Header files

→ iostream
→ iomanip → setw()
→ math
→ algorithm
→ vector
→ queue

　　．
　　：
　　：

cp

bits/stdc++.h

Namespaces

Java

↓

Internal
Personal
Java

namespace std

namespace A {

⬭

}

namespace std;
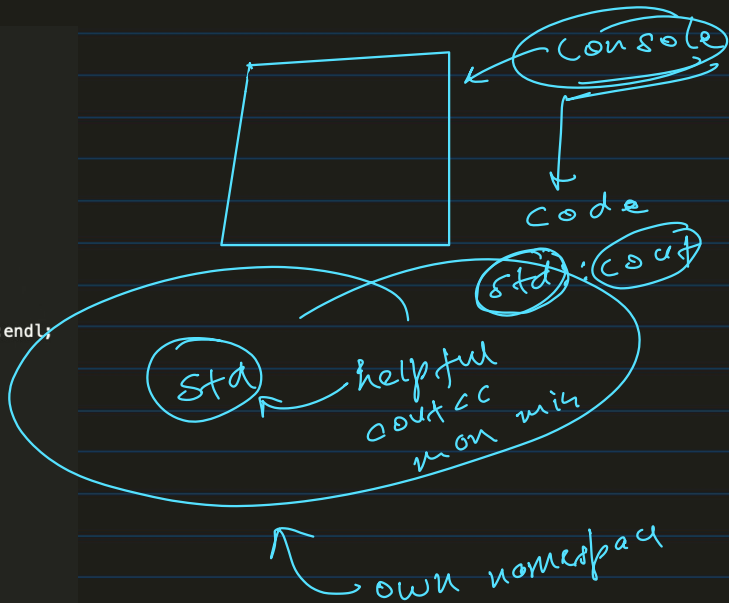
cout << " ",

```cpp
#include<bits/stdc++.h>

void FIO() {
    std::ios_base::sync_with_stdio(0);
    std::cin.tie(0); std::cout.tie(0);
#ifdef ONLINE_JUDGE
    freopen("input.txt", "r", stdin);
    freopen("output.txt", "w", stdout);
#endif
}


namespace alka {

void cout(int a) {
    std::cout << a << " is present" << std::endl;
}

std::string name = "Alka Gupta";
}

int main() {
    FIO();
    using namespace alka;
    cout(1);
    std::cout << "Hi" << std::endl;
    std::cout << name << std::endl;
    return 0;
}
```
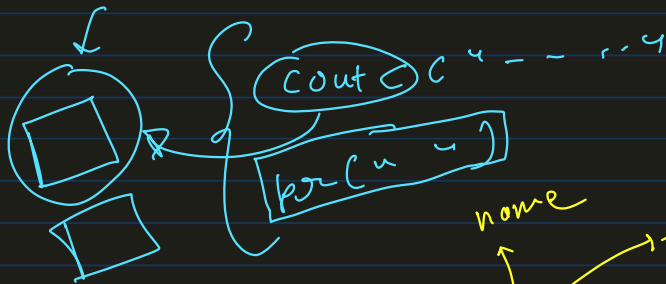
Console

code

std :: cout

std   helpful
       cout <<
       mon mix

own namespace

cout < ...

pr(...)

name

function sign

void print(int a)

return
data type

parameter

Main defination

Compiler's girlfreind

```
Int main ( ) {



    return 0;
}
```

[f1

[f2   segmentation fault

[f3 ─────────────→

int main () {
    ___
    ___
    f2
    ___
    f1
    ___
    ____
    return o;
}

Data Type

→ primary / pre-defined / atomic

    → int
    → char
    → ~~string~~
    → Boolean
    → float
    → double
    → void

$\emptyset$ → (set)

→ User defined

   → class
   → structure
   → Enum
   → Type def

Structure

Student
string name
int roll
float marks

⟶ amalgum of pre-defined data types

Class

public private protected

C++20

```cpp
struct Student {

    string name;
    int roll;
    float marks;

    // Constructor
    Student() {
        cout << "Student is Created!" << endl;
    }

    // destructor
    ~Student() {
        cout << "Student is Destroyed" << endl;
    }
};

class A {
    int a;

};
```
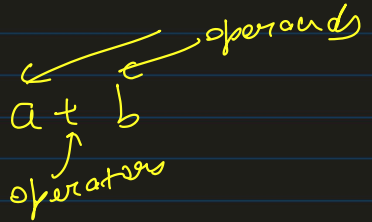
public

private

Operator

Expression
Collection of operators & operands

eg :-   4 × 16 / 8

a + b

operands

operators

Operator	Type
+ +, - -	Unary operator
+, -, *, /, %	Arithmetic operator
<, <=, >, >=, ==, !=	Relational operator
&&, ||, !	Logical operator
&, |, <<, >>, ~, ^	Bitwise operator
=, +=, -=, *=, /=, %=	Assignment operator
?:	Ternary or conditional operator

Operators in C++

Unary operator

Binary operator

Ternary operator

unary → one

→ One operand

$a++$, $a--$, $++a$, $--a$

$!a$

Binary

$a+b$, $a+7$

$a-b$

$a \&\& b$

Ternary

$((a>b)? \ a : \ b)$

Operators in C

Operator	Type
++, -- , !	Unary operator
+, -, *, /, %	Arithmetic operator
<, <=, >, >=, ==, !=	Relational operator
&&, ||, !	Logical operator
&, |, <<, >>, ~, ^	Bitwise operator
=, +=, -=, *=, /=, %=	Assignment operator
?:	Ternary or conditional operator

Unary operator →
Binary operator
Ternary operator →

GeeksforGeeks

Arithematic

$$+ , - , / , * , \%$$

$$x \% o n = [0 \cdots n-1]$$

Relational
↳ compare

$$< , <= , > , >= , == , !=$$

$$a = 4 , \quad b = 6$$

$$\boxed{a <= b} \rightarrow \boxed{4 <= 6} \rightarrow \boxed{1}$$
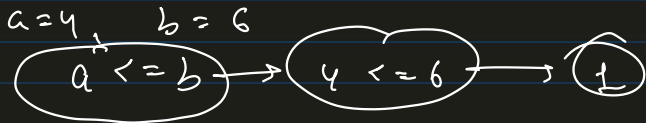
int a = 6   float b = 6.0

a == b

a != b   → 0

#logical

↳ Logic

&& / and

|| / or

! / not

They combine conditions

$a = 6$, $b = 5$, $c = 10$

$a >= b$ && $b <= c$

$6 >= 5$ && $5 <= 10$

1 && 1

1

1 && 0 = 0

!1 = 0   !0 = 1

C++ ⟶ +ve number , -ve number

(true) L

C++ ⟶ 0
   └ false

(4 && 0)   (4 && 5)   while ((-1)) {   ①
    F   T && 1
(4 || 0)   ④
    T
(-0 || +0)   ,-0 ≈ +0   javascript (-0 , +0 , 0)
    └ c++

Bitwise ✶✶

&		bitwise AND
|	"	OR
<<	"	left shift
>>	"	right shift
~	"	Not
^	"	XOR

$a = 6 \qquad b = 4$

$a \ \& \ b \longrightarrow \quad a = 6 = \quad 1 \ 1 \ 0$

$\qquad\qquad\qquad\qquad b = 4 = \ \& 1 \ 0 \ 0$

$\qquad\qquad\qquad\qquad\qquad\qquad 1 \ 0 \ 0$

$a \ \& \ b = 6 \ \& \ 4 = 1 \ 1 \ 0 \ \& \ 1 \ 0 \ 0 = 1 \ 0 \ 0 = 4$

$a \ | \ b \qquad\qquad a = 6 = 1 \ 1 \ 0$

$\qquad\qquad\qquad b = 4 = 1 \ 0 \ 0$

$\qquad\qquad\qquad\qquad\quad 1 \ 1 \ 0 \longrightarrow 6$

$<<\longrightarrow$ <u>left shift</u>

$10<<1 \longrightarrow 1010<<1 \longrightarrow 0101$

① 0 1 1 0 0 ① ← 0

MSB

Lost

LSB
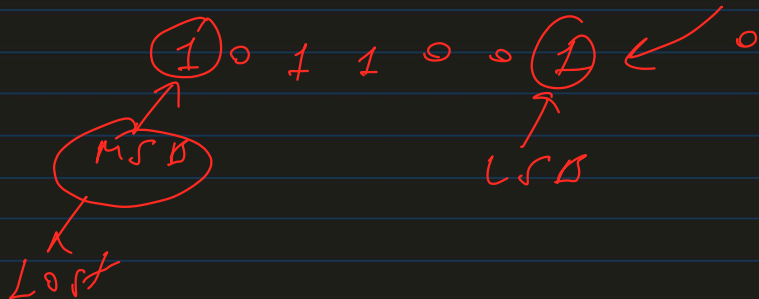
Right shift >>

$$9 >> 2 \longrightarrow 1001 >> 2$$

$$0100$$
$$\hookrightarrow 0010$$

Not

$$\sim 6 = \sim(110) = (001)$$
$$\sim 10 = \sim(1010) = (0101) = (101) = 5$$

XOR (^)

Truth Table

A	B	A^B
0	0	0
0	1	1
1	0	1
1	1	0

4 ^ 6

$$
\begin{array}{r}
100 \\
^{\wedge}\ 110 \\
\hline
010 \\
\hline
\end{array}
$$

$\longrightarrow$ 2

Assignment operator (=)

Language Expression ⟶ Right to Left

operator ⟶ Associativity

$a = 4$

$a += 4 \longrightarrow a = a + 4$

$\underset{4+4}{}$

$a -= 4 = a = a - 4 \ldots$

$a \% = 4$

$a = a \% 4$

$a \&= 4$

$\hookrightarrow a = a \& 4$