PW SKILLS

$4^0 = 1$ $2^0 = 1$

(2) $b = 0$

↳ return 1;

$a = 2$

$b = 3$

Power of an element

↳ $a^b = 2^3 = 8$

Recursive code

Base case condition

(1) $b = 1$

↳ return a;

(2) Recursive calls

Brute force Approach

```
Powerfind(a, b) {
    base case condition
    if(b == 0) {
        return 1;
    }
    else {
        result = a * Powerfind(a, b-1);
    }
    return result;
}
```

a = 2 | b = 3

$a^b = 2^3 = 8$

8

Powerfind(2,3)

4 = 8

2 * Powerfind(2,2)

2 = 4

2 * Powerfind(2,1)

1

2 * Powerfind(2,0)

Time complexity

↳ O(b)

Efficient algorithm

a = 2

b = 64 { Even, odd

$2^{64} = 2^{32} * 2^{32}$

4,29,49,67,296

64

65536 * 65536

2^{32} 2^{32}

65536

2^{16} 2^{16}

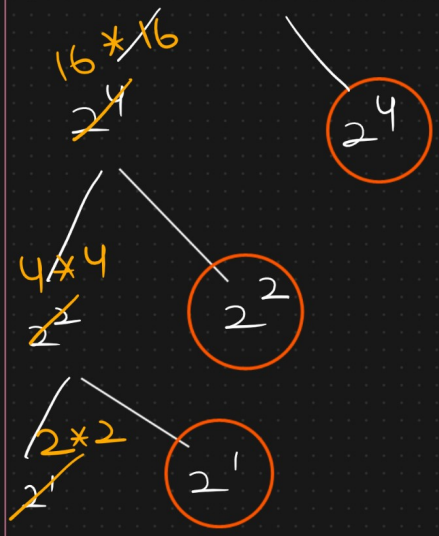
256 * 256

2^8 2^8

b = 65

$2^{65} = 2 * 2^{64}$

$a * a^b$



Divide & Conquer

$T(n)$

```
powerfind(a, b) {
  Base Case Condition
  if(b==1){
    return a;
  }                          } C
  else {
    mid = b/2;               <- Left Side result
    Recursive Calls
    result = powerfind(a, mid)   T(b/2)
    {
      finalResult = result * result;
      if(b%2==0){
        return finalResult;
      }
      else {
        return a * finalResult;
      }
    }                        C
  }
```

Recurrence Relation

$$T(n) = \begin{cases} 1 & b = 1 \\ T(b/2) + 1 & b > 1 \end{cases}$$

$$T(b) = T\left(\frac{b}{2}\right) + 1 \quad \text{---(1)}$$

$$T\left(\frac{b}{2}\right) = T\left(\frac{b}{2^2}\right) + 1$$

Substitution Method

$$T(b) = T\left(\frac{b}{2^2}\right) + 2 \quad \text{---(2)}$$

$$T\left(\frac{b}{2^2}\right) = T\left(\frac{b}{2^3}\right) + 1$$

$$T(b) = T\left(\frac{b}{2^3}\right) + 3 \quad \text{---(3)}$$

↓ k times

$$T(b) = T\left(\frac{b}{2^k}\right) + k$$

$$\frac{b}{2^k} = 1$$

$$b = 2^k$$

$$\log_2 b = k$$

$$T(n) = T\left(\frac{b}{2^{\log_2 b}}\right) + \log_2 b$$

$$= T\left(\frac{\cancel{b}}{\cancel{b}\,\log_2 \cancel{2}}\right) + \log_2 b$$

$$T(n) = O(\log_2 b)$$