ADS 5

P2. a)

we have a and b 2 large integers of n bits
in order to brute force the multiplication of both
w multiply each bit of a with b an shift a
the add the result iterating trough ht each bit of b

we multiply n bits of a with b and shift a
in this case 2n which takes $O(n)$
we add the sifted result to the previous sum
which takes $O(n)$.
so the total time complexity is $\Theta(n^2)$

b) we can represent the large number in
binary and separate them into 2 subparts
in this case knowing that the number of
bits of a is a multiple of 2 we can
write it as follow:

$a = a_1 \cdot 2^{n/2} + a_2$

where a_1 is the higher $n/2$ bits

and a_2 the lower $n/2$ bits

So multiplying both gives us

$$(a_1 \cdot 2^{n/2} + a_2) \cdot b = a_1 \cdot b \cdot 2^{n/2} + a_2 b$$

so we are left with 2 subproblems of size $n/2$

the recurrence for that is

c) $2T(n/2) + O(n)$

$T_n = 2T(n/2) + O(n)$

d) we can expand this function leaving us with:

$$T(n) = 2_1 T(n/4) + O(n) + O(n) = 2^2 T(n/2^2) + 2O(n)$$

$$\Rightarrow T(n) = 2^k T(n/2^k) + k\,O(n)$$

$$T(n) = 2^{\log_2 n} T(1) + \log_2(n)\, O_n$$

$$(=) \; T(n) = O(n) + O(n \log n) = \boxed{O(n \log n)}$$

because $2^{\log_2 n} = n$

e) $T(n) = \underset{a}{2}T\left(n/\underset{b}{2}\right) + \underset{f_n}{O(n)}$

Using Master's method:

$f_{n^{\log_b a}} = O(n) = O(n^{\log_2 2}) = O(n)$

$\Rightarrow T(n) = \Theta(n^{\log_2 2} \log n) = \Theta(n \log n)$

which matches with (d)