Translate the following quick sort algorithm to MIPS assembly language. (YOU ARE TO USE THE MIPS CALLING CONVENTIONS WHEN IMPLEMENTING THE QUICK SORT, PARTITION, AND SWAP SUBPROGRAMS!!!)

```
main( )
begin
    quickSort(array, 0, n-1)
end main

quickSort( in out array values of n integers,
in low : integer,
in high : integer)
local variable
    pivotPosition : integer
begin
    if (low < high) then
        pivotPosition =partition(array, low, high)
        quickSort(array, low, pivotPosition-1)
        quickSort(array, pivotPosition+1, high)
    end if
end quickSort

swap( in out operand1, in out operand 2)
(pass these as addresses of integers in memory)
local variable
    temp : integer
begin
    temp = operand1
    operand1 = operand2
    operand2 = temp
end swap

function partition( in out array values of n integers,
in low, high : integer) : integer
local variables
    left, right, pivot : integer
begin
    left = low
    right = high
    pivot = array[low]
    while left < right do
        while array[right] > pivot do
            right = right - 1
        end while
        while (left < right) and (array[left] <= pivot) do
            left = left + 1
        end while
        if left < right then
            swap(array[left], array[right])
        end if
    end while
    array[low] = array[right]
    array[right] = pivot
    return right
end partition
```

Turn in the following:
1) The MIPS assembly language program -- include comments describing which registers are being used for parameters and local variables
2) The "output" of the MIPS program using the following 10 integers for your initial data: 8 2 7 1 5 6 3 4 9 0 (I want a picture of MIPS simulator including the sorted array in memory after the program has run).