**Q1)**

log\* n is an iterative algorithm until result is less than or equal to 1:

log\* n = 1 + log\*(log n)

If *log\* n=x*

lg(lg\**n*) =lg*x*

from the algorithm: lg\*(lgn)=lg\*n-1

so, lg\*(lgn)=x-1

x-1>lgx

Therefore, lg\*(lgn) is asymptotically larger.

lg\*(lgn) > lg(lg\**n*)

**Q2)** The code is in java file.

Basically, the purpose of the algorithm is that if the next element in the array is greater than the previous max-sum when you add it, the max-sum is updated.

If the new total is not greater than max-sum, a temporary total is kept so that the probability of it being greater than max-sum in the future is preserved.

**Q3)**

Assume T(n) <= cnlog34

T(n)=4T(n/3)+n

By substitution 4T(n/3)+n < 4c(n/3)log34 +n

4T(n/3)+n < 4c(n log34/4) +n

4T(n/3)+n < c\*n log34+ n

Since there is extra n in the equation we cannot prove our assumption in that form.

Now subtract lower-order term d\*n

T(n) <= cnlog34-dn

4T(n/3)+n <= 4[c(n/3)log34 –(d\*n/3)] +n

4T(n/3)+n <= 4[c(nlog34/4)–(d\*n/3)] +n

<= c\*nlog34-d\*n-dn/3+n

<= c\*nlog34-d\*n-(d/3-1)\*n <= c\*nlog34-d\*n

Now substitution holds for the assumption.