COE 205, Term 061

Computer Organization & Assembly Programming

Programming Assignment# 2
Due date: Tuesday, Dec. 5, 2006

Q.1. Quicksort is a sorting algorithm whose worst-case running time is $O(n^2)$ on an input array of n numbers. In spite of this slow worst-case running time, quicksort is often the best practical choice for sorting because it is remarkably efficient on the average: its expected running time is O(n lg n). Quicksort is based on divide-and conquer paradigm. The three-step divide-and-conquer process for sorting a typical array A[p...r] is as follows:

1. **Divide**: The array A[p..r] is partitioned (rearranged) into two nonempty subarrays A[p..q] and A[q+1..r] such that each element of A[p..q] is less than or equal to each element of A[q+1..r]. The index q is computed as part of this partitioning procedure.

2. **Conquer**: The two subarrays A[p..q] and A[q+1..r] are sorted by recursive calls to quicksort.

3. **Combine**: Since the subarrays are sorted in place, no work is needed to combine them: the entire array A[p..r] is now sorted.

The following procedure implements quicksort:

```
QUICKSORT(A, p, r)
1  if ( p < r) Then
2     q ← PARTITION(A, p , r)
3    QUICKSORT(A,  p , q)
4    QUCIKSORT(A, q+1, r)
```

To sort an entire array A, the initial call is QUICKSORT(A, 1, length[A]).

The partition procedure is defined as follows:

```
PARTITION(A, p, r)
1. x ← A[p]
2. i ←  p-1
3. j ←  r+1
4. While TRUE
5.       Repeat
6.            j ←  j-1
7.       Until A[j] ≤ x
8.        Repeat
9.              i ←  i+1
```

```
10.     Until A[i] ≥ x
11.     If (i < j) Then
12.        exchange A[i] ↔A[j]
13.     Else
14.        return j
```

(i) Write a procedure, PARTITION, to implement the partition procedure given above.

(ii) Write a procedure, QUICKSORT, to implement the quicksort procedure given above.

(iii) Ask the user to enter an array of integers (i.e. signed numbers) and store the entered integers in an array called SCORES defined as an array of double words.

(iv) Use the QUICKSORT procedure you implemented to sort the entered array of integers.

(v) Display the entered Array of integers before sorting and after sorting.

(vi) Display the execution time of your procedure in seconds.

A sample execution of the program is shown below:

```
Enter an array of integers:
20
-100
30
15
60

Your entered array before sorting is:
20 -100 30 15 60

Your entered array after sorting is:
-100 15 20 30 60

Program execution time is 2 seconds.
```

The solution should be well organized and your program should be well documented. Submit a soft copy of your solution in a zip file. The soft copy should include a Readme file indicating the file names containing the solution and whether it works or not. The Readme file should also contain your name and ID. Submit both source code file (i.e. .asm) and the executable file (i.e. .exe).