

13. Fast Exponentiation a^b

Example: `Pow(base, power)`

→ $(base)^{power}$

We can solve through
these Approaches

{
NAIVE Approach
Binary Exponentiation
Fast Exponentiation
}

$$\underline{\text{Ex}} \quad 2^{10} = 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2$$
$$\text{pow}(2, 10) = 1024$$

$$\underline{\text{Ex}} \quad 2^7 = 2 \times 2 \times 2 \times 2 \times 2 \times 2 \times 2$$
$$= 128$$

Approach 1: NAIVE Approach

$$\begin{aligned}\text{pow}(\text{base}, \text{power}) &\Rightarrow \text{pow}(2, 10) \\ &\Rightarrow 2^{10} \\ &\Rightarrow 1024\end{aligned}$$

$$T.C. = O(N)$$

$$S.C. = O(1)$$

where, N is power

i	$i \leq \text{power}$
0	$0 < 10$
1	$1 < 10$
2	$2 < 10$
3	$3 < 10$
4	$4 < 10$
5	$5 < 10$
6	$6 < 10$
7	$7 < 10$
8	$8 < 10$
9	$9 < 10$
10	$10 < 10$ STOP

$$\begin{aligned}\text{Ans} &= \text{Ans} * \text{base} \\ &= 1 * 2 \\ &= 2 * 2 \\ &= 4 * 2 \\ &= 8 * 2 \\ &= 16 * 2 \\ &= 32 * 2 \\ &= 64 * 2 \\ &= 128 * 2 \\ &= 256 * 2 \\ &= 512 * 2 \\ &\rightarrow \text{Ans} = 1024\end{aligned}$$

Approach 2: Binary Exponentiation


$$\begin{aligned}\text{pow}(\text{base}, \text{power}) &\Rightarrow \text{pow}(2, 10) \\ &\Rightarrow 2^{10} \\ &\Rightarrow 1024\end{aligned}$$

power = even

$$\Rightarrow 2^{10} = 2^{2 \times 5} = (2^2)^5 = 4^5$$

power = odd

$$\Rightarrow 2^7 = 2 \times (2^6)$$


$$2^6 = 2^{2 \times 3} = (2^2)^3 = 4^3$$

Logic

Ex $(base)^{power} = (2)^{10}$

$$\Rightarrow 2^{10} = (2^2)^5 = (4)^5$$

Ans = 1

$$\begin{aligned} \rightarrow power &= power / 2 \\ &= 10 / 2 \\ &= 5 \end{aligned}$$

$$\begin{aligned} \rightarrow base &= base \times base \\ &= 2 \times 2 \\ &= 4 \end{aligned}$$

$$\Rightarrow 4^5 = \textcircled{4} \times 4^{\textcircled{4}} \rightarrow \begin{aligned} power &= power - 1 \\ &= 5 - 1 \\ &= 4 \end{aligned}$$

Ans = 4

$$\begin{aligned} \rightarrow Ans &= Ans \times base \\ &= 1 \times 4 \\ &= 4 \end{aligned}$$

DRY

power even

$$\begin{aligned} \rightarrow base &= base \times base \\ power &= power / 2 \end{aligned}$$

power odd

$$\begin{aligned} \rightarrow Ans &= Ans \times base \\ power &-- \end{aligned}$$

DRY RUN $2^{10} = 1024$

$$2^{10} = 1024$$

(I) $2^{10} \Rightarrow (2^2)^5 = (4)^5$, Ans = 1

(II) $4^5 \Rightarrow 4 \times 4^4$, Ans = $\frac{1}{4} * 4$
 = 4

(IV) $4^4 \Rightarrow (4^2)^2 = (16)^2$, Ans = 4

Q10 $16^2 \Rightarrow (16^2)^1 = (256)^1$, Ans = 1

(v) $256^1 \Rightarrow 256 \times 256^0$, $\text{Ans} = 4 \times 256$
 $= 1024$

(VI) $256^0 \Rightarrow$ δTOP $power_0 = 0$

↑ Base ↙ power

Ex CORNER CASE

$$\Rightarrow (2)^{-2} \Rightarrow \left(\frac{1}{2}\right)^2 = \frac{1}{4} = 0.25$$

База

Вариант $(\rho_{\text{полн}} < 0)$

↳ $base = 1/base$
 $power = abs(power)$



$$\left(\frac{1}{2}\right)^2 = (0.5)^2 \Rightarrow (0.25)$$

Ans 

```
// Approach 2: Binary Exponentiation (Leetcode-50)
```

```
class Solution {  
public:  
    double myPow(double base, int power)  
    {  
        // Corner Case  
        if (power < 0)  
        {  
            base = 1 / base;  
            power = abs(power);  
        }  
  
        // Main Logic to find the pow(base,power)  
        double ans = 1;  
        while(power > 0)  
        {  
            if (power % 2 == 0)  
            {  
                // Power is even  
                base = base * base;  
                power = power / 2;  
            }  
            else  
            {  
                // Power is odd  
                ans = (ans * base);  
                power--;  
            }  
        }  
        return ans;  
    }  
};
```

$$T.C. = O(\log N)$$

$$S.C. = O(1)$$

where, N is power

Approach 3: Fast Exponentiation

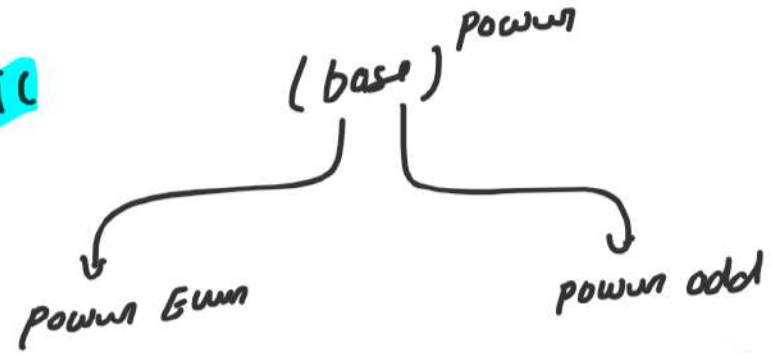
$$\begin{aligned}\text{pow}(\text{base}, \text{power}) &\Rightarrow \text{pow}(2, 10) \\ &\Rightarrow 2^{10} \\ &\Rightarrow 1024\end{aligned}$$

Explanation

$$10/2 = 5$$

$$\begin{aligned}10 &\Rightarrow 0000\ 1010 \\ 10 \gg 1 &\Rightarrow 0000\ 0101 \\ &\quad \boxed{5}\end{aligned}$$

Logic



$$\text{power} \neq 1 == 0$$

- $\text{base} = \text{base} * \text{base}$
- $\text{power} = \text{power} \gg 1$

$$\text{power} \neq 1 == 1$$

- $\text{Ans} = \text{Ans} * \text{base}$
- $\text{base} = \text{base} * \text{base}$
- $\text{power} = \text{power} \gg 1$

😊 power $\gg 1$ क्या मतलब क्या है?

$$\begin{aligned}&\hookrightarrow \text{power} / 2^1 \\ &\left\{ \begin{array}{l} \text{power} \gg 2 \\ \hookrightarrow \text{power} / 2^2 \end{array} \right.\end{aligned}$$

DRY RUN


$$\begin{aligned}\text{pow}(\text{base}, \text{power}) &\Rightarrow \text{pow}(2, 10) \\ &\Rightarrow 2^{10} \\ &\Rightarrow 1024\end{aligned}$$

$$\textcircled{\text{I}} \quad \text{power} = 10 \quad \begin{array}{l} \text{base} = 2 \times 2 \\ = 4 \end{array} \quad \begin{array}{l} \text{power} = 10/2 \\ = 5 \end{array} \quad \text{Ans} = 1$$

$$\textcircled{\text{II}} \quad \text{power} = 5 \quad \begin{array}{l} \text{Ans} = 1 * 4 \\ = 4 \end{array} \quad \begin{array}{l} \text{base} = 4 \times 4 \\ = 16 \end{array} \quad \begin{array}{l} \text{power} = 5/2 \\ = 2 \end{array}$$

$$\textcircled{\text{III}} \quad \text{power} = 2 \quad \begin{array}{l} \text{base} = 16 \times 16 \\ = 256 \end{array} \quad \begin{array}{l} \text{power} = 2/2 \\ = 1 \end{array} \quad \text{ans} = 4$$

$$\textcircled{\text{IV}} \quad \text{power} = 1 \quad \begin{array}{l} \text{Ans} = 4 * 256 \\ = 1024 \end{array} \quad \begin{array}{l} \text{base} = 256 \times 256 \\ = 65536 \end{array} \quad \begin{array}{l} \text{power} = 1/2 \\ = 0 \end{array} \rightarrow \text{stop} \quad (\text{power} \neq 0)$$




```
// Approach 3: Fast Exponentiation
```

```
class Solution {  
public:  
    double myPow(double base, int power)  
    {  
        // Corner case  
        if (power < 0)  
        {  
            base = 1 / base;  
            power = abs(power);  
        }  
  
        // Main Logic to find the pow(base,power)  
        double ans = 1;  
        while(power > 0)  
        {  
            if (power & 1)  
            {  
                // Power is odd  
                ans = ans * base;  
            }  
            // Power is even  
            base = base * base;  
            power = power >> 1;  
        }  
        return ans;  
    }  
};
```

$$T.C. = O(\log N)$$

$$S.C. = O(1)$$

Where, N is power