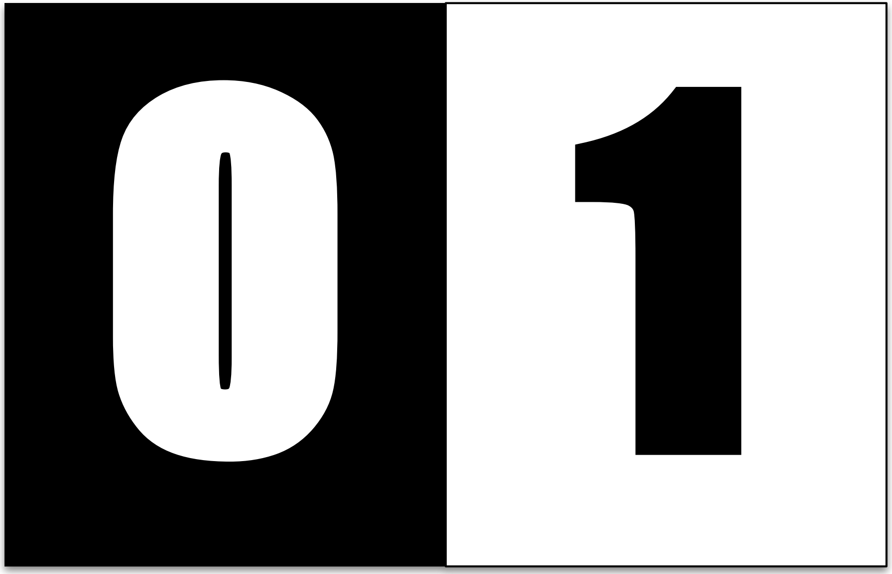
Bitwise Operations



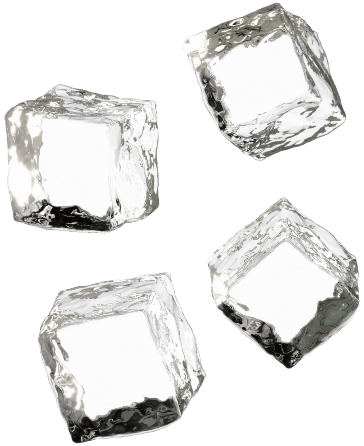
Africa To Silicon Valley

Lecture Flow

- Pre-requisites
- Definition of Bits
- Bit representation of integers
- Bitwise operators
- Bitwise operators properties
- Bit Masking
- Bit Masking operations
- Python's built in functions
- Practice questions
- Quote of the day

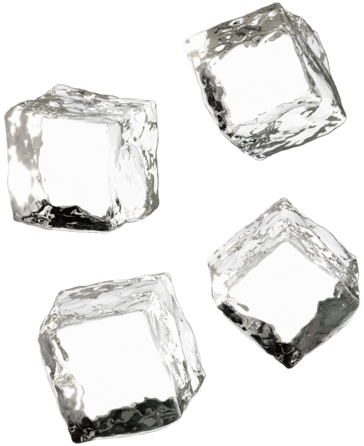
Pre-requisites

No prerequisites folks



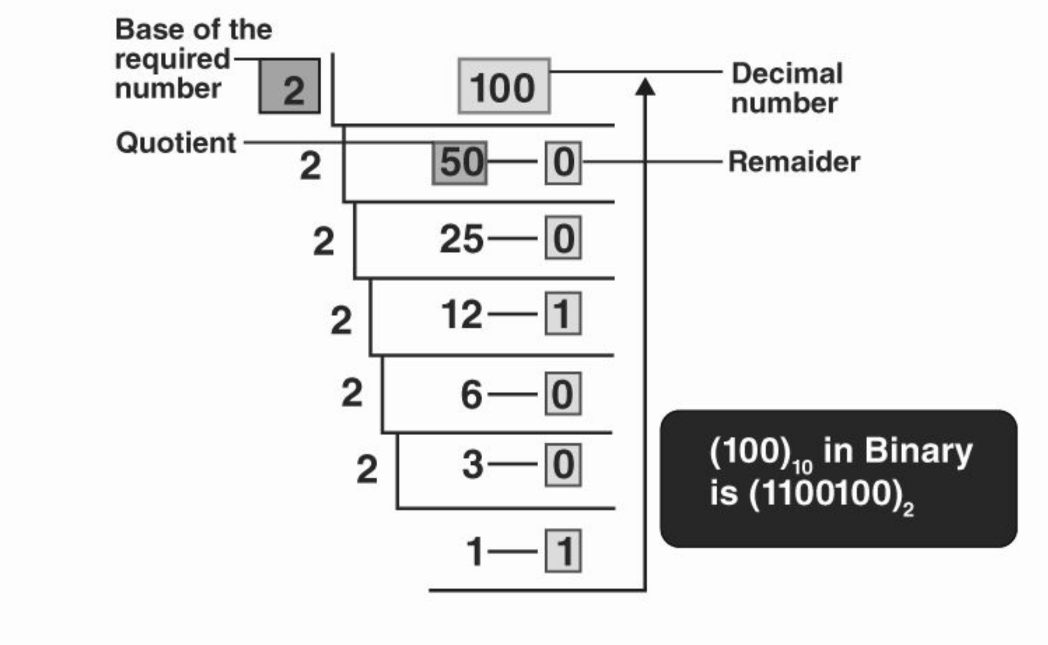
What are Bits?

Bit is the smallest unit of data in a computer system and represents a **binary digit**, which can have one of two values: **0** or **1**.



Bit Representation for unsigned?

Bit Representation



Binary **Decimal**

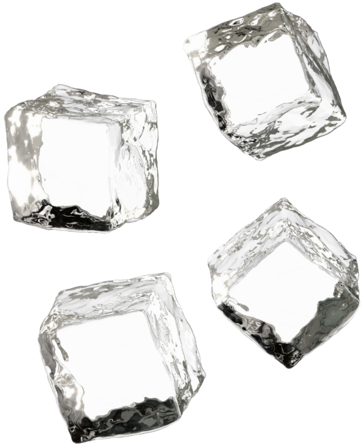
MSB **LSB**

00000000 = 000

$2^7\ 2^6\ 2^5\ 2^4\ 2^3\ 2^2\ 2^1\ 2^0$ $\qquad 10^2\,10^1\,10^0$

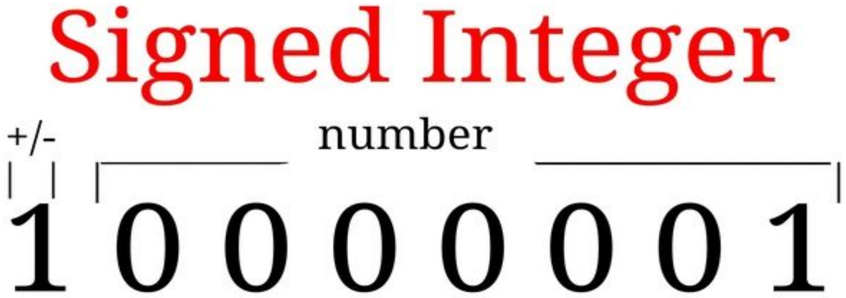
0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 = 0 + 0 + 0

Africa To Silicon Valley



Bit Representation for signed?

Bit Representation



2’s Complement

2's Complement

- A method to represent negative numbers
- Most popular method
- Allows adding negative numbers with the same logic gates as positive numbers
- Main Idea: $x + (-x) = 0$

2’s Complement

How to convert number to 2's complement

1. Convert the positive number to binary
2. Flip the bits (1 to 0 and 0 to 1)
3. Add 1

Two's complement binary	Decimal
0111	+7
0110	+6
0101	+5
0100	+4
0011	+3
0010	+2
0001	+1
0000	0
1111	-1
1110	-2
1101	-3
1100	-4
1011	-5
1010	-6
1001	-7
1000	-8

Why Bits?

- Some questions require bit manipulations.
- It can be used to optimize solutions and simplify solutions.

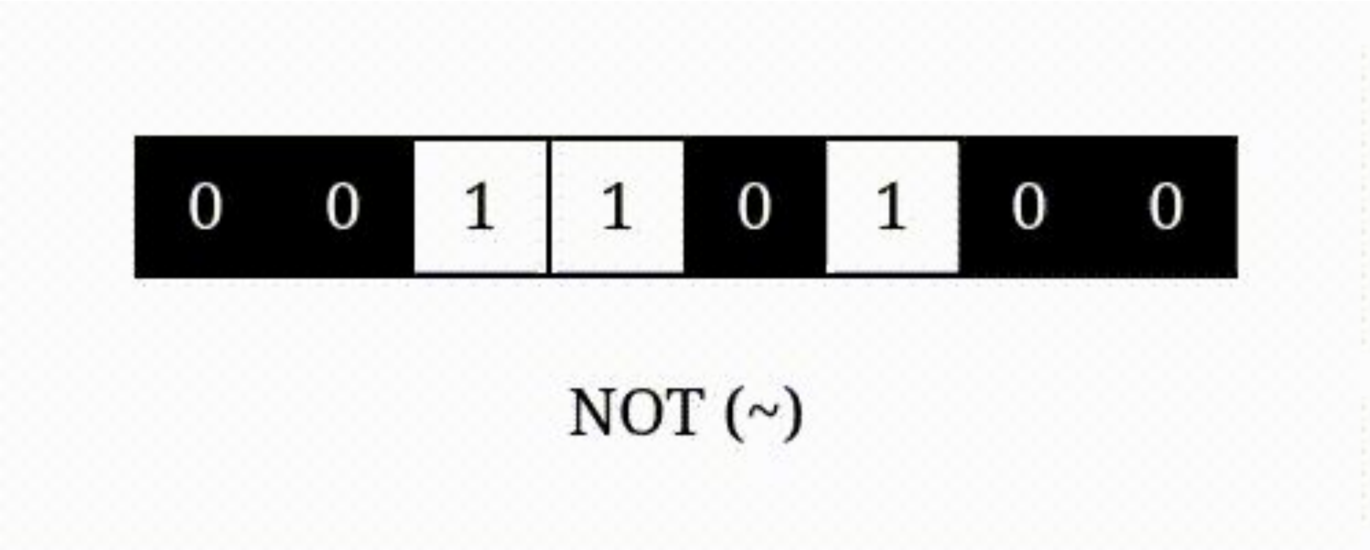
Bit Operators

Bit Operators

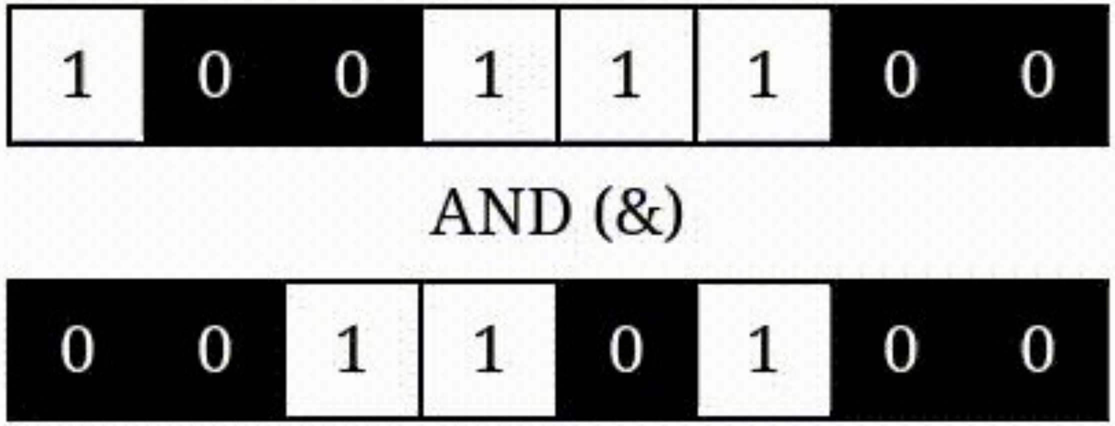
- NOT (~)
- AND (&)
- OR (|)
- XOR (^)
- Bit Shifts (<<,>>)

Africa To Silicon Valley

NOT



AND

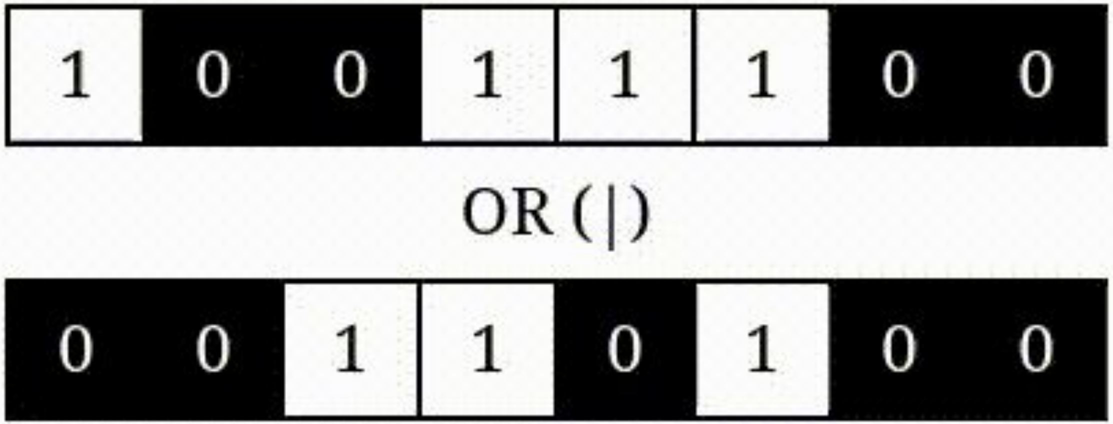


Africa To Silicon Valley

Check yourself

$$3 \,\&\, 2 = ?$$

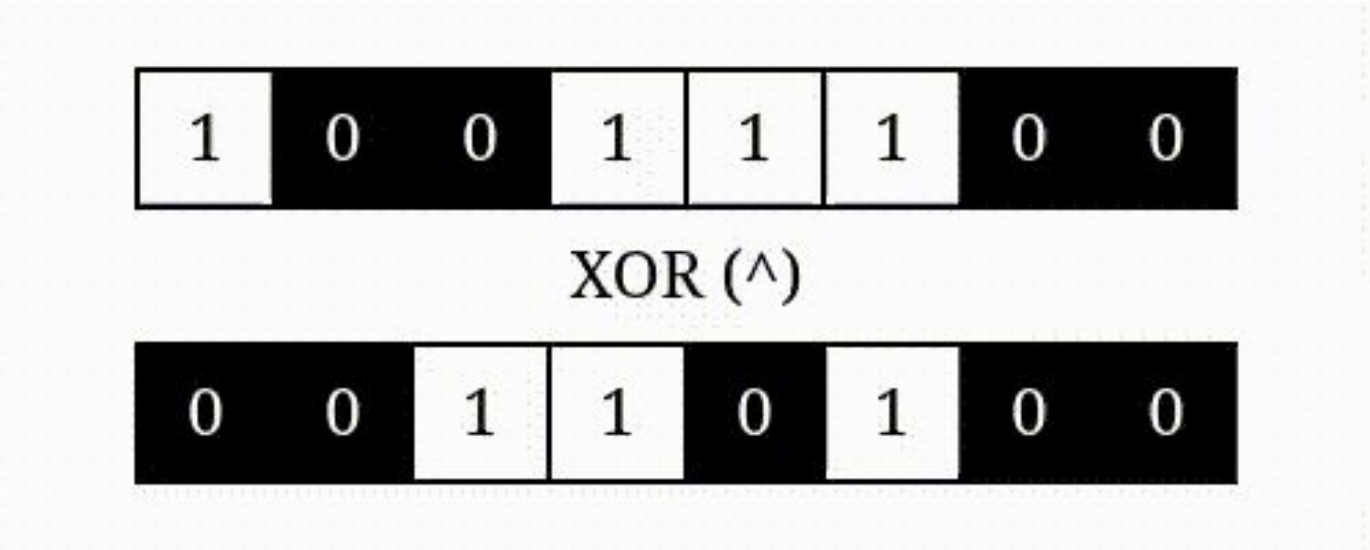
OR



Check yourself

$$7 \mid 8 = ?$$

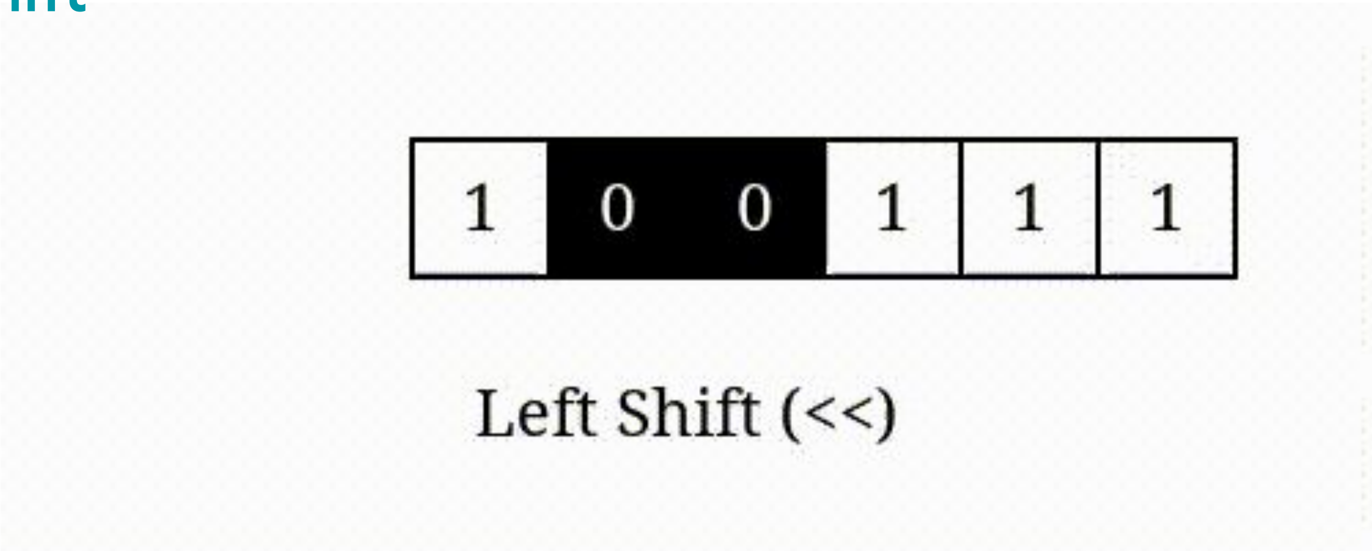
XOR



Check yourself

$4 \wedge 9 = ?$

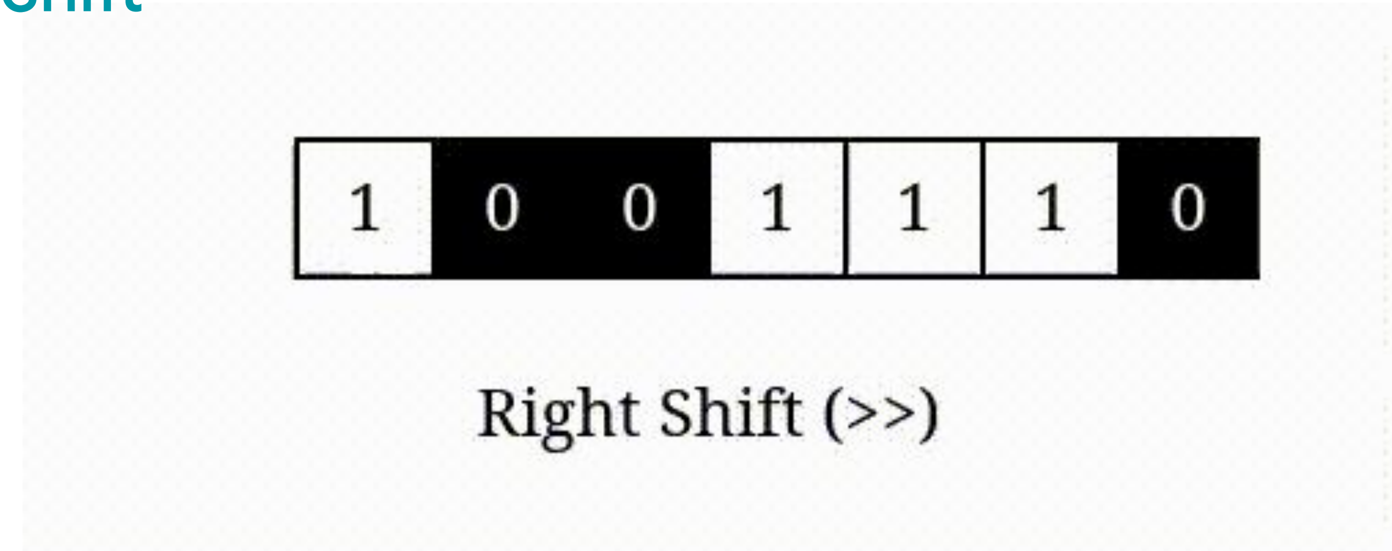
Left Shift



Check yourself

$$1 << 3 = ?$$

Right Shift



Check yourself

$$16 >> 3 = ?$$

- Each right shift operation reduces the number to its half.

Bit Operators

A	B	A | B	A & B	A^ B	~A
0	0	0	0	0	1
0	1	1	0	1	1
1	0	1	0	1	0
1	1	1	1	0	0

Checkpoint

Question #1

First let's solve it using normal approach.

Then let's solve it using bits (shifting operation)

338. Counting Bits

Hint

Easy 8.6K 408

Companies

Given an integer `n`, return *an array* `ans` *of length* `n + 1` *such that for each* `i` (`0 <= i <= n`), `ans[i]` *is the **number of*** `1`***'s** in the binary representation of* `i`.

Example 1:

```
Input: n = 2
Output: [0,1,1]
Explanation:
0 --> 0
1 --> 1
2 --> 10
```

Africa To Silicon Valley

Bitwise operators properties

- Commutative
    - x ^ y = y ^ x
- Associative
    - x & (y & z) = (x & y) & z
- Do these properties hold for all bit operators?

Africa To Silicon Valley

More Bitwise Operator Properties

- Identity
  - XOR: x ^ 0 = x
  - What about the other operators?
- Inverse
  - XOR: x ^ x = 0
  - What about the other operators?

Africa To Silicon Valley

Checkpoint

Africa To Silicon Valley

Question #2

First let's solve it using normal approach.

Then let's solve it using bits.

268. Missing Number

Easy 7834 3002 Add to List Share

Given an array `nums` containing `n` distinct numbers in the range `[0, n]`, return *the only number in the range that is missing from the array.*

Example 1:

```
Input: nums = [3,0,1]
Output: 2
Explanation: n = 3 since there are 3 numbers, so all numbers are in the
range [0,3]. 2 is the missing number in the range since it does not appear
in nums.
```

Example 2:

```
Input: nums = [0,1]
Output: 2
Explanation: n = 2 since there are 2 numbers, so all numbers are in the
range [0,2]. 2 is the missing number in the range since it does not appear
in nums.
```

Example 3:

```
Input: nums = [9,6,4,2,3,5,7,0,1]
Output: 8
Explanation: n = 9 since there are 9 numbers, so all numbers are in the
range [0,9]. 8 is the missing number in the range since it does not appear
in nums.
```

Bit masking

A2SV Africa To Silicon Valley

Bit masking

- Way of optimizing storage
- Store information in a single bit

Test 5th Bit

```
num = 100010001000111

 1  = 000000000000001
```

Test 5th Bit

```
num   = 100010001000111

  1   = 000000000000001

1<<5  = 000000000100000

   100010001000111
 & 000000000100000
 ------------------
   000000000100000 (!= 0)
```

Africa To Silicon Valley

Implement

Bit masking operations

Test k^{th} bit is set: `num & (1 << k) != 0`.

Set k^{th} bit: `num |= (1 << k)`.

Turn off k^{th} bit: `num &= ~(1 << k)`.

Toggle the k^{th} bit: `num ^= (1 << k)`.

A2SV
Africa To Silicon Valley

Checkpoint

46. Permutations

Medium 15.1K 256

Companies

Given an array `nums` of distinct integers, return *all the possible permutations*. You can return the answer in **any order**.

Example 1:

```
Input: nums = [1,2,3]
Output: [[1,2,3],[1,3,2],[2,1,3],[2,3,1],[3,1,2],[3,2,1]]
```

Example 2:

```
Input: nums = [0,1]
Output: [[0,1],[1,0]]
```

Example 3:

```
Input: nums = [1]
Output: [[1]]
```

Question #3

- Use bit mask to keep track of used numbers

Python Built-in Functions

A2SV
Africa To Silicon Valley

Bit masking

- bin(): This built-in function can be used to convert an integer to a binary string.
- int(): This built-in function can be used to convert a binary string to an integer.

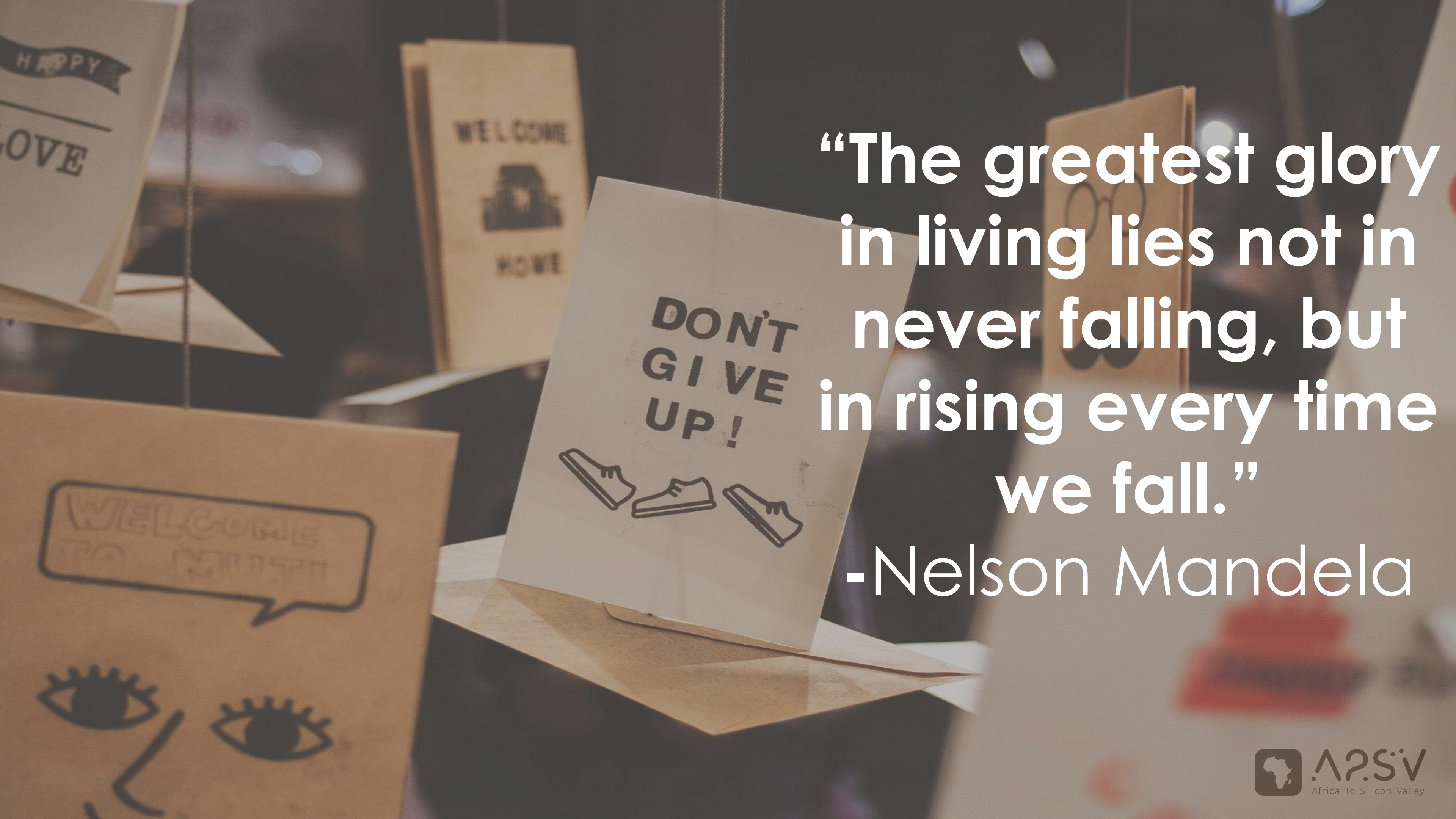
Bit masking

- bit_length(): This method can be called on an integer and returns the number of bits required to represent the integer in binary, excluding the sign bit.
- bit_count() : this method can be called on an integer and returns the number of set bits.

Practice Problems

- Number Complement
- Hamming Distance
- Cirno's Perfect Bitmasks Classroom
- Subsets
- Add Binary
- Single Number II
- Single Number III
- Sum of Two Integers
- Maximum Product of Word Lengths

Africa To Silicon Valley



"The greatest glory in living lies not in never falling, but in rising every time we fall."

-Nelson Mandela

A2SV
Africa To Silicon Valley