#include<stdio.h>

int main()

{

int x;

x = 693;

printf("%d\n",x);

return 0;

}

int x; means 4 bytes will be allocated for x means 32 bits

The data type of x is int

In 32 bits -> 1 sign bit and 31 data bits.

X = 693; 693 is in decimal form means this symbol is between 01,2,3,4,5,6,7,8,9

The decimal system comprises 10 symbols 0,1,2,3,4,5,6,7,8,9

This can vary based on language in chines it may be different it will be 10 symbols.

| x |  |
| --- | --- |
|  | 4 bytes |

We have given input in decimal format

And wherever data will be written in RAM will be in Binary format.

We have to convert it in binary from decimal.

| 2 | 693 | 1 |
| --- | --- | --- |
| 2 | 346 | 0 |
| 2 | 173 | 1 |
| 2 | 86 | 0 |
| 2 | 43 | 1 |
| 2 | 21 | 1 |
| 2 | 10 | 0 |
| 2 | 5 | 1 |
| 2 | 2 | 0 |
| 2 | 1 | 1 |

Now write these reminders in order from bottom to top

1 0 1 0 1 1 0 1 0 1

10 9 8 7 6 5 4 3 2 1 -> this will have only 10 bits but we have to make it 32 bits long

00000000000000000000001010110101 - this is a binary of 693

Now meaning of %d

%d -> Consider 1st as a sign bit and the rest as data bits.

0000000000000000000001010110101

~

This first bit will be consider as sign bit

If the singh bit is

0 - it means printf will consider it as binary of a positive number

1 - it means printf will consider it as Binary of negative number

Now how printf convert binary to decimal

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 32 | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 2147483648 | 1073741824 | 536870912 |  |  |  |  |  |  |  |  |  |  |  |  | 65538 | 32768 | 16384 | 8192 | 4096 | 2048 | 1024 | 512 | 256 | 128 | 64 | 32 | 16 | 8 | 3 | 2 | 1 |

First binary values

2nd counter

3rd two to the power

4th value of 2 to the power like 2^0 = 1, 2^1 = 2, 2^2 = 4 … 2^31 = 2147483648

Now we have values

Because it is treated as binary of a positive number printf will add the place value of all On (1) bits

| 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 1 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2^9 | 2^8 | 2^7 | 2^6 | 2^5 | 2^4 | 2^3 | 2^2 | 2^1 | 2^0 |
| 512x1 | 256x0 | 128x1 | 64x0 | 32x1 | 16x1 | 8x0 | 4x1 | 2x0 | 1x1 |
| 512 | 0 | 128 | 0 | 32 | 16 | 0 | 4 | 0 | 1 |

512+128+32+16+4+1 = 693

Example of a Negative number

#include<stdio.h>

int main()

{

int x;

x = -569;

printf("%d\n",x);

return 0;

}

Steps

1st find binary of 569: 1000111001

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Find 2’s Compliment of this number

2 steps

1. Make all 0 to 1 and 1 to 0

| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

1. Add 1 bit to right-most bit

| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |

Bit Calculations

0+0 = 0

1+0 = 0

0+1= 1

1+1 = 1 (Carry over 1)

This is a Binary of -569

| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Now how will printf print this value?

Print will consider 1st bit as a sign bit and its value is 1

The binary is of a negative number

So printf will take its 2’s compliment

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Then add one

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | 1 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Result:

| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |

Find value

| 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2^9 | 2^8 | 2^7 | 2^6 | 2^5 | 2^4 | 2^3 | 2^2 | 2^1 | 2^0 |
| 512x1 | 256x0 | 128x1 | 64x0 | 32x1 | 16x1 | 8x0 | 4x1 | 2x0 | 1x1 |
| 512 | 0 | 0 | 0 | 32 | 16 | 8 | 0 | 0 | 1 |

512+32+16+8+1 = 569

Then printf will put this number with minus symbol due to 1 at first place.

printf(“%d”, x)

-569

What is Limit for int x;?

32 bits -> 1 sign and 31 values

So range is -2^31 to 2^31-1 i.e. -2147483648 to 2147483647

What if i assign value larger then the given range or smaller then than given range.

Example:

#include<stdio.h>

int main()

{

int x;

x = 2147483659;

printf("%d\n", x);

x = -2147493660;

printf("%d", x);

return 0;

}

/\*

THIS IS WARNING NOT ERROR:

eg48.c:5:5: warning: implicit conversion from 'long' to 'int' changes value from 2147483659 to -2147483637 [-Wconstant-conversion]

x = 2147483659;

~ ^~~~~~~~~~

eg48.c:7:5: warning: implicit conversion from 'long' to 'int' changes value from -2147493660 to 2147473636 [-Wconstant-conversion]

x = -2147493660;

~ ^~~~~~~~~~~

2 warnings generated.

\*/

Output:

-2147483637

2147473636