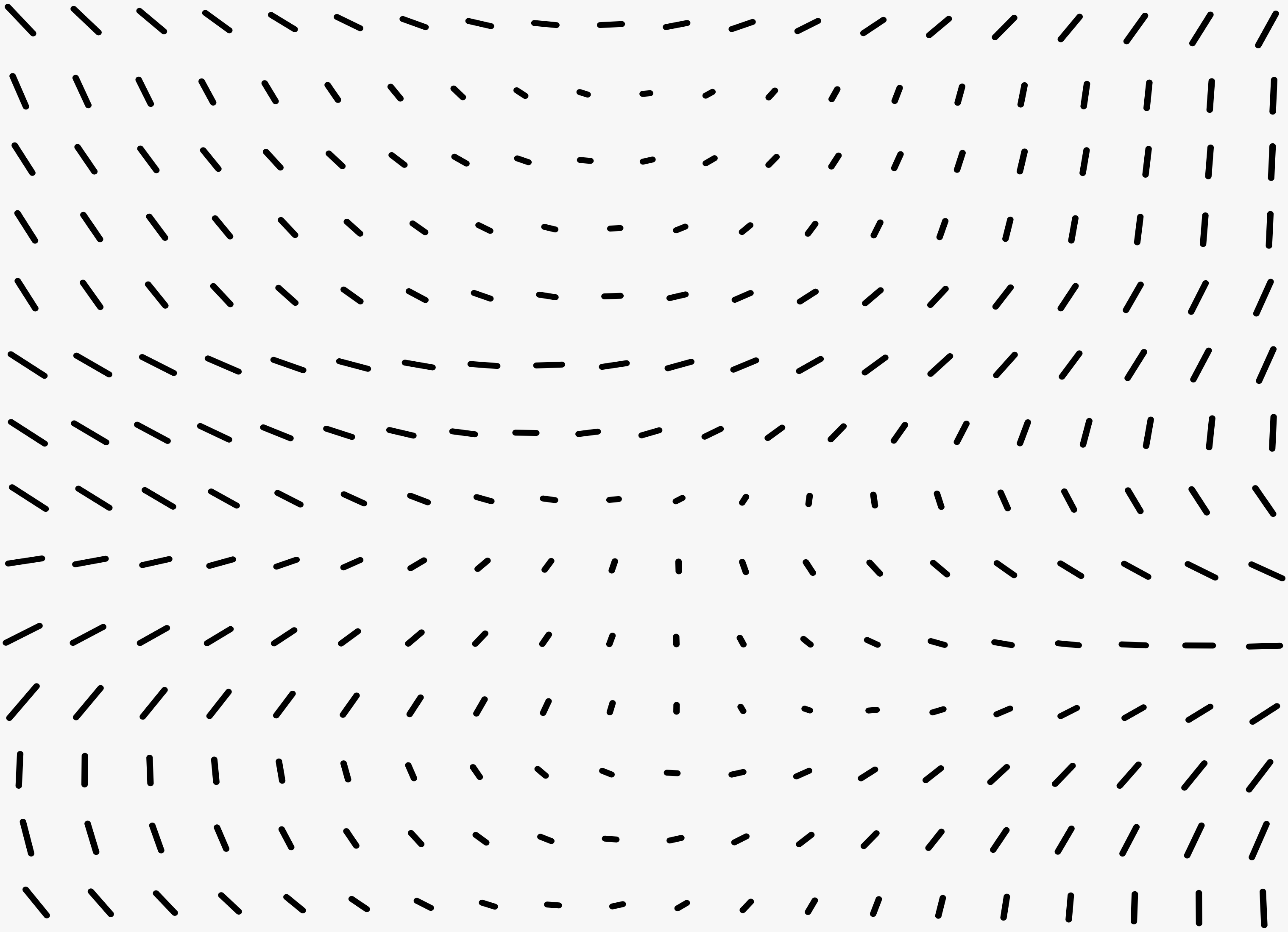

Bit Manipulation

Decimal number system: {0 - 9} digits

We fill the ones place, then go and fill tens place, hundreds and so on.

Binary number system: {0 - 1} digits

Conversion

1. Convert decimal to binary.

Ex: a) 30

```
2 | 30
2 | 15 - 0   ↑
2 |  7 - 1   |
2 |  3 - 1   |
2 |  1 - 1   |
     0 - 1
```

$30_{(10)} - 011110_{(2)}$

b) 45

```
2 | 45
2 | 22 - 1
2 | 11 - 0   ↑
2 |  5 - 1   |
2 |  2 - 1   |
2 |  1 - 0   |
     0 - 1
```

$45_{(10)} - 0101101_{(2)}$

2. Convert binary to decimal number

Ex: a) $(\overset{4}{1}\ \overset{3}{0}\ \overset{2}{1}\ \overset{1}{0}\ \overset{0}{1})_2$ → index of each bit

$$1\times2^4 + 0\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0$$

$$16 + 0 + 4 + 0 + 1 = 21_{(10)}$$

NOTE -

0 - unset bit / off bit

1 - set bit / on bit

3. Add binary numbers

$0 + 0 \to 0$

$0 + 1 \to 1$

$1 + 0 \to 1$

$1 + 1 \to 10$

carry

$1 + 1 + 1 \to 11$

carry

Ex: a)

$$\begin{array}{cccccc} 1 & 1 & 1 & 1 & & \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ \hline 1 & 1 & 0 & 1 & 0 & 1_{(2)} \\ \hline \end{array}$$

carry

b)

$$\begin{array}{ccccccc} 1 & & 1 & 1 & 1 & 1 & \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ \hline 1 & 0 & 1 & 0 & 0 & 1 & 0_{(2)} \\ \hline \end{array}$$

Bitwise operators - done only on bits

AND, OR, XOR, LEFT SHIFT, RIGHT SHIFT

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

Ex: 1. 26 & 35

→ firstly convert to binary numbers

→ perform bitwise & operation

```
      26  →   0 1 1 0 1 0
      35  →   1 0 0 0 1 1
     (&)     _____________
              0 0 0 0 1 0  —  2(10)
             _____________
```

2. 23 | 10

```
      23  →   0 1 0 1 1 1
      10  →   0 0 1 0 1 0
             _____________
     (|)      0 1 1 1 1 1  —  31(10)
             _____________
```

3. 23 ^ 10

```
23 – 0 1 0 1 1 1
10 – 0 0 1 0 1 0
     ___________
(^)  0 1 1 1 0 1  –  29(10)
```

Given a number identify whether it is even or odd without using arithmetic operators.

All even numbers LSB will be always 0 whereas LSB for odd numbers will be always 1. To check last bit, use bitwise AND. If the answer is 0 we say the number is even otherwise odd.

Ex: 1) N = 10 – even   2) N = 11 – odd

```
    1 0 1 0                  1 0 1 1
&   0 0 0 1              &   0 0 0 1
   _________                _________
    0 0 0 0 – zero           0 0 0 1 – non-zero
```

```
void checkEven(int n){
    if (n & 1 == 0){
        print ("even");
    } else {
        print ("odd");
    }
}
```