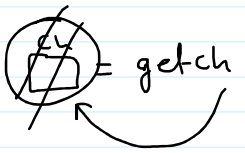
Day 01

17 March 2025 06:48 AM

AT & T



stdio.h ← { scanf, getchar

Conio.h ← { getche, getch

char ch;

scanf() → int | float | char | string

→ scanf("%c", &ch);

getchar(); → input only a char

ch = getchar();

getche() → input only a char

ch = getche();

getch()

ch = getch();

Enter key → Display input char

Enter ← input char

```
#include<stdio.h>
#include<conio.h>
int main()
{
    char ch;
    printf("enter a char:");
    //scanf("%c",&ch);
    // ch = getchar();
    // ch = getche();
    ch = getch();
    printf("\nChar = %c",ch);
    return 0;
}
```

output

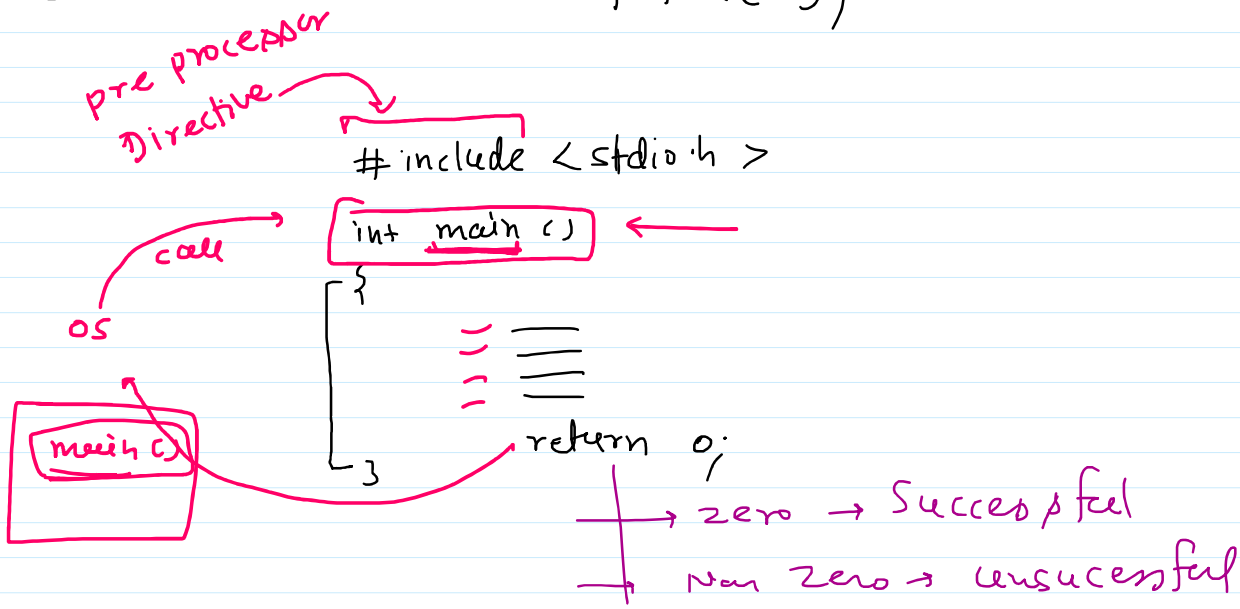
char ch = 'A';

stdio.h { printf(); → int | float | char | string

printf("%c", ch);

stdio.h { printf("%c", ch);
putchar();
putchar(ch);

conio.h ← putch()
putch(ch);



Tokens :→

① Keywords (Reserved words)
② Identifiers (Name)
③ Literals (constant)
④ Operators (+, −, *, /)
⑤ seperators (; , ( ) { } :)

① ② ③
int a = 10 ; ← ⑤
④

① Keywords :→

32 keywords

② Identifiers → name of a variable, function, pointer array, string, structure, union.

Rules :→ ① It contains alphabets [a-z, A-Z), digits [0-9],

Identifiers :→

① It contains alphabets [a-z, A-Z), digits [0-9], underscore (_) only

② It cannot contain space, any other symbols.

③ must not start with any digit

int as; ✓

int 5a; ✗

④ must not match with any keywords

int if; ✗

⑤ case sensitive

int If; ✓

int a; ✓

int A; ✓

⑥ length → 32 char

③ Literals :→

① integer ② Floating ③ character ④ string

① Integer

int short long long long

short a;

or

short int a;

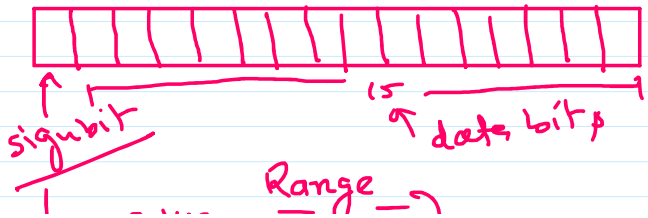
%hd

a [-2bytes-]

bits → 16 bits

d

$2^{16-1} \rightarrow 2^{15} \rightarrow 32768$



Range →

signbit

→ 0 → +ve
→ 1 → -ve

Range → −32768 to +32767

%ld ← long

long a; → a [— 4 bytes —]

or

long int a;

2^{32}

2^{31} → 2147483648

Range

−2147483648 to +2147483647

%d ← int → Compiler Dependent

16 bits	32 bits	64 bits
2 bytes	4 bytes	8 bytes

signed | unsigned :→

default → signed int a

unsigned int a; → only +ve

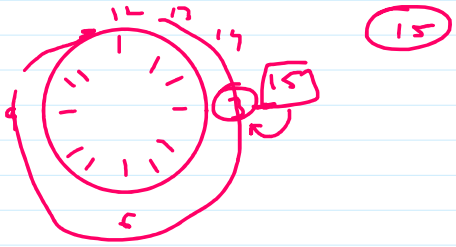
unsigned short a; a [— 2 —]

−32768 to +32767

0 to 65535

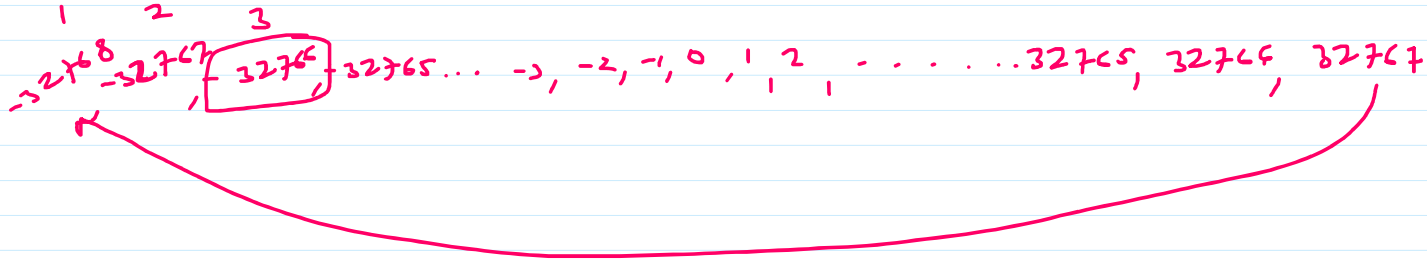
signed int → %d	unsigned int → %u | %i
signed short → %hd	unsigned short → %hu
signed long → %ld	unsigned long → %lu

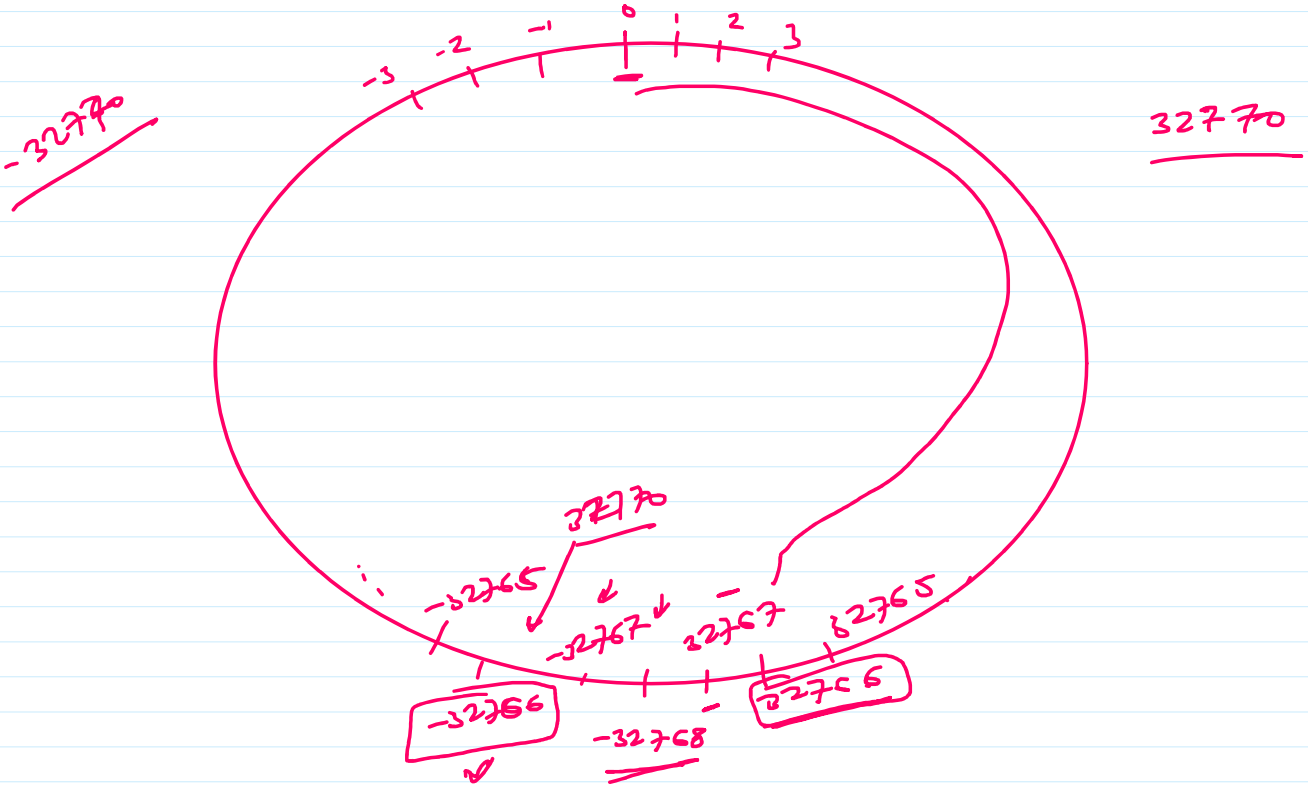
Circular rotation →



short a = 32770;

printf("%hd", a); → -32766





floating literals :→

float double long double

%f ← float a; → 4 bytes

Range → 3.4e-38 to 3.4e+38

%lf ← double a → 8 bytes

Range 1.7e-308 to 1.7e+308

✓ %Lf
✓ %LR

long double → Compiler dependent

16 bits	32 bit	64 bit
8 bytes	12 bytes	16 bytes

1.23+4

0.000123 → $\frac{1.23}{10000}$ → $\frac{1.23}{10^4}$

$1.23 * 10^{-4}$

1.23e-4

mantissa, Exponent

float a = 0.000123;

printf("%f", a); → 0.000123

printf("%e", a); → 1.23e-4

float b = 0.000000123;

printf("%f", b); → 0.000000

printf("%e", b); → 1.23e-7

%g →

10000000000 X
1000000000

printf("%g", a); → 0.000123

printf("%g", b); → 1.23e-7

③ Character →

%c char → 1 byte →

ASCII-7 → 128

ASCII-8 → 256

↳ 128

-128 to 127

%c unsigned char → 1 byte

range → 0 to 255 → 256

④ string :→

"abcd"

'\0'

eos

printf("%d", sizeof("abcd")); → 4+1 → 5

printf("%d", sizeof(5.4)); → 8

↑ double

printf("%d", sizeof(5.4f)); → 4

↑ float

Operators :→

Operators :→

① Unary	② Binary	③ Ternary.
single operand	Two operands	Three operands

① Arithmetic op :→ [left to Right]

+ , − , * , / , %

low — high

```
    ②  ①
a = 5 + 6 * 2
a = 5 + 12
a = 17
```

```
      ①  ②  ③  ④
a = 5 * 2 / 3 % 7 * 5
    10 / 3 % 7 * 5
       3 % 7 * 5
         3 * 5
          15
```

② Unary operators

+ , − , ++ , −− , sizeof , typecasting

sign op

+5 ✗
−5 ✓

sign change

```
int a = 5;
a = -a;      -(+5)
             -5

a = -a;      -(-5)
             +5
```

a = 5 → −5

++ (increment) (inc. by one)

−− (decrement)

++ (increment)
(inc by one)

```
int a=5;
a++;
printf("%d", a); → 6
```

a | 6

-- (decrement)
(dec by one)

```
int a=5;
a--;
printf("%d", a); → 4
```

a | 4

sizeof → op | keyword

sizeof (data type | variable-name | value)

typecasting :→

```
int a=5;
printf("%f", a); → 0.000000
printf("%f", (float)a); → 5.000000
```

typecasting

Pre (++a)

```
int a=5;
++a;
printf("%d", a); → 6
```

a | 6

```
a=5;
printf("%d", ++a); → 6
```

Post (a++)

```
int a=5;
a++;
printf("%d", a); → 6
```

a | 6

```
a=5;
printf("%d", a++); → 5
```

```
int a=5, b;
b = a++;
printf("%d %d", a, b); →
```

a | 6

b | 5

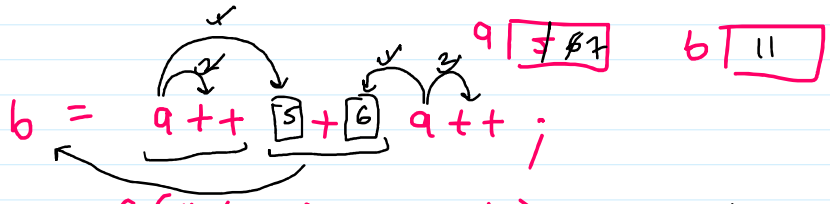
printf("%d %d", a, b); →

6 5

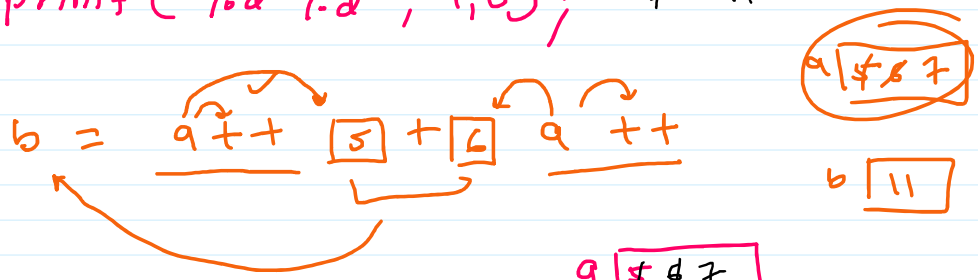
a 5

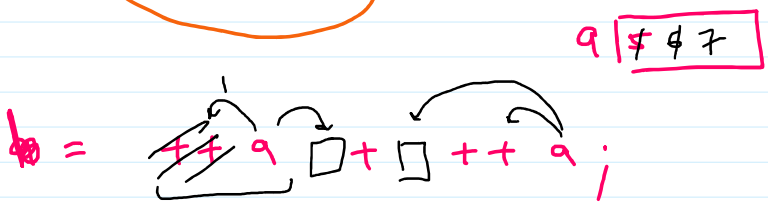
b = ++a;

printf("%d %d", a, b); → 6, 6



printf("%d %d", a, b); 7 11





printf("%d %d", a, b); → 7, 13

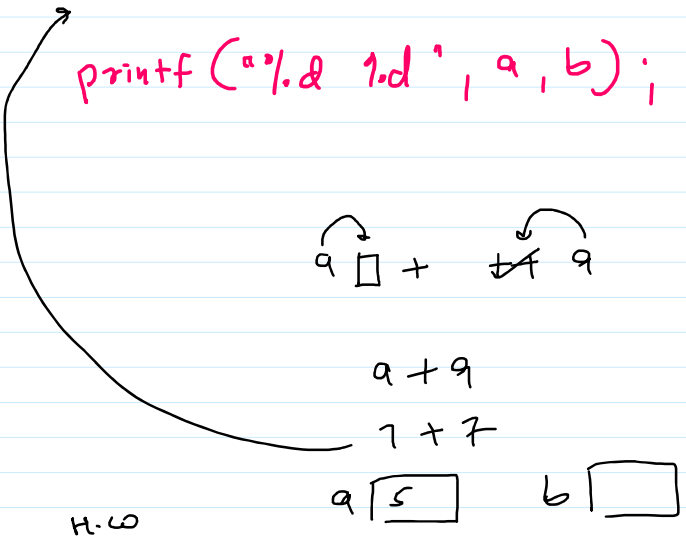
7, 14

a 7

b = ++a + ++a

a + a

7 + 7



H.W

① b = a++ + ++a

② b = ++a + a++

③ b = a++ + a++ + a++

④ b = ++a + ++a + ++a

a | 5 | 6 | 7

a++ | 5 + 6 | a++

a = a+1

④ Relational → boolean result

>, <, >=, <=

true → 1 , false → 0

Non-zero → True

zero → False

if (condition)

printf ("%d", 5>4); → 1

⑤ Equality operators

==, !=

5==4 → 0

5==5 → 1

5!=4 → 1

5!=5 → 0

a b c

5==5==5 → 0

1

X a==b==c

a==b && b==c

T T

⑥ Assignment op :- (Right to left)

=, +=, -=, *=, /=, %= ← short hand op

↑ Assign

int a,b,c,d;

a | 10

b | 10

c | 10

L value = R-value

↑ variable ↑ constant

a = b = c = d = 10;

(4) (3) (2) (1)

d 10

variable
Equation

5 + 7 = 3 ; X

int a = 5;

a = a + 2; → a += 2;

a = a - 2; → a -= 2;

a = a * 2; → a *= 2;

a = a / 2; → a /= 2;