BITWISE OPERATORS

Bitwise operator's works on bits.

Turbo-c is a 16 bit compiler. Due to this bitwise operations are limited to 16 bits only [2^0 to 2^{15}].

Bitwise operators operate **integer** type values only.

We have to calculate only the **on** bits [**1**].

When the first bit[**Sign bit**] is **1** then the number is **Negative** and it is **0** then the number is **positive**.

They are very much used in system software development.

Note: Bitwise operator is low level feature.

C-Language supports following bitwise operators.

& -Bitwise and

| - Bitwise or

^ - XOR ==> Exclusive OR

~ - Compliment operator

<< - Left shift operator

>> - Right shift operator

& - Bitwise and: In this both bits are 1's then result bit is 1. Otherwise result bit is 0.

Eg: **25 & 15 = 9**

```
25 = 0000  0000  0001  1001
15 = 0000  0000  0000  1111
```

```
2 | 25
2 | 12 - 1
2 | 6 - 0
2 | 3 - 0
    1 - 1
```

```
2 | 15
2 | 7 - 1
2 | 3 - 1
    1 - 1
```

---

25 & 15 = 9

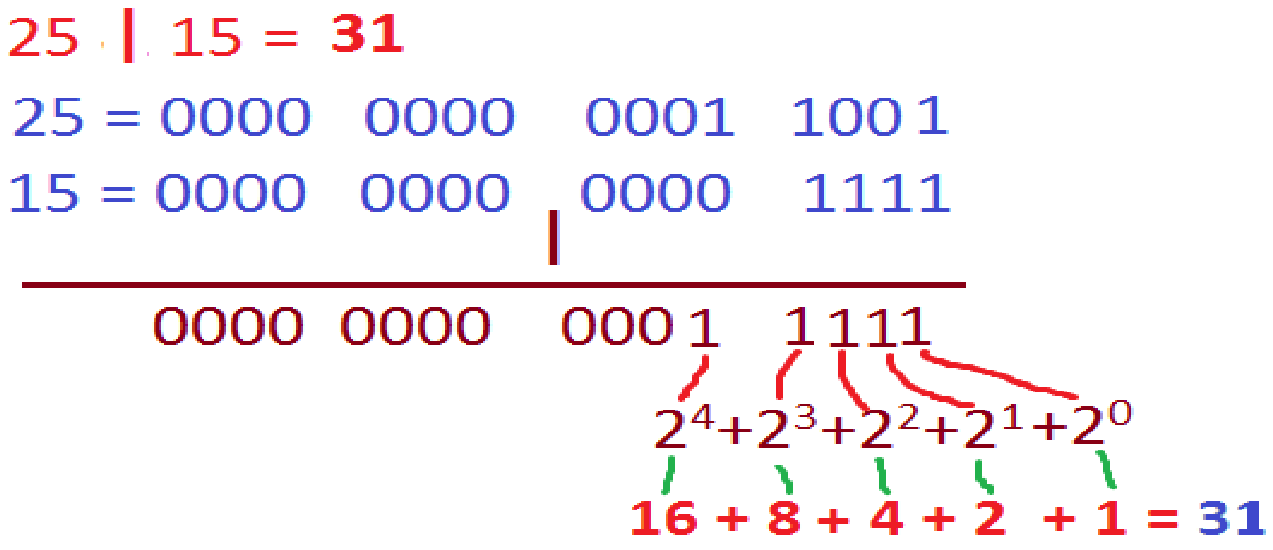
```
25 = 0000  0000  0001  1001
15 = 0000  0000  0000  1111
                &
---------------------------
     0000  0000  0000  1001
```

$$2^3 + 2^0$$

$$8 + 1 = \mathbf{9}$$

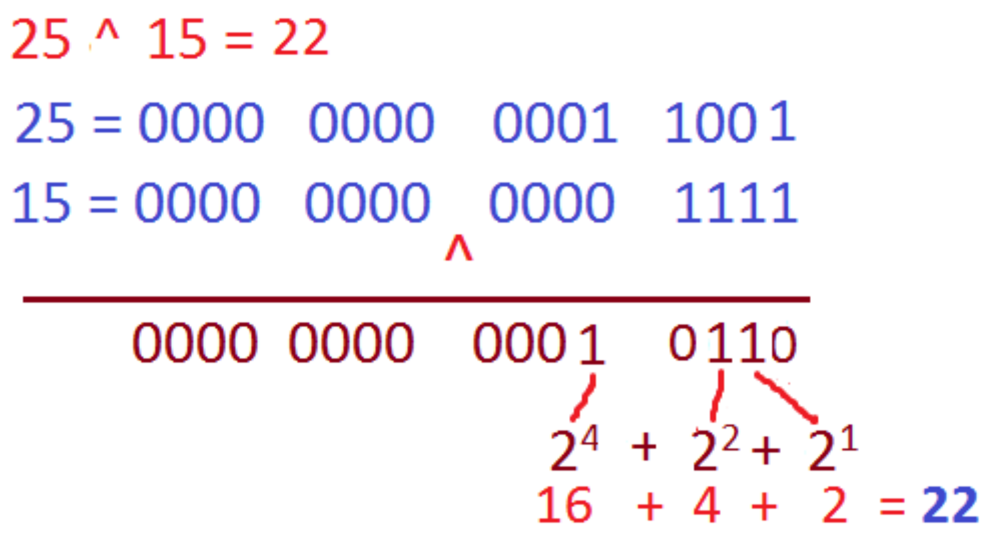
| - Bitwise or: In this both bits are 0’s then result bit is 0. Otherwise result bit is 1.

Eg: 25 | 15 = 31



^ - XOR [Exclusive OR]: In this both bits are same then result bit is 0. Otherwise result bit is 1.

Eg: 25 ^ 15 = 22



```
25 ^ 15 = 22
25 = 0000  0000  0001  1001
15 = 0000  0000  0000  1111
                ^
____________________________
     0000  0000  0001  0110
```

$2^4 + 2^2 + 2^1$

$16 + 4 + 2 = 22$

~ - Compliment operator: In compliment operation the bits are complimented. i.e. 1’s become 0’s and 0’s become 1’s. Due to this +Ve no becomes –Ve and –Ve no becomes +Ve.

Formula: -(n+1)

eg: **~25 ➔ -26**

$$25 = \boxed{\begin{matrix}0000\ 0000\\1111\ 1111\end{matrix}}\ \begin{matrix}0001\ 1001\\1110\ 0110\end{matrix}$$

$$-128+\underline{64+32+4+2}=-26$$

$$-128 + 102 = -26$$

```
25 = 0000 0000 0001 1001
~   = 1111 1111 1110 0110
                 \    | \
                 5    2 1
```

2+4+32+64+128+256+512+1024+2048+4096+8192+16384-32768=-26

```
~-25 = 0000 0000 0001 1001          1    0    0    01
1's ~ = 1111 1111 1110 0110         0    1    0     1
2's ~ = 0000 0000 0000 0001         1    1    0    10
        1111 1111 1110 0111
                   /    \
                 2^4    2^3
                 16+8=24
```

Note: When starting bit is 1 given no is –Ve.

Eg: ~-25 → +24

~-25 = +24

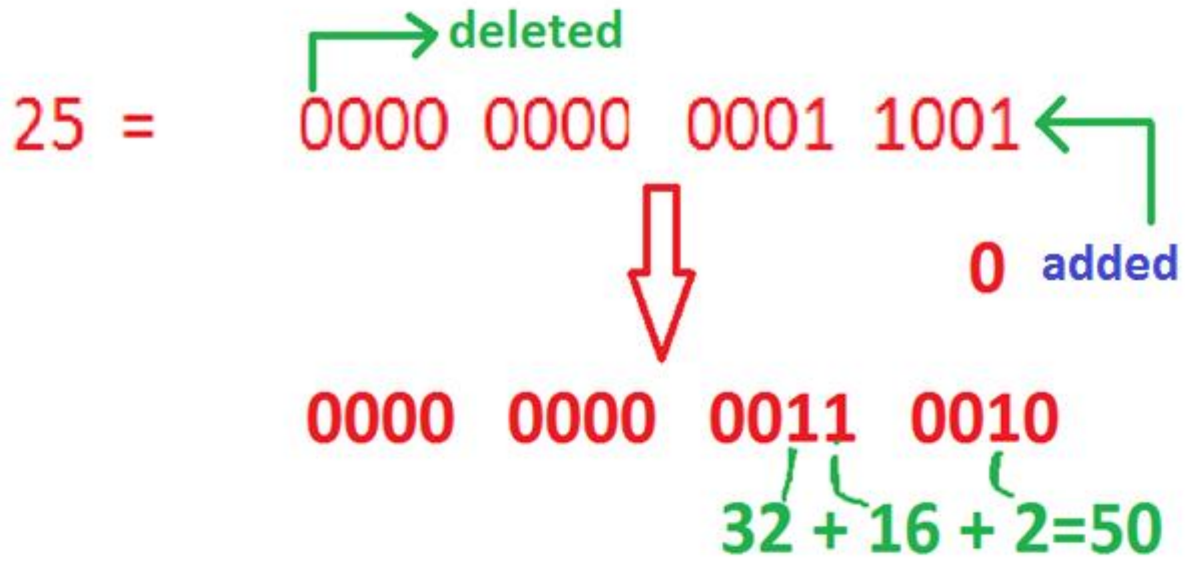
```
25 = 0000 0000  0001 1001
     1111 1111  1110 0110   <== 1's compliment
                      +1    <== 2's Compliment
     ---------------------
     1111 1111  1110 0111
  ~  0000 0000  0001 1000
                   ↓   ↓
                  16+8=24
```

<< - left shift operator:

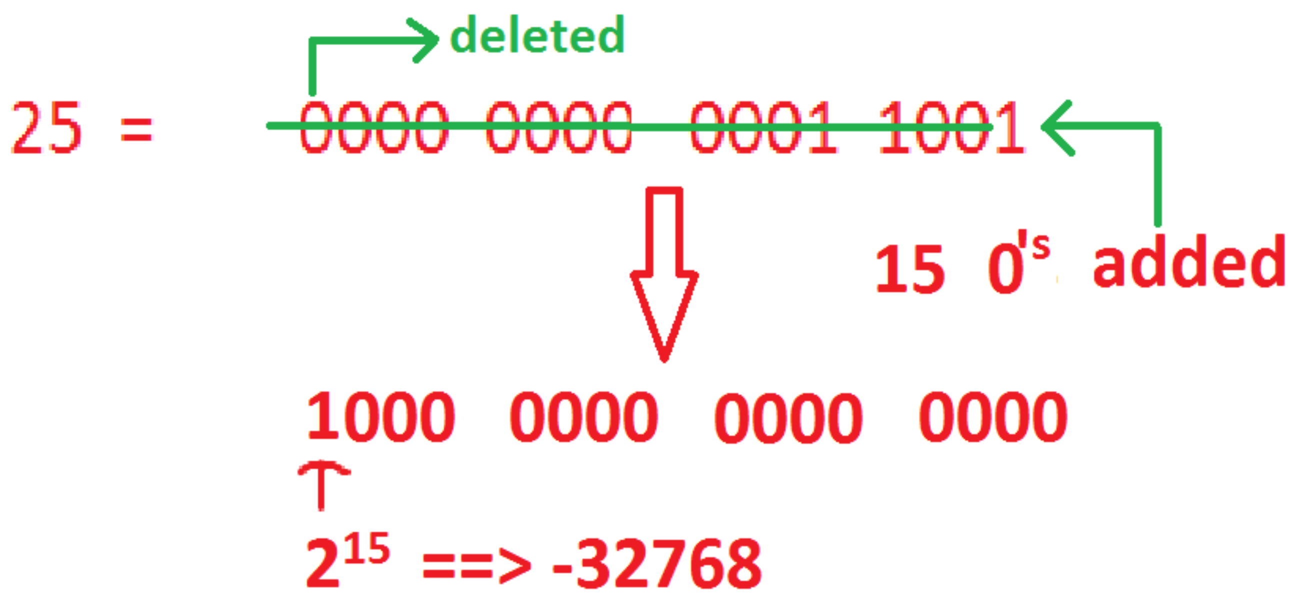
In left shift operation, the specified no of bits are deleted from left side and the same no of zeros added on right side. In left shift operation, most probably the value is multiplied with 2 that no of times.

Eg:25<<1=50, 25<<2=100, 25<<15 =-32768, 25<<16=0

eg: $25<<1=50$


deleted
25 = 0000 0000 0001 1001
0 added
0000 0000 0011 0010
32 + 16 + 2=50

eg: 25<<15 = -32768



Note: When starting bit 1 no is negative.

>> - Right shift operator:

In right shift operation, the bits are moved to right side i.e. the specified no.of bits are deleted from right side and same no.of zero's are added left side. Due to this always the number is divided with 2 that no of times.

Eg:**25>>1=12, 25>>2=6, 25>>3=3, 25>>4=1,25>>5=0** eg: 25 >>1 = 12

deleted

25 = 0000 0000 0001 1001

0 added

$\Downarrow$

0000 0000 0000 1100

8 + 4 = 12

eg: 25 >> 5 = 0

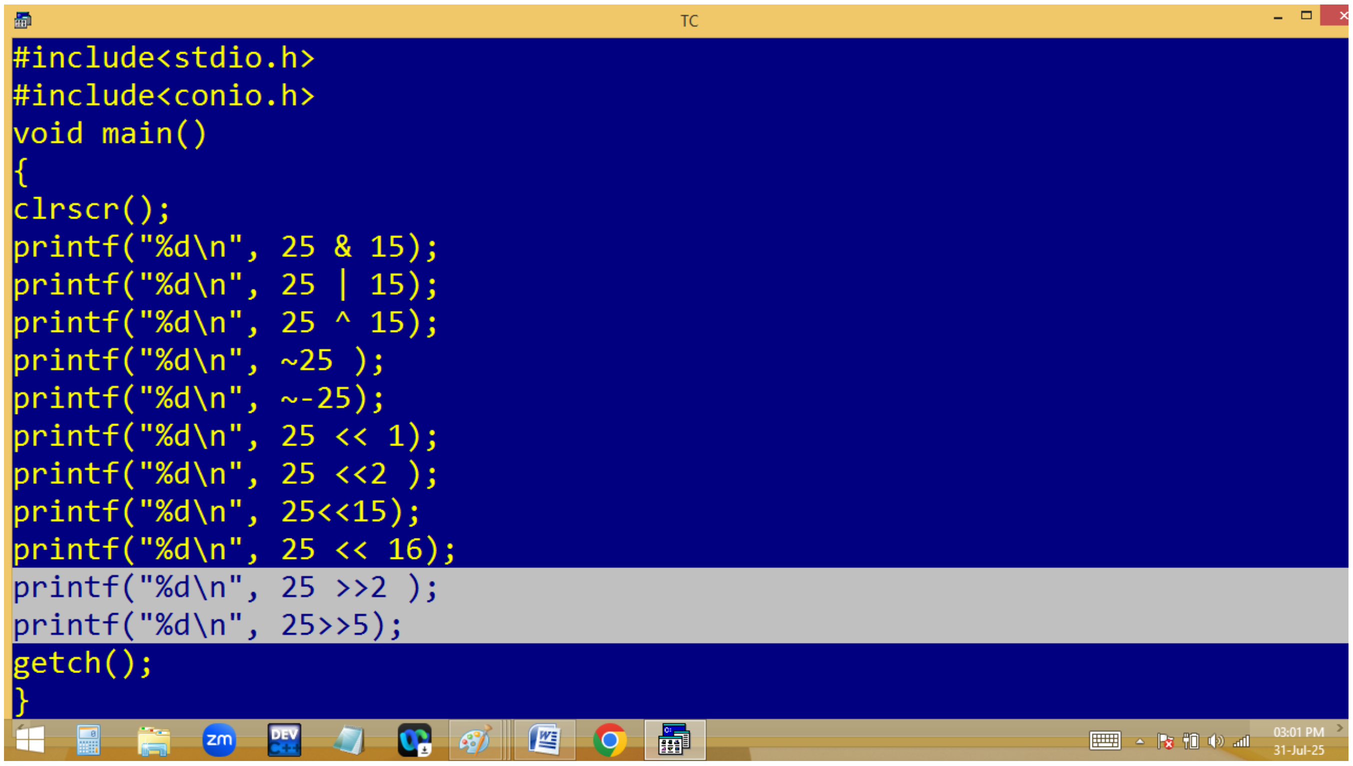
deleted

25 = 0000 0000 000$\boxed{1\ 1001}$

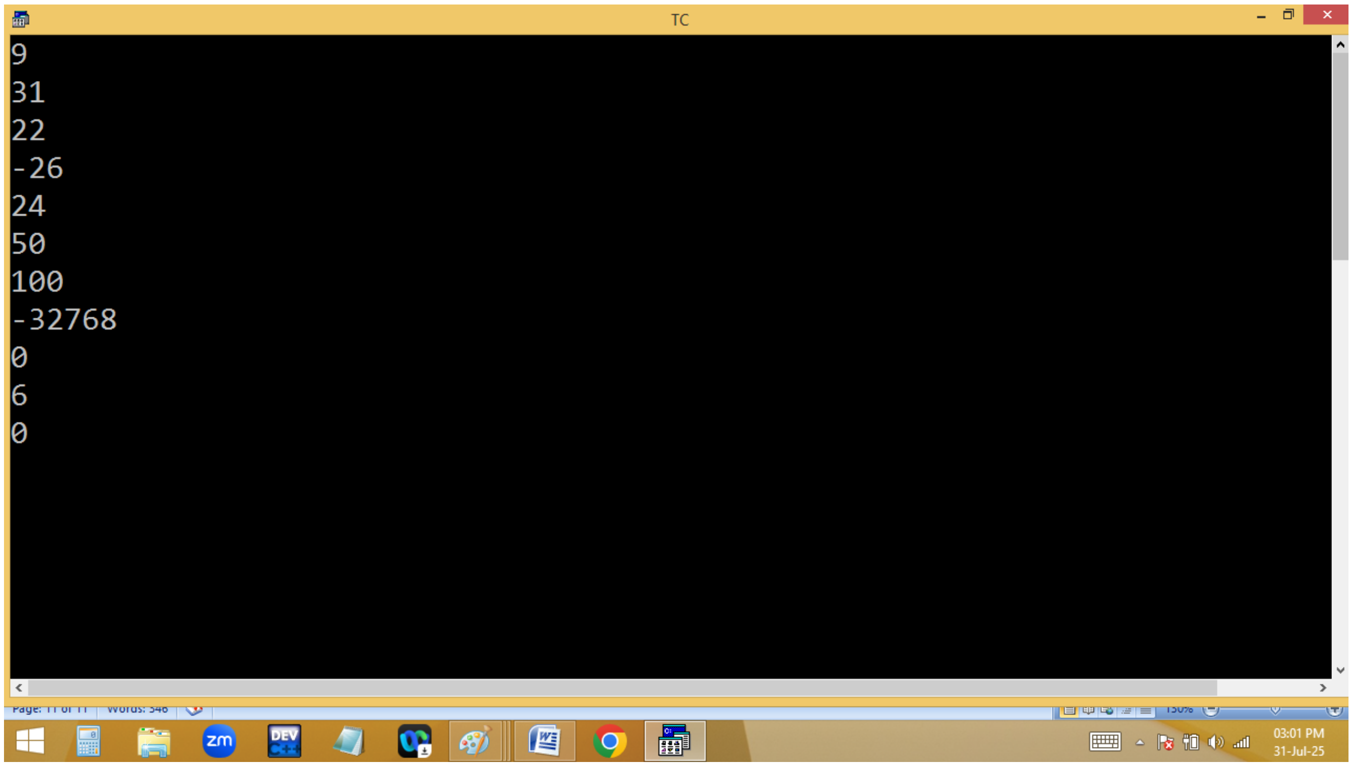
5 0's added

$\Downarrow$

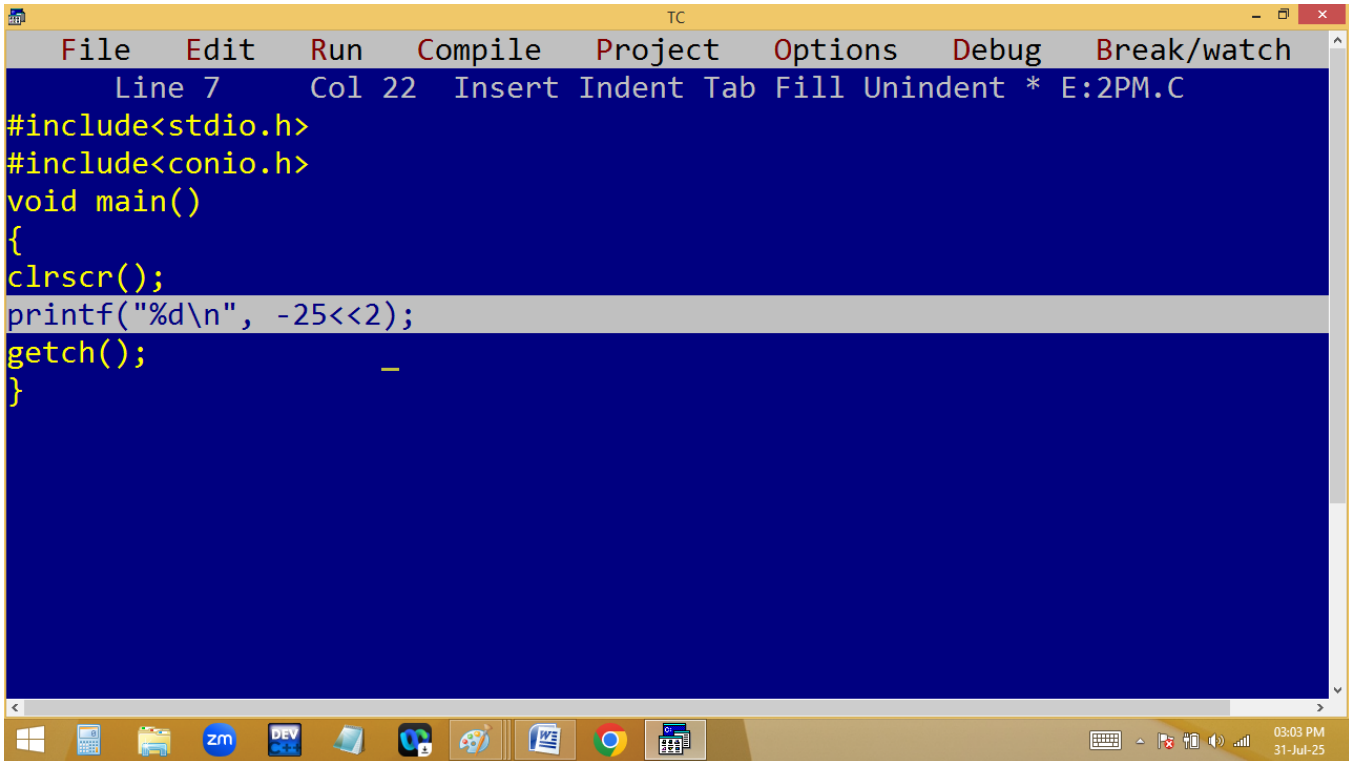
0000 0000 0000 0000 = 0



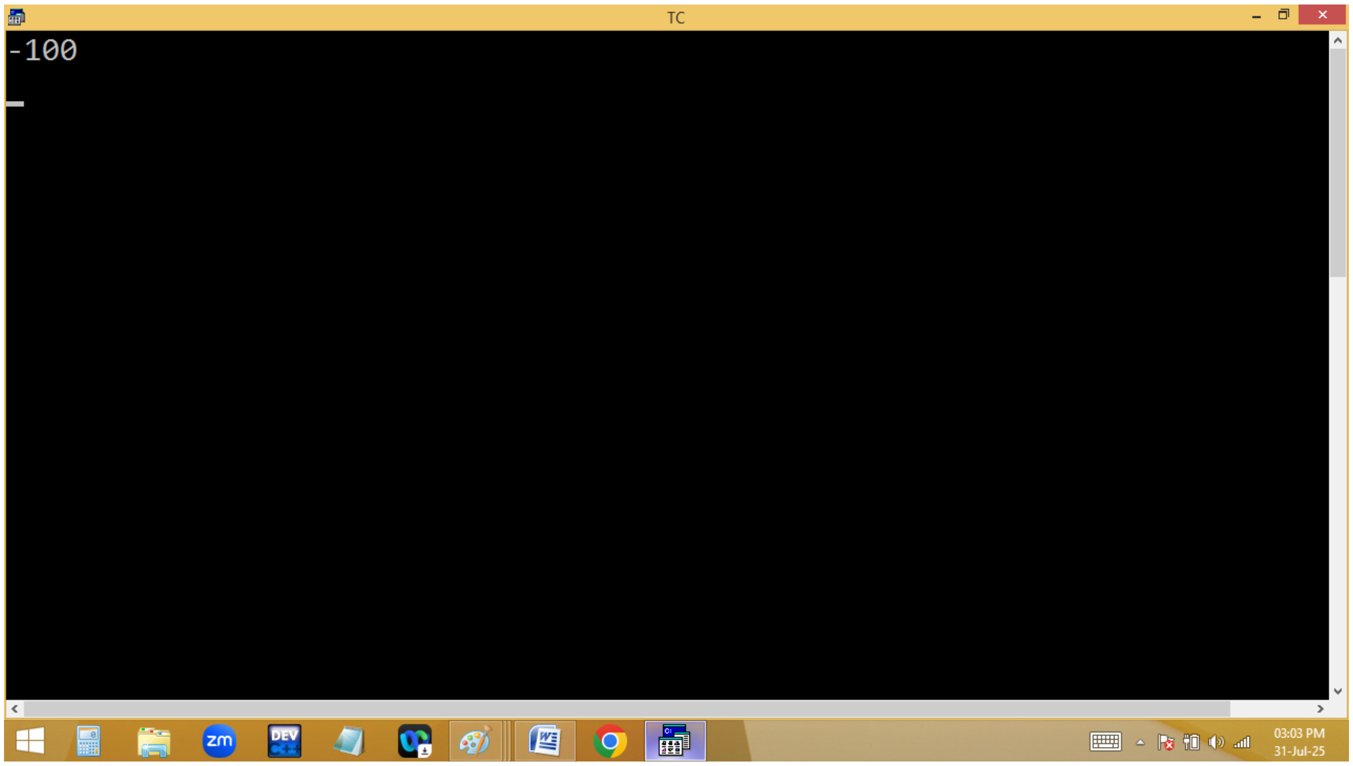
```
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
printf("%d\n", 25 & 15);
printf("%d\n", 25 | 15);
printf("%d\n", 25 ^ 15);
printf("%d\n", ~25 );
printf("%d\n", ~-25);
printf("%d\n", 25 << 1);
printf("%d\n", 25 <<2 );
printf("%d\n", 25<<15);
printf("%d\n", 25 << 16);
printf("%d\n", 25 >>2 );
printf("%d\n", 25>>5);
getch();
}
```



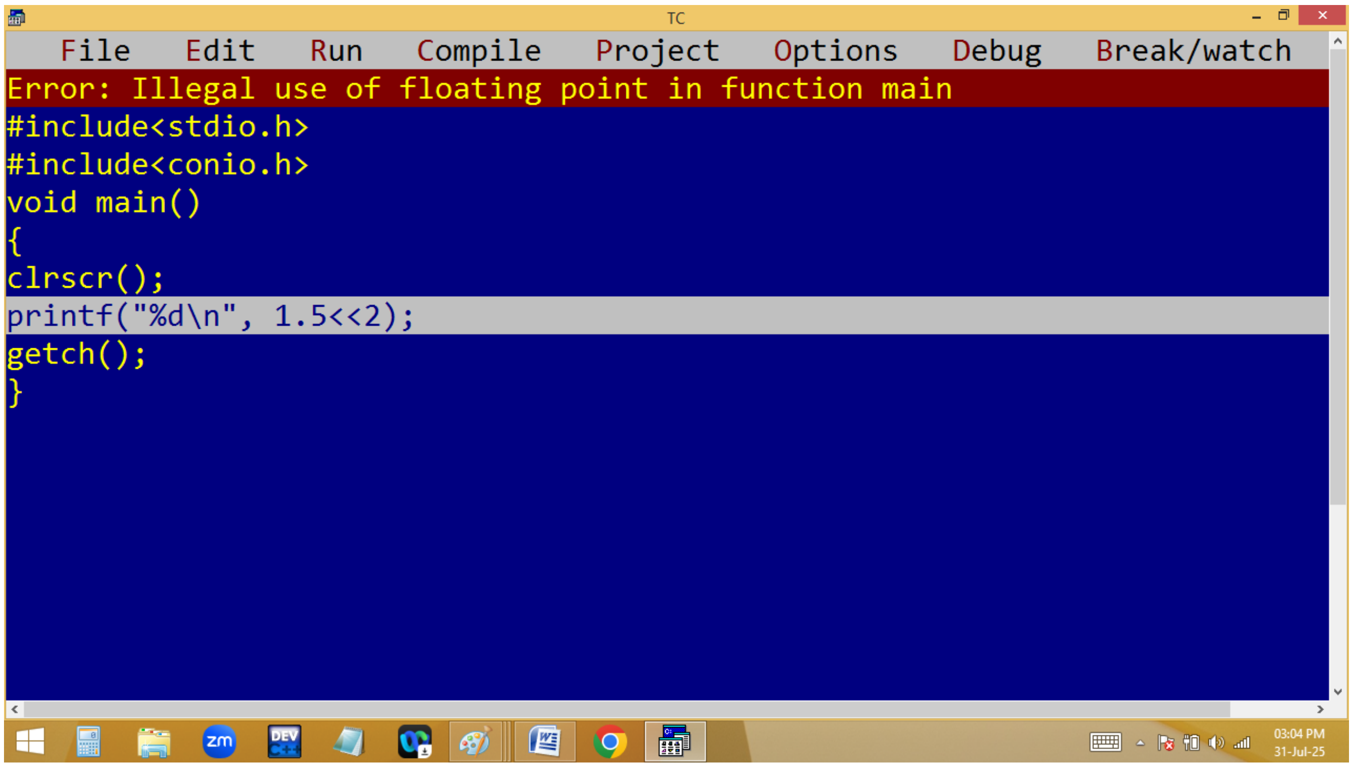
```
9
31
22
-26
24
50
100
-32768
0
6
0
```



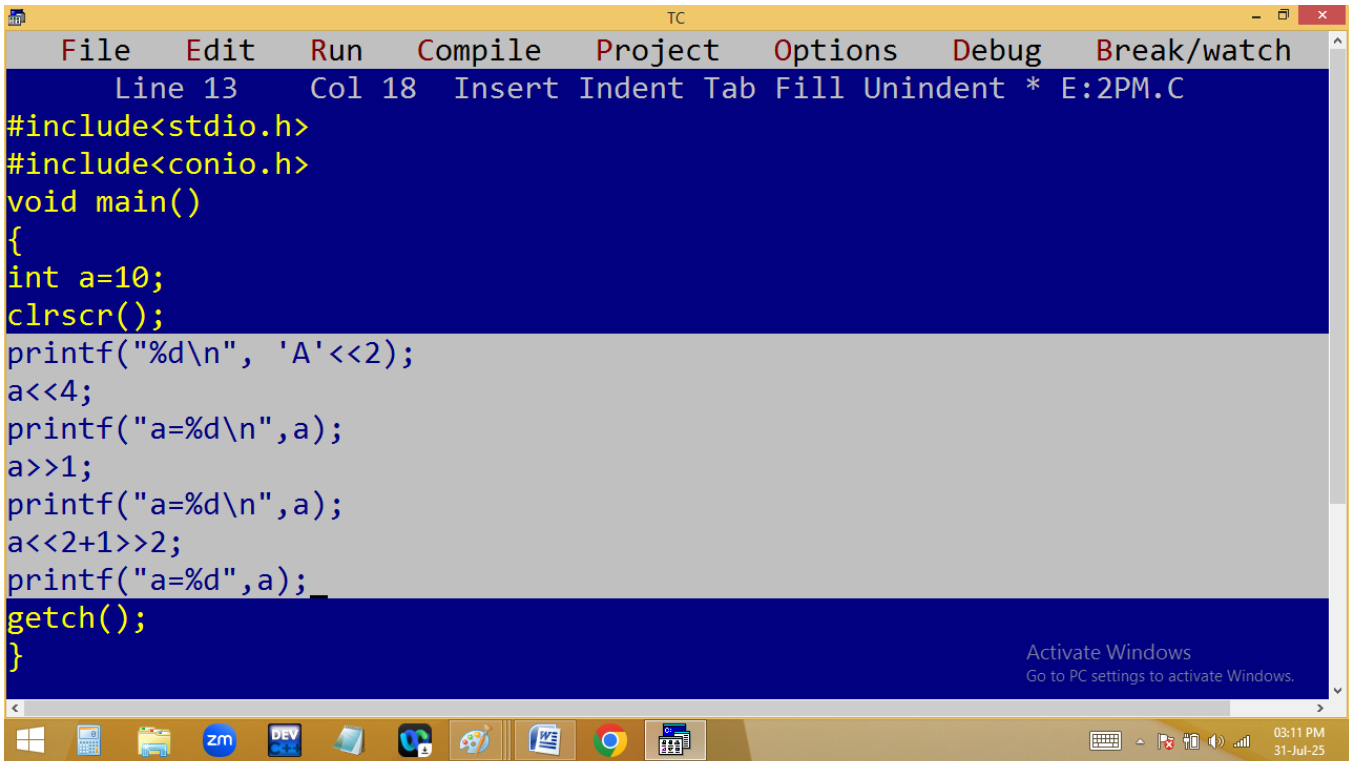
```
     Line 7     Col 22  Insert Indent Tab Fill Unindent * E:2PM.C
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
printf("%d\n", -25<<2);
getch();
}
```



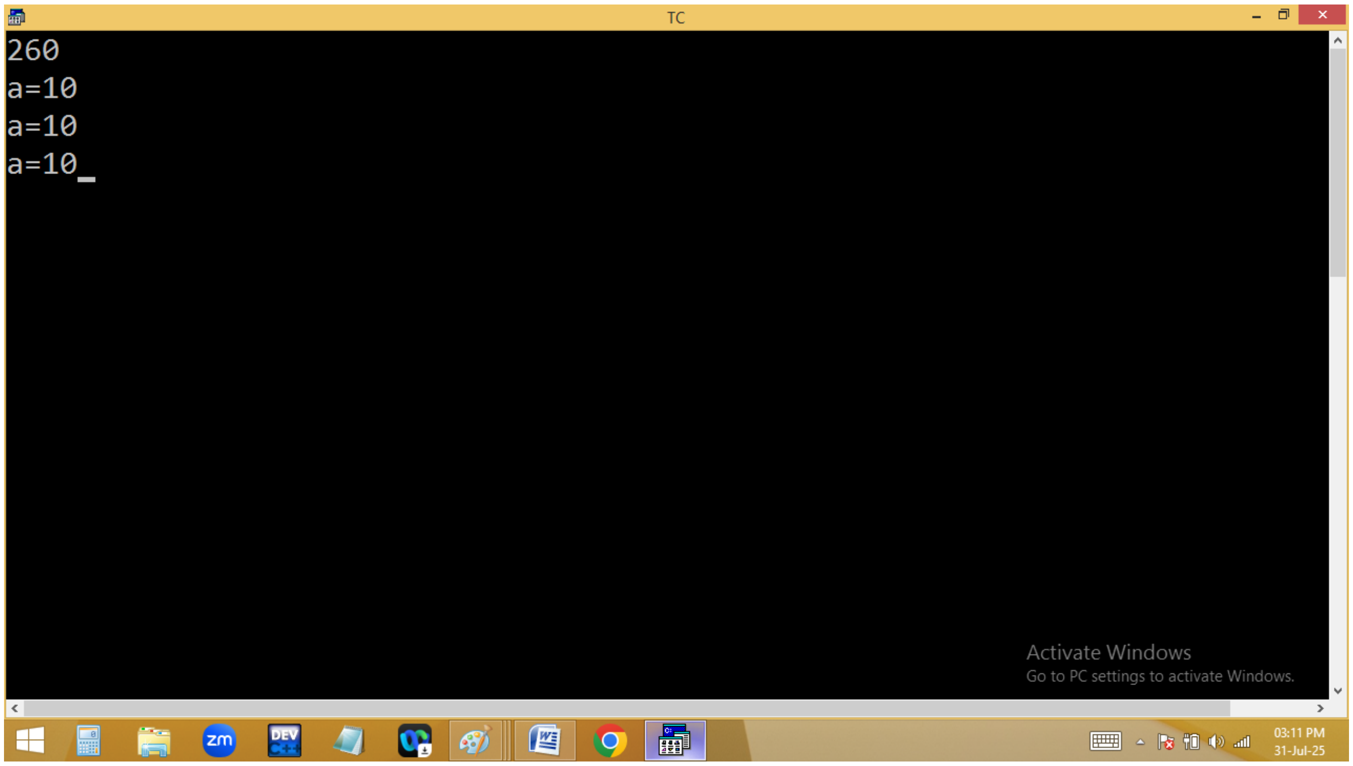
```
-100
```



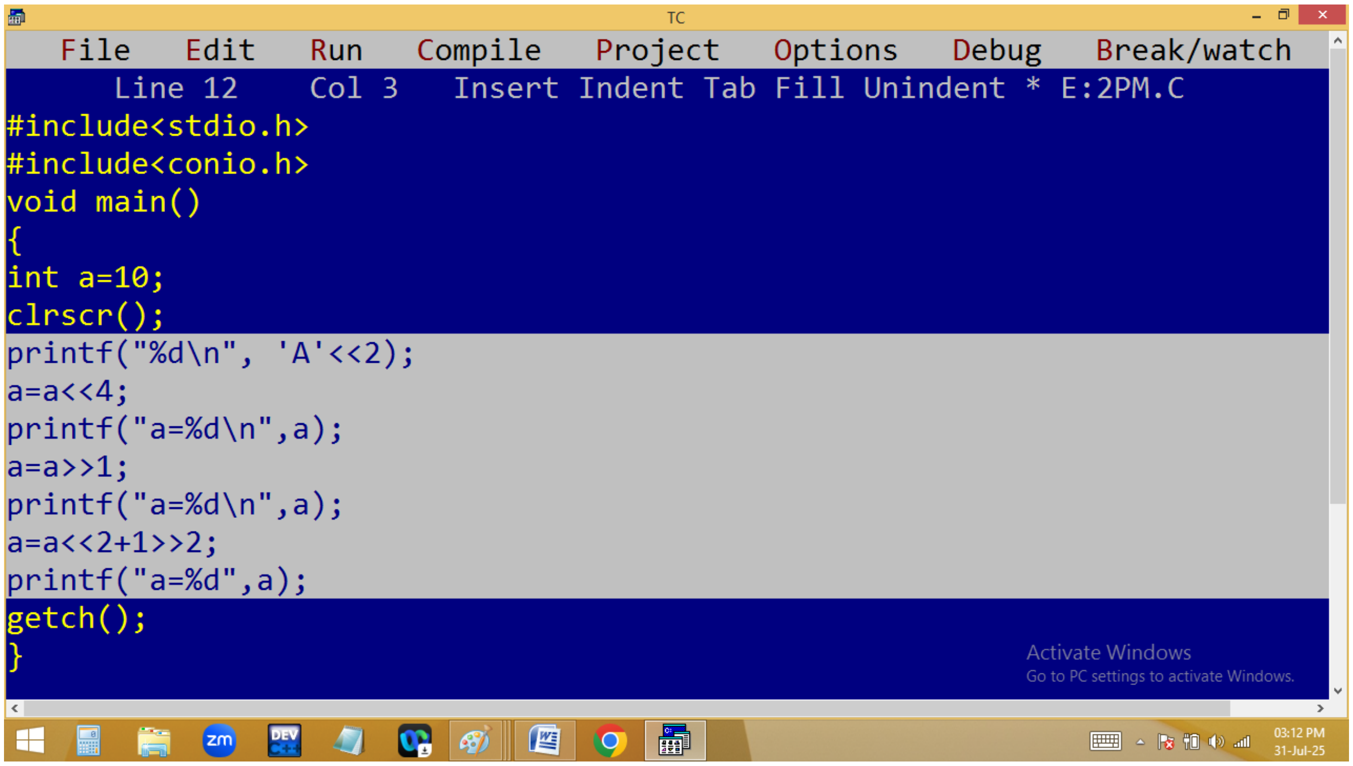
```
Error: Illegal use of floating point in function main
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
printf("%d\n", 1.5<<2);
getch();
}
```


TC
File Edit Run Compile Project Options Debug Break/watch
Line 13 Col 18 Insert Indent Tab Fill Unindent * E:2PM.C
Activate Windows
Go to PC settings to activate Windows.
03:11 PM
31-Jul-25

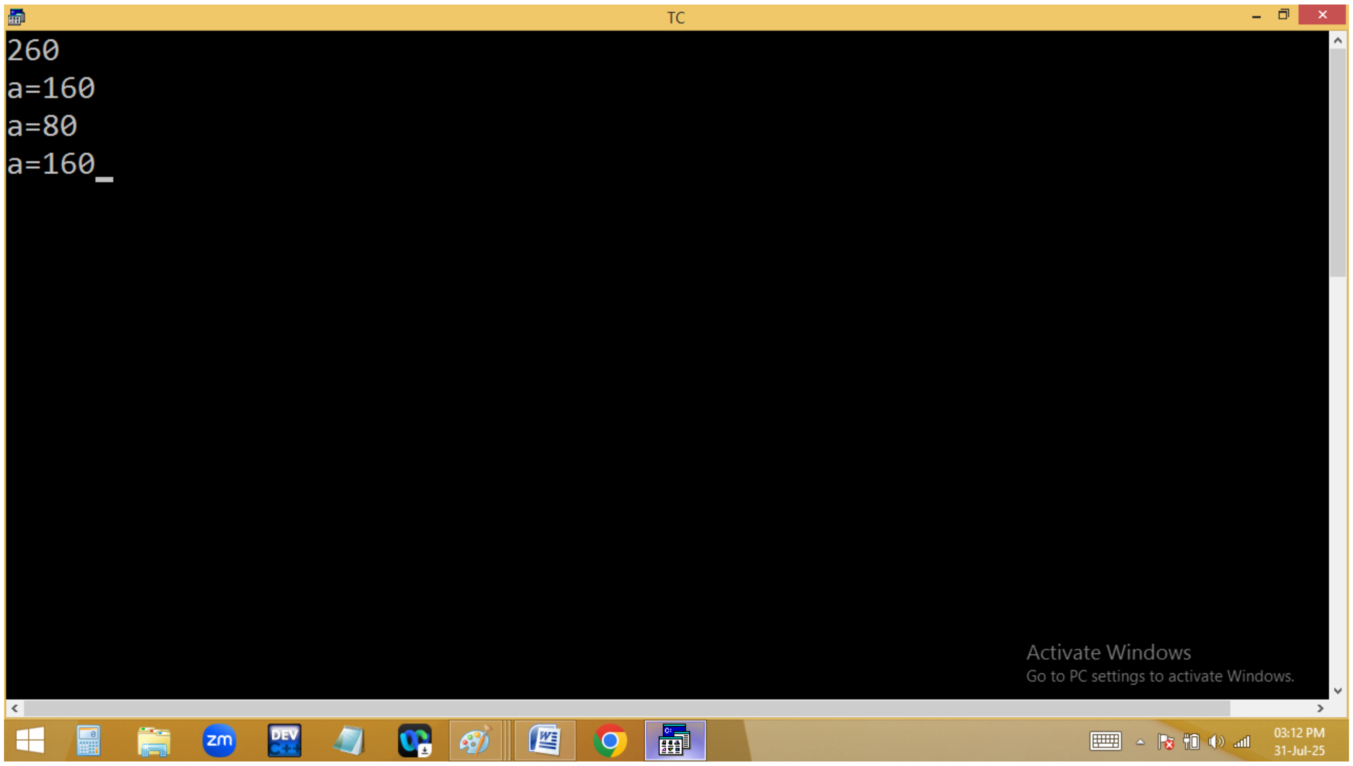
```
#include<stdio.h>
#include<conio.h>
void main()
{
int a=10;
clrscr();
printf("%d\n", 'A'<<2);
a<<4;
printf("a=%d\n",a);
a>>1;
printf("a=%d\n",a);
a<<2+1>>2;
printf("a=%d",a);_
getch();
}
```



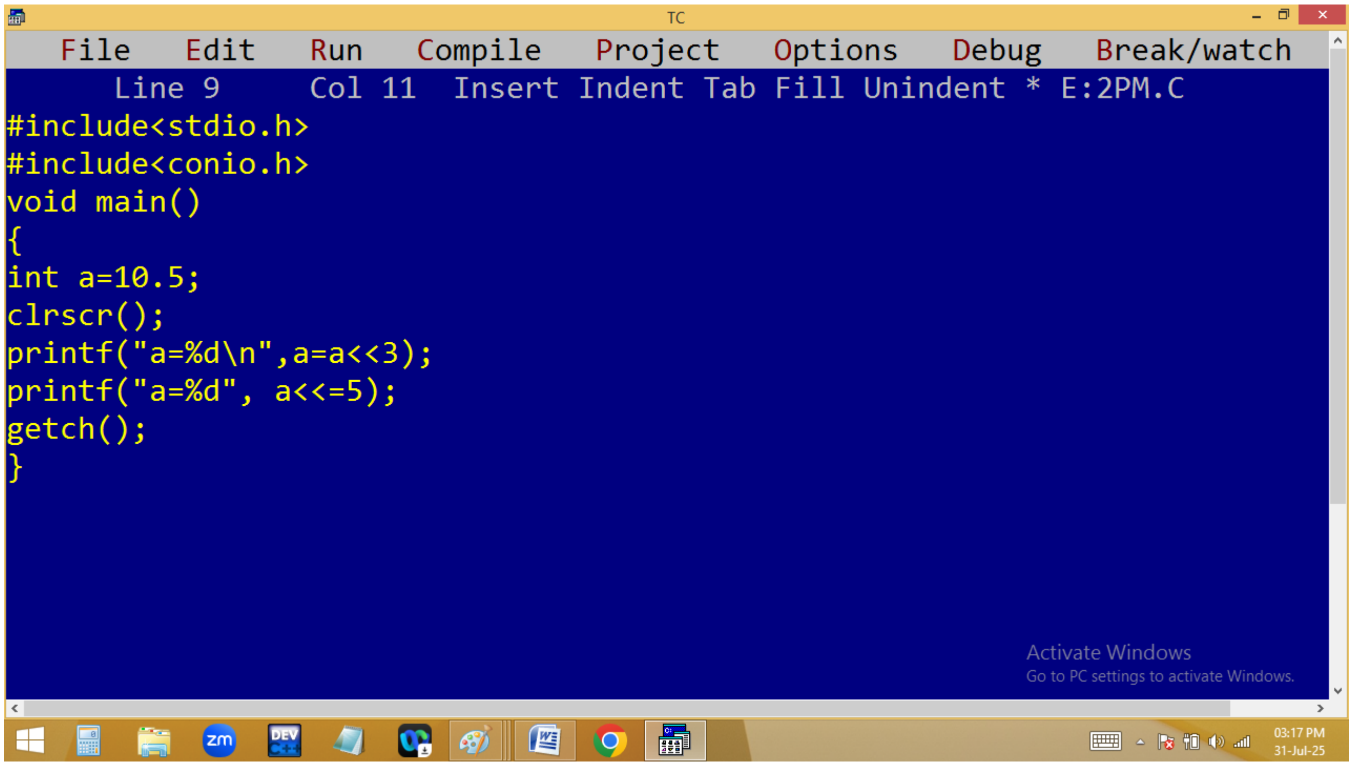
```
260
a=10
a=10
a=10_
```


TC
File Edit Run Compile Project Options Debug Break/watch
Line 12 Col 3 Insert Indent Tab Fill Unindent * E:2PM.C
Activate Windows
Go to PC settings to activate Windows.
03:12 PM
31-Jul-25

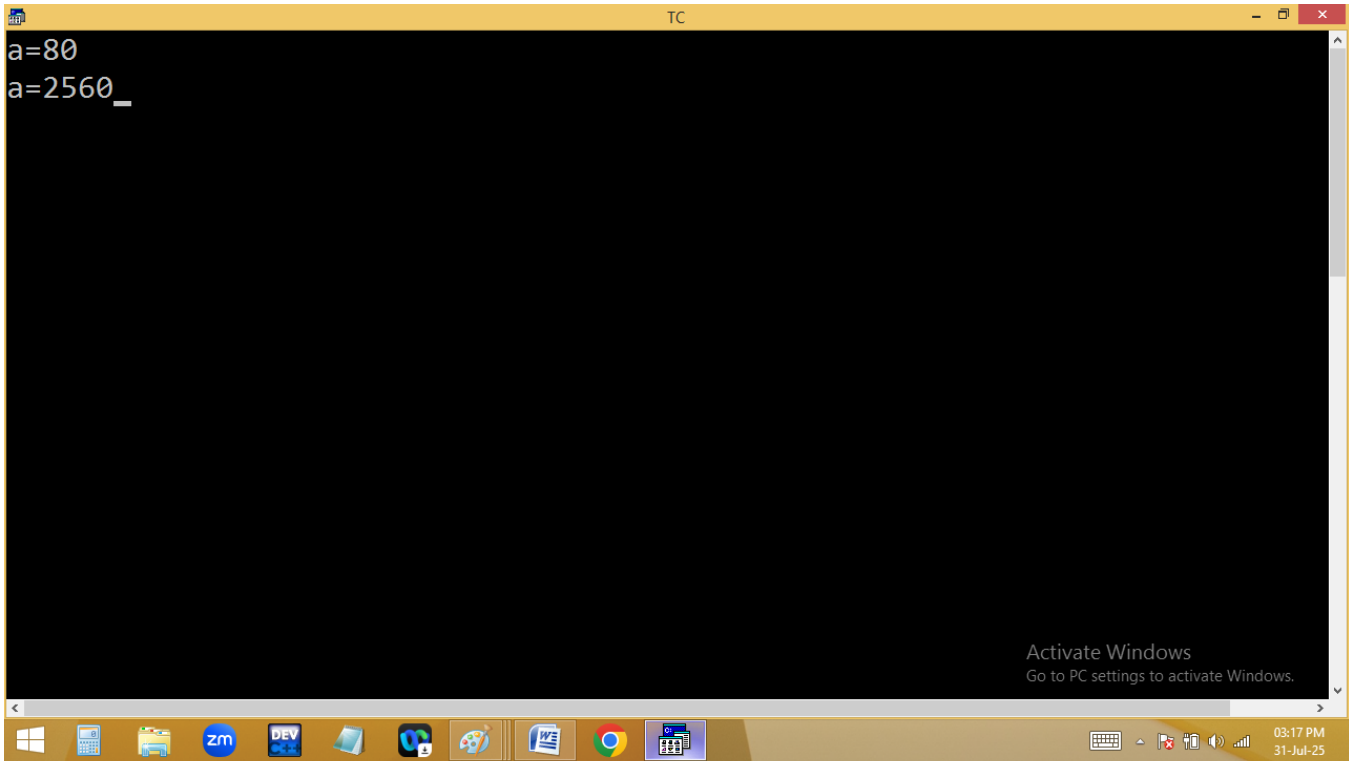
```c
#include<stdio.h>
#include<conio.h>
void main()
{
int a=10;
clrscr();
printf("%d\n", 'A'<<2);
a=a<<4;
printf("a=%d\n",a);
a=a>>1;
printf("a=%d\n",a);
a=a<<2+1>>2;
printf("a=%d",a);
getch();
}
```



```
260
a=160
a=80
a=160_
```


TC
File Edit Run Compile Project Options Debug Break/watch
Line 9 Col 11 Insert Indent Tab Fill Unindent * E:2PM.C
Activate Windows
Go to PC settings to activate Windows.
03:17 PM
31-Jul-25

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a=10.5;
clrscr();
printf("a=%d\n",a=a<<3);
printf("a=%d", a<<=5);
getch();
}
```



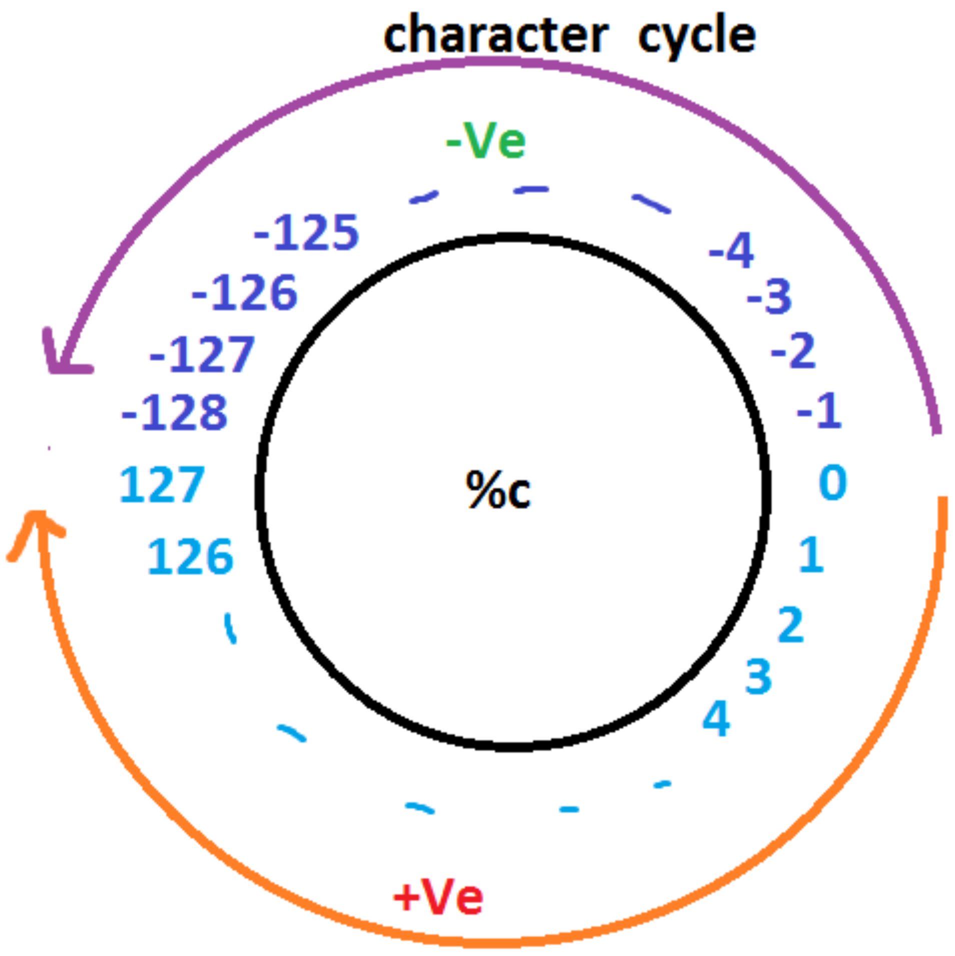
```
a=80
a=2560_
```

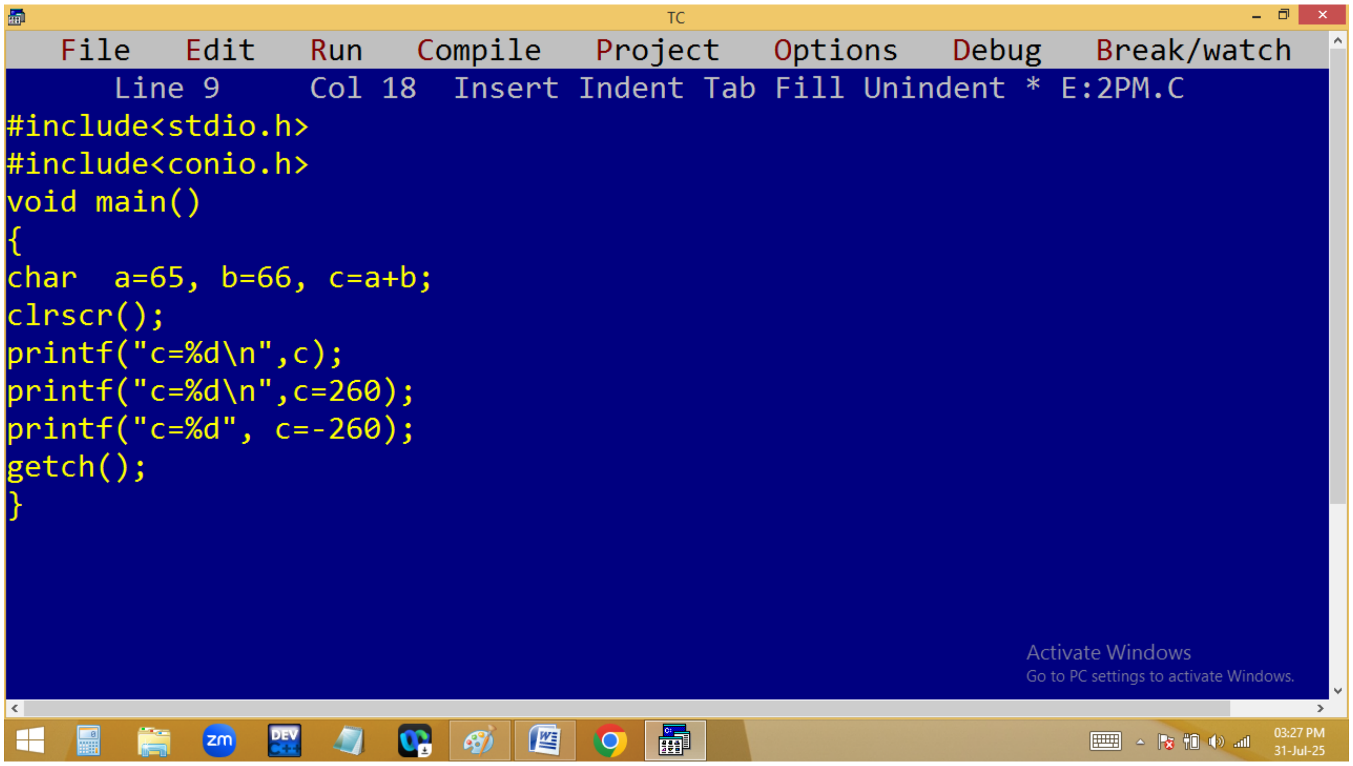
Data type cycles:

character data type cycle:

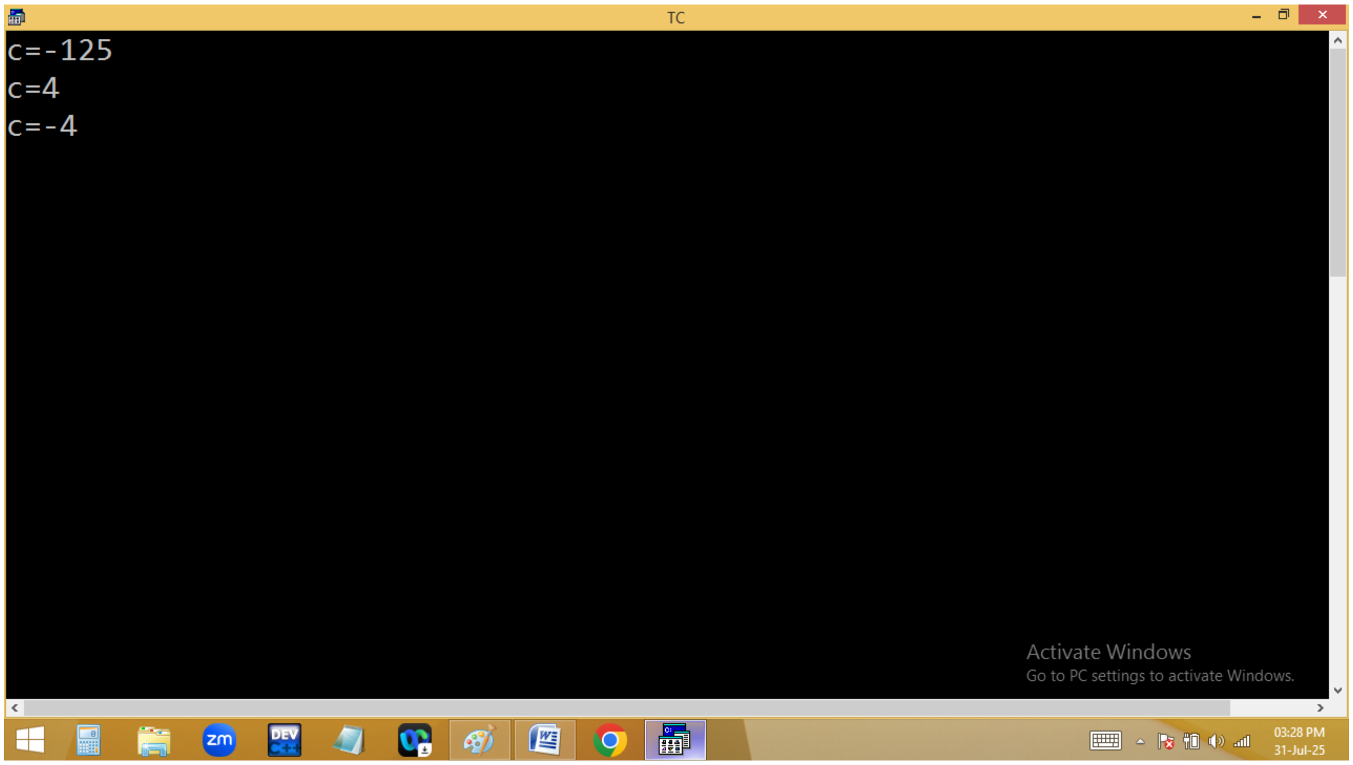
C & C++ are using ASCII character set, which comes with 256 characters. They are divided into 2 types.

1. Signed char ➔ -128 to +127
2. Unsigned char ➔ 0 to 255


character cycle
-Ve
-125
-126
-127
-128
127
126
-4
-3
-2
-1
0
1
2
3
4
%c
+Ve


TC
File Edit Run Compile Project Options Debug Break/watch
Line 9 Col 18 Insert Indent Tab Fill Unindent * E:2PM.C
Activate Windows
Go to PC settings to activate Windows.
03:27 PM
31-Jul-25

```
#include<stdio.h>
#include<conio.h>
void main()
{
char  a=65, b=66, c=a+b;
clrscr();
printf("c=%d\n",c);
printf("c=%d\n",c=260);
printf("c=%d", c=-260);
getch();
}
```



```
c=-125
c=4
c=-4
```

a = 65

b = 66

c=131

256

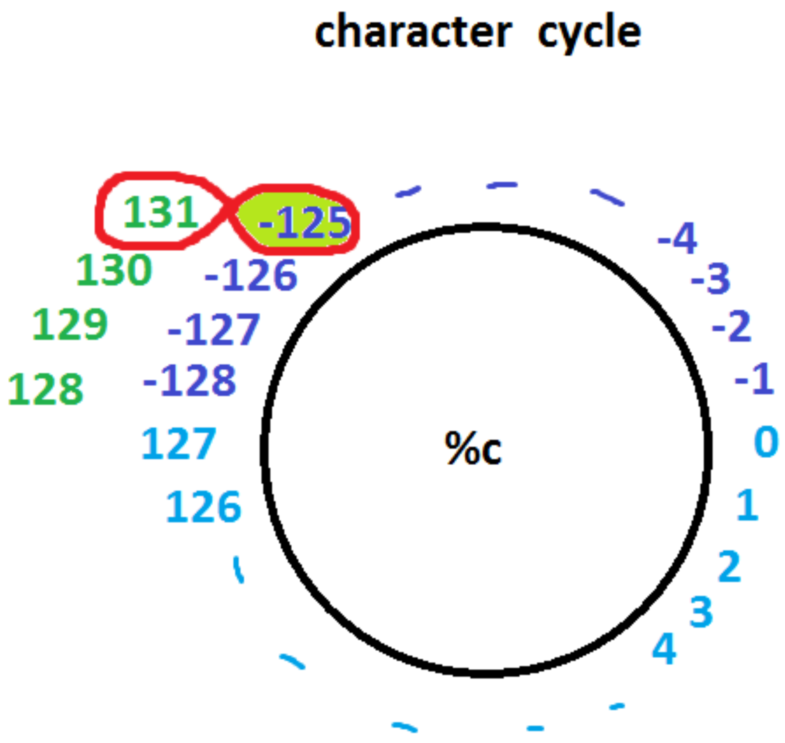
-131

-125

131

-128 + 3 = -125

3



Int cycle:

C & C++ are using 16 bit compilers and in 16 bit compilers int size is 2 bytes i.e. 2^{16} ➔ 65536

This 65536 divided into 2 types.

1. **Signed int ➔ -32768 to +32767**
2. **Unsigned int ➔ 0 to 65535**

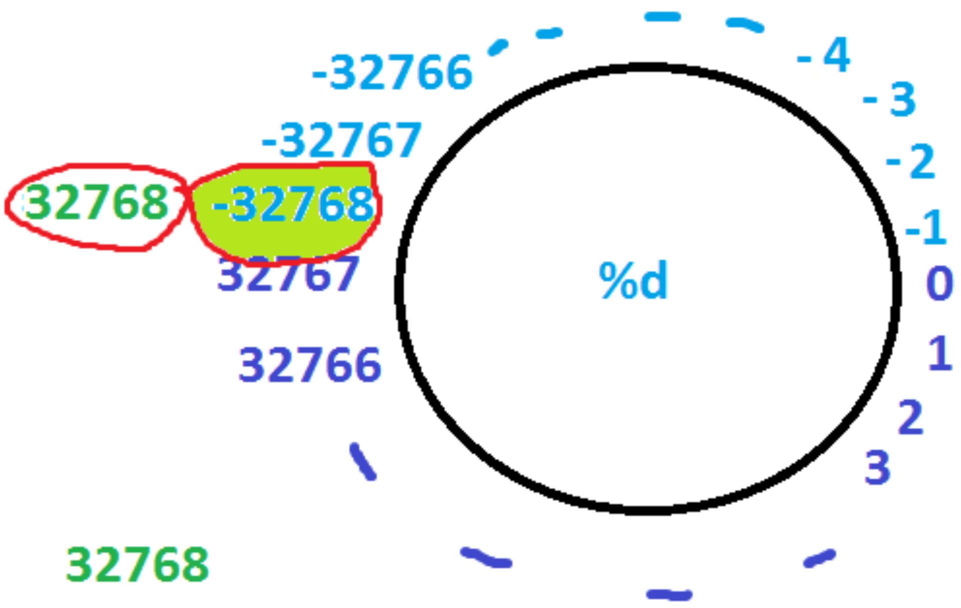
java & .net are using 32 bit compilers and in 32 bit compilers int size is 4 bytes i.e. 2^{32} ➔ 4294967296

This 4294967296 divided into 2 types.

1. **Signed int ➔ -2147483648 to +2147483647**
2. **Unsigned int ➔ 0 to 4294967295**

int cycle

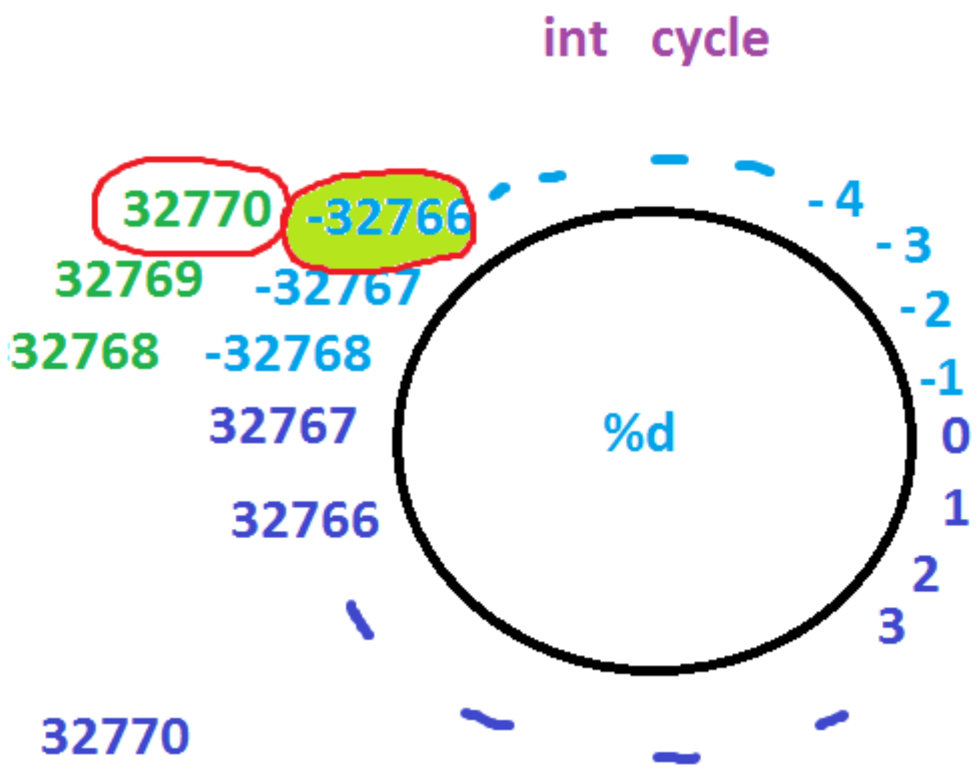
a=32768



$$\begin{array}{r} 65536 \\ \underline{-32768} \\ \underline{-32768} \end{array}$$

$$\begin{array}{r} 32768 \\ \underline{-32768} \\ \underline{\quad 0} \end{array} + 0 = -32768$$

a = 32770



65536

<u>-32770</u>

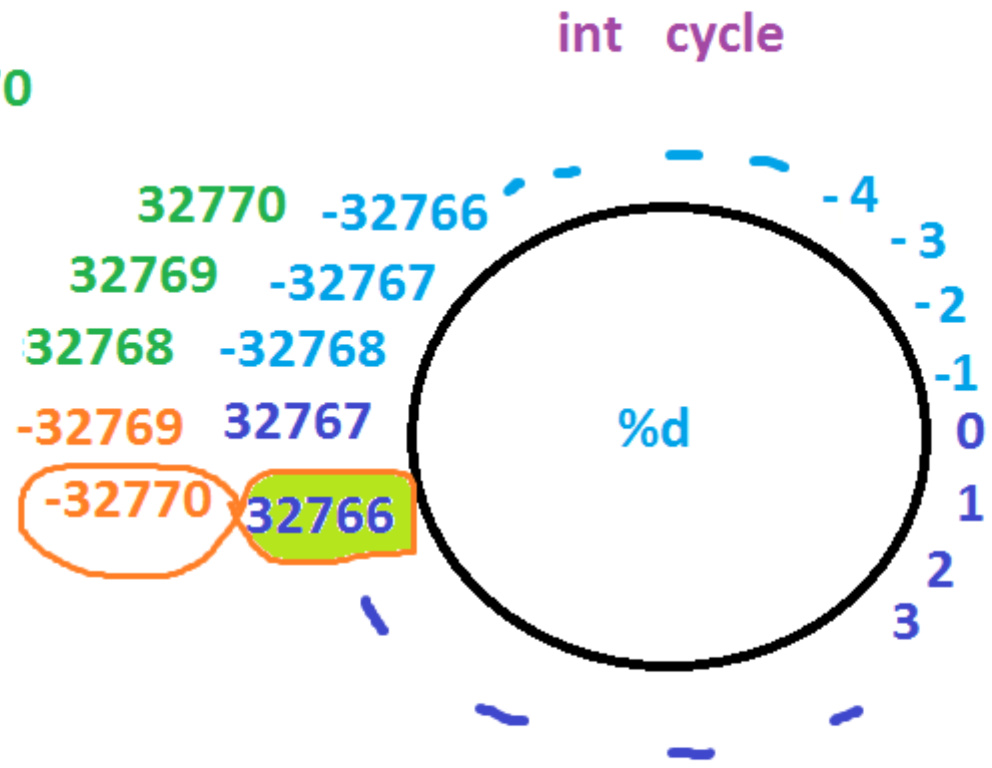
<u>-32766</u>

32770

<u>-32768</u> + 2 = -32766

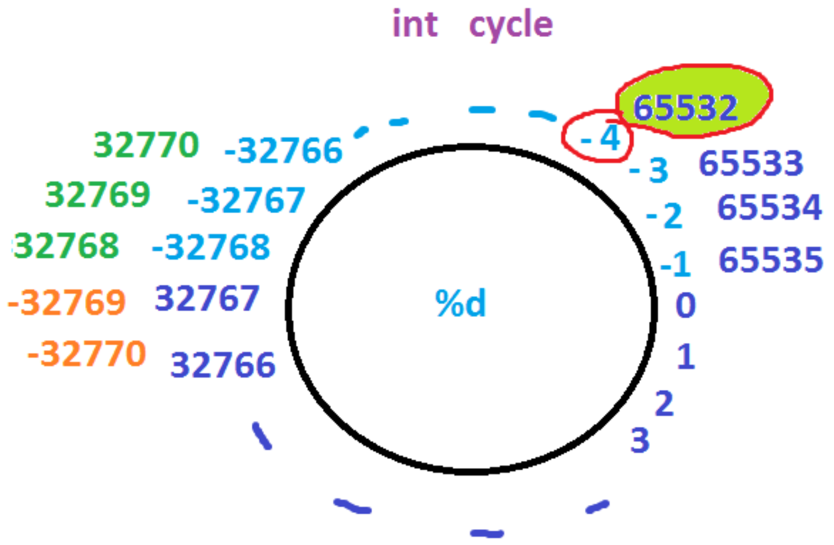
2

a = -32770



$$\begin{array}{r} 65536 \\ \underline{-32770} \\ \underline{32766} \end{array}$$

a = -4



$$\begin{array}{r} 65536 \\ \underline{-4} \\ \underline{65532} \end{array}$$