1.

a. f(n) = 5n^2 + 6n^(3/2)

g(n) = n^2

We need to prove

f(n) <= Cn^2 for n >= n0 , then:

5 + 6n^(-1/2) <= C

=> choose C = 11, n0 = 1

f(n) / g(n) = (5n^2 + 6n^(3/2)) / n^2 <= 11

true

b. f(n) = 2n + 1

g(n) = n

We need to prove

f(n) <= Cn for n >= Cn0, then

2 + 1/n <= C

=> Choose n0=1, C=3

f(n) / g(n) = (2n + 1) / n <= 3

true

c. f(n) = n^2 -5n

g(n) = n

There are n0 and C satisfy:

f(n) / g(n) <= C

=> n - 5 <= C

for some large n, there's no constant C satisfy.

so it's not O(n)

true

d. f(n) = (n-3)^2

g(n) = n

Assume the statement is true, then there are n0 and C satisfy:

f(n) <= Cn for n >= Cn0, then

n - 6 + 9/n <= C

for some large n, there's no constant C satisfy.

so it's not O(n)

the statment is false

2.

a. T(n) = 3T(n/3) + 1

prove that T(n) prove T(n) <= cn

Assume a bound hold for n/3

T(n) <= cn/3 <= cn for some large n

c > 1, we done

b.

T(n) = T(n-1) + 2n

Guess T(n) = n

there exists a c and n0 such that

T(n) >= cn for n > n0 and c > 0

if n=1 then T(1) = 1 > c1lg(1) or all c>0

T(n) = T(n-1) + 2n >= c(n-1)

=> T(n) = cΩ(n)

3.

a.

a=7, b=3, f(n) = n^2

n^(log3(7)) < n^2

=> case 2

=> T(n) is Θ(n^2)

b. a=3, b=3, f(n) = n^(1/2)

n^(log3(3)) > n^(1/2)

=> case 3

T(n) is Θ(n^(log3(3))) = Θ(n)

5.

a.

max = - INT\_MAX

finalmax = 0

for i = 0 to n

finalmax = finalmax+a[i]

max = max(max, finalmax)

if (finalmax < 0)

finalmax = 0

print finalmax

complexity: O(n)

b.

cursum = array[0]

for i=0 to n-1:

cursum = max(array[i], cursum + array[i+1])

maxsum = max(maxsum, cursum)

print maxsum

time complexity: O(n)