Algorithm 1 QuickSort Algorithm

```
procedure QuickSort($A, low, high$)
    if $low < high$ then
        $pivot \leftarrow$ Partition($A, low, high$)
        QuickSort($A, low, pivot - 1$)
        QuickSort($A, pivot + 1, high$)
    end if
end procedure

function Partition($A, low, high$)
    $pivot \leftarrow A[high]$
    $i \leftarrow low - 1$
    for $j \leftarrow low$ to $high - 1$ do
        if $A[j] \leq pivot$ then
            $i \leftarrow i + 1$
            Swap $A[i]$ with $A[j]$
        end if
    end for
    Swap $A[i + 1]$ with $A[high]$ return $i + 1$
end function
```

Algorithm 2 Simplex Algorithm

```
procedure Simplex($A, b, c$)
    Initialize a feasible basis $B$
    while there exists a non-basic variable with a positive coefficient in the objective function do
        Choose entering variable $x_e$
        Determine leaving variable $x_l$ using the minimum ratio test
        if no leaving variable can be found then
            return ”Unbounded”
        end if
        Pivot on entry corresponding to $x_e$ and $x_l$
        Update $B$ to new basis
    end whilereturn ”Optimal solution found”
end procedure
```

Algorithm 3 QuickSort Algorithm

```
procedure QuickSort($A, low, high$)
    if $low < high$ then
        $pivot \leftarrow$ Partition($A, low, high$)
        QuickSort($A, low, pivot - 1$)
        QuickSort($A, pivot + 1, high$)
    end if
end procedure

function Partition($A, low, high$)
    $pivot \leftarrow A[high]$
    $i \leftarrow low - 1$
    for $j \leftarrow low$ to $high - 1$ do
        if $A[j] \leq pivot$ then
            $i \leftarrow i + 1$
            Swap $A[i]$ with $A[j]$
        end if
    end for
    Swap $A[i + 1]$ with $A[high]$ return $i + 1$
end function
```