CS200 Spring 2015 Recitation 9: Recurrences

name:

id:

1. Using the Master Theorem:

Let f be an increasing function that satisfies
$f(n) = a \cdot f(n/b) + c \cdot n^d$
whenever $n = b^k$, where k is a positive integer, $a \geq 1$, b is an integer > 1, and c and d are real numbers with c positive and d nonnegative. Then

$$f(n) = \begin{cases} O(n^d) & \text{if } a < b^d \\ O(n^d \log n) & \text{if } a = b^d \\ O(n^{\log_b a}) & \text{if } a > b^d \end{cases}$$

What are the big-O bounds recurrence relations?

a) f(n) = 3f(n/3) + 5n

b) f(n) = 16f(n/4) + 8

c) f(n) = f(n/2) + n^3

d) f(n) = 2 f(n/2) + 1

2. Find a solution to the following recurrence relation, using repeated substitution:

```
f(1) = 1
f(n) = 4 f(n-1)  for n>1
```

3. Given the following method:

```
public int recSrch (int[] A, int k){
     return recSrch(A,k,0,A.length-1);
}
private int recSrch(int[]A, int k, int lo, int hi){
     if(lo>hi) return -1;
     else{
          int mid = (lo+hi)/2;
          if(k==A[mid]) return mid;
          int i1 = recSrch(A,k,lo,mid-1);
          int i2 = recSrch(A,k,mid+1,hi);
          return Math.max(i1,i2);
     }
}
```

a) Derive a recurrence rS(n) relation for recSrch(A, k), where n is the length of A.

rS(n) = 1 for n = 1

rS(n) = for n > 1

b) Use the Master Theorem to solve the recurrence.