b) power

- The loop runs for 'exponent' times
- each iteration involves a simple multiplication

∴ $\Theta(\text{exponent})$

power2

- Recurrence relation of power2 (BigInteger base, int exponent)

let exponent = n

$$T(n) = \begin{cases} \Theta(1) & \text{if } n=0 \\ T\left(\frac{n}{2}\right) + \Theta(1) & \text{otherwise} \end{cases}$$

using the master theorem

$a = 1$

$b = 2$

Cost of leaves $= n^{\log_b(a)}$

$= n^{\log_2(1)} = n^0 = 1 \longrightarrow C$

Cost of root $= C$

this is case 2 of Master theorem

Since $C = \Theta(1)$

Hence $T(n) = \Theta(\log n)$

Recurrance relation of findPairs.--- (int [] arr, int target Sum)

let n = arr.length

$$T(n) = \begin{cases} \Theta(1) & \text{if } n=1 \\ 2T\left(\frac{n}{2}\right) + \Theta(n) + \Theta(n) & \text{otherwise} \end{cases}$$

using the master theorem

$a=2$

$b=2$

Cost of leaves $= n^{\log_b(a)} = n^{\log_2(2)} = n$

Cost of root $= n$

this is Case 2 of M.T $\therefore T(n) = \Theta(n \log n)$

2. Recurrance relation of BinarySearch

$a=1$

$b=2$

Cost of leaves $= n^{\log_b(a)} = n^{\log_2(1)} = n^0 = 1$

$\rightarrow C$

Cost of root = 1 $\rightarrow c$

case 2 of M.T Hence $T(n) = \Theta \log(n)$

3. Finally $T(n)$ of findPairSum

let n = arr. length

$T(n) = n \log n + \log n = \Theta(n \log n)$

↗ higher order term

d) experimental results do confirm the theoretical results.