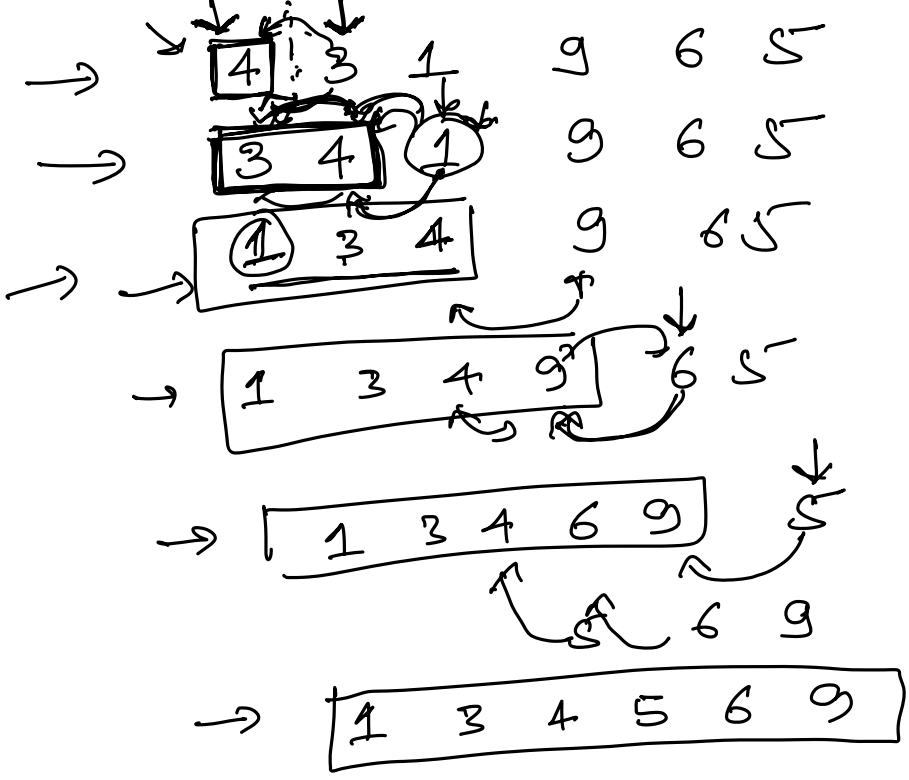
9:05

Insertion Sort



1. we consider the first element as sorted
2. we pick next element and insert in the right place of the sorted.

```
for (i = 1; i < n; i++){
    j = i - 1
    while (j >= 0 && arr[j] > arr[j+1]){
        swap (arr[j], arr[j+1]);
        j--;
    }
}
```

i
5 4 1 2 3
j

T.C. :- $O(n^2)$

S.C :- $O(1)$

Decimal Number System

- base 10
- every day life.
- 0 — 9

$342 \Rightarrow 3\times10^2 + 4\times10^1 + 2\times10^0$

$2563 \Rightarrow 2000 + 500 + 60 + 3.$

$\Rightarrow 2\times10^3 + 5\times10^2 + 6\times10^1 + 3\times10^0$

Binary Numbers System

- base 2
- 0, 1
- number can be representin in power of 2

$110 \rightarrow 1\times2^2 + 1\times2^1 + 0\times2^0$

$= 4 + 2 + 0 = 6$

$1011 \Rightarrow 1\times2^3 + 0\times2^2 + 1\times2^1 + 1\times2^0$

$= 8 + 0 + 2 + 1$

$= 11$

Binary to Decimal.

$\Rightarrow$

$\Rightarrow$ 1 0 1 1 0 1 0

$64\times1 + 32\times0 + 16\times1 + 8\times1 + 0\times4 + 2 + 0$

$64 + 0 + 16 + 8 + 0 + 2$

$= 90$

Decimal to Binary

Remainder.

2	20	
2	10	0
2	5	0
2	2	1
2	1	0
	0	1

$(20)_{10} = (1\ 0\ 1\ 0\ 0)_2$

Remainder.

2	30	
2	15	0
2	7	1
2	3	1
2	1	1
	0	1

$1\ 1\ 1\ 1\ 0$

Quiz 2. 45 → 101101

2	45	
2	22	1
2	11	0
2	5	1
2	2	1
2	1	0
	0	1

101101

Add Decimal number

[3, 6, 8]
[4, 5, 3]
[8, 2, 1]

```
  1  1  1   0 - C = 0
  0  3  6  8  → d1
+ 0  8  5  3  → d2
-----------
  1  2  2  1
```

$$digit = (carry + d_1 + d_2) \% 10$$

$$carry = (carry + d_1 + d_2) / 10$$

digit = (0 + 8 + 3) % 10
= 11 % 10
= 1

carry = (0 + 8 + 3) / 10
= 11 / 10
= 1

digit = (1 + 6 + 5) % 10
= 12 % 10
= 2

carry = (12) / 10
= 1

digit = (1 + 3 + 8) % 10
= 12 % 10 = 2

carry = 12 / 10 = 0

183
43
126 %

digit = (1+0+0)%10 ✓ = 1

carry = 1/10 = 0

Addition of Binary Numbers

```
  1  1  0  1  0
  1  0  1  0  1
+    1  1  0  1
-------------------
  1  0  0  0  1  0
```

32 2

= 34

16+4+1 = 21

8+4+1 = 13

34

digit = (carry + d_1 + d_2) % 2

carry = (carry + d_1 + d_2) / 2

(1+1+0) % 2 = 2 % 2

carry = 2/2 = 1

digit = (1+0+0) % 2 = 1

carry = 1/2

(0+0+0) & % 2 = 0

```
  1  0  0  1  0  0  0
  0  1  1  0  1  0  1
  0  1  0  0  1  1  0
-----------------------
  1  0  1  1  0  1  1
```

= (0+1+0) % 2

1/2

0+1+1

= 2 % 2 =

2/2 = 1

Quiz 8

```
    1  1  0
  1  0  1  1  0
+    0  0  1  1  1
-----------------
  1  1  1  0  1
```

(1+1+1)

= 3 % 2

carry = 3/2

= 1

10:11 pm

Bitwise Operators

Bitwise operators are used to perform operations on individual bits of a binary number.

0 → false / unset
1 → true / set.

( ) (&) ( )

AND (&)

0	0	0
0	1	0
1	0	0
1	1	1

resulting bit will be one if both the bit are set

OR (|)

0	0	0
0	1	1
1	0	1
1	1	1

resulting bit will be 1 if either/both of the bit is set.

XOR (^)

0	0	0
0	1	1
1	0	1
1	1	0

if bits are same = 0

if bits are different = 1

Not (!)

0	1
1	0

it reverse the bit

set → unset

unset → set.

---

if you want to find if a number is even number

15 | 1010

10 |

$2^0 = 1 \times 0 = 0$

$1 \times 1 = 1$

$2^0 \times 1 = 1$

even + 1

{ %2 == 0 } → even

!= 0 → odd.

(n & 1 == 1) = odd

else even

1111 & 0001 = 0001

1010 & 0001 = 0000

⇒ 5 & 6

```
  101
&  110
-------
  100   = 4
```

⇒ 92 | 154

⇒ ① 01011100
   10011010

   11011110

④ ! 1001

⇒ 0110

⑤ ^ 92 ^ 154

```
  01011100
^ 10011010
-----------
  11000110
```

Qs

20 ⇒ 00010100

45 ⇒ 00101101

00111001

$1*2^5 + 1*2^4 + 1*2^3 + 0*2^2 + 0*2^1 + 1*2^0$

$= 32 + 16 + 8 + 1$

$= 57$

⇒ Negative number in binary.

2's complement

comput ⇒ 32 bit

int ↗

↳ 4byte

−7 → 2's

1 byte = 8 bit

4 byte = 32 bit.

a computer store negative numbers in 2's complement :

↳ ① → invert all the bits

② → add 1

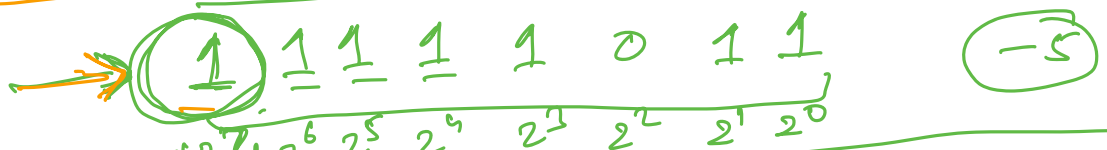
int → 8 bit

(-5) in binary.

MSB → Most Significant bit

① Invert all bits. 1 1 1 1 1 0 1 0

② add 1 0 0 0 0 0 0 0 1 +



MSB → used to store whether negative or positive

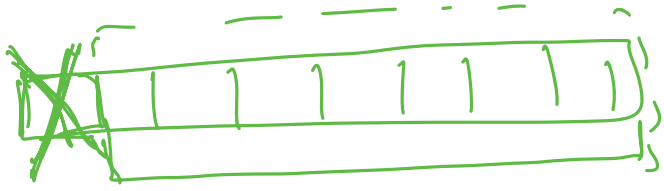
MSB = 1 → number is negative
MSB = 0 → number is positive.

$2^4 = |-128 + 64 + 32 + 16 + 8 + 0 + 2 + 1|$

$= \boxed{-5}$

① → Computer understan MSB → is a negative number.

Int → $-\boxed{2^{32} - 1}$



(-7)

= 0 0 0 0 0 1 1 1] → 8 bit

① invert all bits. 1 1 1 1 1 0 0 0

+ 0 0 0 0 0 0 0 1

② add 1 1 1 1 1 1 0 0 1 → stored computer

251 stored in 32bit. → 251

Quiz −3 → 2's complement 8-bit

0 0 0 0 0 0 1 1

(1) Invert 1 1 1 1 1 1 0 0

(2) add 1 → 0 0 0 0 0 0 0 1

1 1 1 1 1 1 0 1

Quiz (−10) → 8 bit

0 0 0 0 1 0 1 0

(1) → Invert 1 1 1 1 0 1 0 1

(2) Add 0 0 0 0 0 0 0 1

1 1 1 1 0 1 1 0

Range → int → $[-2^{32-1}, 2^{32-1}-1]$ $[-10^9$ to $10^9]$

long → $[-2^{64-1}, 2^{64-1}-1] = -10^{18}$ to 10^{18}

Importance of Constraints

10 0 0 0 0 0 0 0

int ⇒ 10^9

int a = 10^5 .5 <
int b = 10^6 .6 < → 10^{11}

① int c = a × b = 10^{11} > 10^9 ✗

overflow

② long c = [a × b] → ALU (Arith. Logicum.)

int int

10^{11} > 10^9 ✗

garbage

③ long c = long (a × b) ✗

ALU → int × int

garbage

④ long c = (long) a × b ;

ALU → long × int

long × int = long

10^{11} 10^{18}

Qu Given an array of size N.
Calculate the sum of array.

Constraints:
$1 <= N <= 10^5$
$1 <= A[i] <= 10^6$

$10^6 \times 10^5 = 10^{11} \rightarrow 10^9$ int range.

```
long ~~int~~ sum = 0
for (i = 0; i < n; i++)
{
    sum = sum + Arr[i];
}
print(sum).
```