1b) Given:
print(binary_digits(256))
print(binary_digits(750))

OUTPUT:
9
10

1c) $f(0) = 1$ and $f(1) = 1$
$n = 2 = 1 + f(1) = 2$
$n = 4 = 1 + f(2) = 3$
$n = 8 = 1 + f(4) = 4$
$n = 16 = 1 + f(8) = 5$

We can see that we need to double the previous n to increase to the next digit in binary.
Recurrence relation would be:

$f(n) = 1 + f(n/2)$

1d) Using the Master Method:
$f(n) = aT(n/b) + n^d$
$f(n) = 1T(n/2) + n^0$

$a = 1, b = 2, d = 0$

$1 < 2^0$? FALSE
$1 = 2^0$? TRUE

$O(n^0 \times \log(n) = O(\log(n))$

ANSWER: $O(\log(n))$

2b) Given:
print(square_sum(12))
print(square_sum(20))

OUTPUT:
650
2870

2c) $f(1) = 1$
$n = 2 = f(1) + 2^2 = 5$
$n = 3 = f(2) + 3^2 = 14$
$n = 4 = f(3) + 4^2 = 30$

We can see that we are calling the current number n^2 and adding the previous nth term sum to it (n - 1) which gives us the recurrence relation:

$$\mathbf{f(n) = n^2 + f(n - 1)}$$

2d) Using back-substitution:

$f(n) = n^2 + f(n - 1)$
$= n^2 + ((n - 1)^2 + f(n - 2))$
$= n^2 + (n^2 - 2n + 1 + f(n - 2))$
$= 2n^2 - 2n + 1 + f(n - 2)$
$= 2n^2 - 2n + 1 + ((n - 2)^2 + f(n - 3))$
$= 2n^2 - 2n + 1 + (n^2 - 4n + 4 + f(n - 3))$
$= 3n^2 - 6n + 5 + f(n - 3)$
…
ANSWER: $O(n^2)$