Algorithm Design and Analysis
Review

Tuesday, March 3, 2009

1. For the following program, what is the order of the running time? For an input value of N, what value does func1 return?

```
int func1(int n)
{
  int a, i, j;
  a = 0;
  for (i = 0 ; i < n ; i++)
  {
    for (j = i ; j >= 0 ; j--)
    {
      a++;
    }
  }
  return a;
}
```

2. What is the *worst-case* order of each of the following sorting algorithms?

Insertion Sort_____

Bubble Sort _____

Merge Sort_____

Quick Sort _____

Heap Sort_____

Radix Sort_____

Useful Recurrences and their running time:

`T(n) = T(n/2) + O(1)`	$O(\log n)$
`T(n) = T(n-1) + O(1)`	$O(n)$
`T(n) = 2 T(n/2) + O(1)`	$O(n)$
`T(n) = T(n-1) + O(n)`	$O(n^2)$
`T(n) = 2 T(n/2) + O(n)`	$O(n \log n)$

3. Here's an algorithm:

```
int mystery(int n)
{
  int s=0;
  if (n = 0)
   return s;
  else
   return n + mystery(n-1);
}
```

a) What does it compute?

b) Write the recurrence that describes the running time of this algorithm, and solve it.

c) Write an algorithm that accomplishes the same task as the one above, but does not use recursion. Make sure the running time of your algorithm is at least as good or better

4. Let A[1..n] be an array of n distinct numbers. If i < j and A[i] > A[j] then the pair (i, j) is called an inversion. For example, the array <2, 3, 8, 6, 1> contains five inversions. Write an algorithm INVERSIONS(A, n) that determines the number of inversions in A[1..n]. Give the running time of your algorithm.

5. Professor Lupin is applying for tenure. In his application he sites the following algorithm as one of his greatest achievements. A is an array, p is the index of the first element, and n is the number of elements in the array.

```
LUPIN( A, p, n )
 if (n <= 2)
   then return
 if (A[1] > A[n-1])
   then exchange A[1] with A[n]
 LUPIN(A, 2, n-2)
 if (A[1] > A[2])
   then exchange A[1] with A[2]
 LUPIN(A, 2, n-1)
```

(a) Write down a recurrence describing the number of times two members of array A are compared, measured as a function of the array length n.

(b) Find the running time of algorithm LUPIN by solving the recurrence in (a)

(c) Which problem is algorithm LUPIN solving?

(d) Can you solve the same problem more efficiently? How?

(e) Would you recommend Prof. LUPIN for tenure?

6. What are the running times of HEAPSORT and QUICKSORT. Which would be most appropriate to use if most of the numbers are sorted? Why? Discuss your answer.

7. For a sorted array A[1..n], give and justify expressions for worst-case complexity (O) for:

(a) Inserting an element into A

(b) Deleting an element from A

(c) Determining A[i] when i is given

(d) Determining i when A[i] is given