**Question 1**

This question will give you further practice with the Master Method. Suppose the running time of an algorithm is governed by the recurrence T(n)=7∗T(n/3)+n2. What's the overall asymptotic running time (i.e., the value of T(n))? Note: If you take this quiz multiple times, you may see different variations of this question.

θ(n2log⁡n)

θ(n2)

θ(n2.81)

θ(nlog⁡n)

**Question 2**

Consider the following pseudocode for calculating ab (where a and b are positive integers)

FastPower(a,b) :

if b = 1

return a

otherwise

c := a\*a

ans := FastPower(c,[b/2])

if b is odd

return a\*ans

otherwise return ans

end

Here [x] denotes the floor function, that is, the largest integer less than or equal to x.   
Now assuming that you use a calculator that supports multiplication and division (i.e., you can do multiplications and divisions in constant time), what would be the overall asymptotic running time of the above algorithm (as a function of b)?

Θ(blog⁡(b))

Θ(b)

Θ(log⁡(b))

Θ(b)

**Question 3**

Let 0<α<.5 be some constant (independent of the input array length n). Recall the Partition subroutine employed by the QuickSort algorithm, as explained in lecture. What is the probability that, with a randomly chosen pivot element, the Partition subroutine produces a split in which the size of the smaller of the two subarrays is ≥α times the size of the original array?

1−α

2−2∗α

1−2∗α

α

**Question 4**

Now assume that you achieve the approximately balanced splits above in every recursive call --- that is, assume that whenever a recursive call is given an array of length k, then each of its two recursive calls is passed a subarray with length between αk and (1−α)k (where α is a fixed constant strictly between 0 and .5). How many recursive calls can occur before you hit the base case? Equivalently, which levels of the recursion tree can contain leaves? Express your answer as a range of possible numbers d, from the minimum to the maximum number of recursive calls that might be needed.

−log(n)/log(α) ≤ d ≤ −log(n)/log(1−α)

−log(n)/log(1−2∗α) ≤ d ≤ −log(n)/log(1−α)

0 ≤ d ≤ −log(n)/log (α)

−log(n)/log(1−α) ≤ d ≤ −log(n)/log(α)

**Question 5**

Define the recursion depth of QuickSort to be the maximum number of successive recursive calls before it hits the base case --- equivalently, the number of the last level of the corresponding recursion tree. Note that the recursion depth is a random variable, which depends on which pivots get chosen. What is the minimum-possible and maximum-possible recursion depth of QuickSort, respectively?

Minimum: Θ(log⁡(n)) ; Maximum: Θ(n)

Minimum: Θ(log⁡(n)) ; Maximum: Θ(nlog⁡(n))

Minimum: Θ(1) ; Maximum: Θ(n)

Minimum: Θ(n) ; Maximum: Θ(n)