→
```
# include <stdio.h>
# include <math.h>
  main ()
{
   int a, b;
  Scanf ("%d, %d", &a, &b);
  int max = a > b ? a : b;
  printf ("%d", max);
}
```

[Operator: "? : a"]

q) A cashier has currency notes of denominations 1, 2, 5, 10, 50 & 100. If the amount to be withdraw -n is input through the keyboard. Find the total number of currency notes of each denomination the cashier will have to give to the withdrawer (only integer amount).

→
```
# include <stdio.h>
# include <math.h>
  main ()
{
  int amt;
```

```
int hund = amt / 100;
    amt = amt % 100;
int fif = amt / 50;
    amt = amt % 50;
int ten = amt / 10;
    amt = amt % 10;
int five = amt / 5;
    amt = amt % 5;
int two = amt / 2;
    amt = amt % 2;
int one = amt / 1;
    amt = amt
printf ("Hundered = %d", hund);
printf ("fifty = %d", fif);
printf ("ten = %d", ten);
printf ("five = %d", five);
printf ("two = %d", two);
printf ("one = %d", one);
}
```

Q) Write a C prgm to find k^{th} bit of a number.

Interview

Ex: N = 25 ⇒ K = 2

	4	3	2	1	0
25 ⇒	1	1	0	0	1

output: 2nd bit is 0 (Reset)

→ Masking: We make those bits as Zero which are not asked.

K th bit		2nd number
1	⇒	2
2	⇒	4
3	⇒	8

⇒ 2nd number = pow(2, K) = a

⇒ res = N & a;

Note: pow function gives answer in double datatype, bitwise operators cannot be applied on float or double.

i) ∴ res = N & (int) pow(2, K);

Here the value given by power function is converted into int (Type casting)

also if y = x << k ⇒ y = x * 2^k

⇒ y = 1 << k;

ii) res = N & (1<<k); // 1 * 2^k //

Eg:

	1	1	0	0	1
&	0	0	1	0	0
	0	0	0	0	0 ⇒ Reset

	1	1	0	0	1
&	0	1	0	0	0
	0	1	0	0	0 ⇒ Set

if res == 0 , reset ;
if res != 0 , Set ;
Code: res == 0 ? printf ("Reset") : printf ("Set");

⇒
```
# include<stdio.h>
# include <math.h>
main ()
{
  int N, k ;
Scanf ("%d %d", &N &k );
int res ;
 res = N & (int) pow (2, k);
 printf ("kth bit = %d", res);
 res == 0 ? printf ("Reset") : printf ("Set");
}
```

Note: 0111 1110 — 4bits nibble, 4bits nibble — 1 byte = 8 bits, 4bits = 1 nibble

Getting last 4 bits of a byte

N =	0	1	1	1	1	0	0	1
2nd =	0	0	0	0	1	1	1	1
	0	0	0	0	1	0	0	1

Q) Read conversion Hexadecimal to binary & binary to Hexadecimal (Ram structure)

*) INF: (infinity):

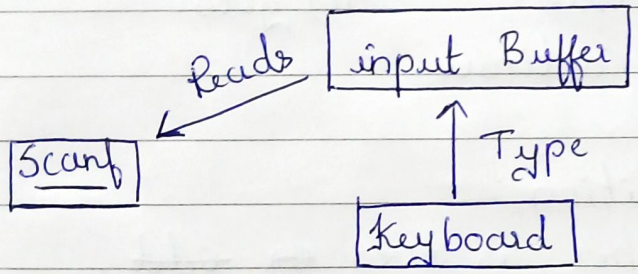
Eg: $\frac{1.0}{0}$ = INF

*) NAN: (Not a numba):

If we give -ve input a Sqrt function, then some compilers will give NAN as the answer

Eg: $\sqrt{-1}$ = NAN.

*) Reading a character:



Ascii value of \n = 10.

Eg:
```
int main () { int a; char c;
Scanf ("%d", &a);
Scanf ("%*c %c", &c); }
```

⇒ %*c :- It means read a character and do not store in any variable. It is used to remove \n, \t or a space from the input Buffer. Write %*c when we are reading character not for the 1st time. Similarly we have %*d: It means read an integer & do not store.