Question 1 Output:
==========
B) Number of digits in the binary representation of 256: 9
B) Number of digits in the binary representation of 750: 10
==========
C) T(n) = T(n/2) + 1
==========
D)
a = 1
b = 2
f(n) = 1
log_b(a) = log_2(1) = 0
f(n) = 1 > log_2(1), So we have case 2
T(n) = Θ(n^log_b(a) * log n) = Θ(log n)
Big-oh of Θ(log n)

Process finished with exit code 0

Question 2 Output:
==========
B) The sum of squares for 12 is: 650
B) The sum of squares for 20 is: 2870
==========
C) T(n) = T(n - 1) + 1
==========
D)
T(n) = T(n-1)+1
T(n-1) = T[(n-1)-1]+1 = T[(n-2)+1]+1 = T(n-2)+2
T(n-2) = T[(n-2)-1]+1 = T[(n-3)+1]+2 = T(n-3)+3
T(n-3) = T[(n-3)-1]+1 = T[(n-4)+1]+3 = T(n-4)+4
T(n) = T(n-k)+k
assume n-k=0 so n=k
so
T(n) = T(n-n)+n
T(n) = T(0)+n
T(n) = 1+n
Big-oh of n or O(n)

Process finished with exit code 0