1.

a.

f(n) = log3(n3) = 3log3(n)

g(n) = n

We need to prove

f(n) <= Cg(n) for n>=n0 . Then:

3log3(n) / n <= C

=> choose no = 1 and C = 3

3log3(n) / n <= 3 with n >= 1

True

b.

f(n) = 3n

g(n) = nk with k >= 1

Assume 3n >= Cnk then:

3n / nk >= C

The left side has no limit with any constant k when n to infinity.

False

3.

a. T (n) = 4T (n/2) + 1

According to Master Theorem, a=4, b=2, f(n)=1

n^(log2(4) - *ε*) = n^0

=> case 1

*ε* = 2

T (n) = Θ(n2)

True

b. T (n) = 3T (n/5) + 3

According to Master Theorem, a=3, b=5, f(n)=3

n^(log5(3) - *ε*) = n^0

=> case 1

*ε* = log5(3)

T (n) = Θ(n^(log5(3)))

True

4.

a.

f(n) = n2 − 6n + 3

g(n) = n

We need to prove

f(n) <= Cg(n) for n>=n0 . Then:

n – 6 + 3/n <= C

no exist constant C with all n >= 1

False

b.

f(n) = 5n

g(n) = 4n

We need to prove

f(n) <= Cg(n) for n>=n0 . Then:

(5/4)^n <= C

=> The left side has no limit when n to infinity => no exist constant C with all n >= 1

False

c.

f(n) = 5n(log(n))^3

g(n) = n^(4/3)

We need to prove

f(n) <= Cg(n) for n>=n0 . Then:

5n(log(n))^3 / n^(4/3) <= C

<=> 5 log(n)^3 / n^(1/3) <= C

<=> 5 log(n)^3 / (n^(1/9))^3 <= C

compare n and 2^(n1/9), the left side doesn’t increase speed as the right side

So the statement is true

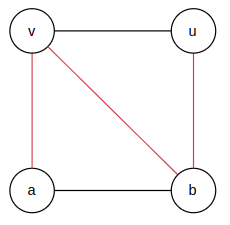
5.

In order to check “A is an integer array with n elements” statement, we can iterate all elements in A to check whether all elements are integer and exist integer x in A elements.

It can be run in O(n) with n is size of A. It runs in polynomial time, therefore this P is in NP.

6.

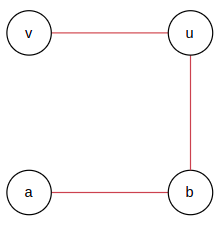
a. Since T is MST, so there is a unique path (shortest path) from u → v with u,v in T and u,v ∈ V



In the image above, The MST is a-v-b-u, there is only path from u->v

b. Since the edges have the same weight, so the MST T is not unique, we can form another MST T’

In the image above we can form a MST like this:



7.

We simply calculate n clauses with certificate C, then calculate the formula the formed from clauses.

Loop calculate 1 clause: k

Loop calculate n clauses: n

O(nk), run in polynomial time.

8.

This fasted algorithm running on O(n) if x is always the first element if B.

The efficient time complexity is O(nlogm) in case x is not in B, then we can find it by binary search tree since B is a sorted array, so to find x is in B it takes O(log(m)) time

2.

a. Guess *T*(*n*) ≤ *k*n−*b* for some two constants *k* and *b*

T(n) = 6T (n/7) + 4n

Inductive case: We assume our property is true for n-1. We now want to show that it is true for n

T(n) = 6T (n/7) + 4n <= 6(*kn/7*−*b) + 4n <= kn – b*

*This is true as long as*

Diagram

Description automatically generated9.

10.

Shortest paths between vertex A and all others

A -> A = 0

A-> E -> B = 3

A -> C = 7

A -> E -> D = 4

A-> E = 2

A -> F = 4

A -> F -> G =5

