q3: Big O Cases

The worst scenario (Big O) for the iterative GCD algorithm is that the first input number, A, is smaller than the second, B. This will require swapping the two values, an operation which does not have to be performed otherwise.

The Big O for the tree drawing algorithm is when the number of lines to draw exceeds one or two. Where one and two simply require printing one or two characters, more lines requires the use of an iterative operation which is much more complex.

The Big O of the rock paper scissors algorithm is for the selection of both players repeatedly being the same. If this were to keep happening, the game could in theory never end. Each time the selections match, the iterative part of the algorithm must be performed and additional time.