5. Bitwise operators (6)

- Bitwise operators perform on the binary representations of integers. **OR** Bitwise operators work on individual bits of integers. These are basically used for testing or shifting bits.

Operator	Description	Example
&	Binary AND Operator copies a bit to the result if it exists in both operands.	(A & B) will give 12, which is 0000 1100
|	Binary OR Operator copies a bit if it exists in either operand.	(A | B) will give 61, which is 0011 1101
^	Binary XOR Operator copies the bit if it is set in one operand but not both.	(A ^ B) will give 49, which is 0011 0001
~	Binary Ones Complement Operator is unary and has the effect of 'flipping' bits.	(~A) will give -61, which is 1100 0011 in 2's complement form. Result in decimal : ~A = -(A+1)
<<	Binary Left Shift Operator. The left operands value is moved left by the number of bits specified by the right operand.	A << 2 will give 240 which is 1111 0000
>>	Binary Right Shift Operator. The left operands value is moved right by the number of bits specified by the right operand.	A >> 2 will give 15 which is 0000 1111

```
1   #include<iostream>
2   #include<conio.h>
3   using namespace std;
4   int main()
5   {
6       int a=5,b=3;
7       cout << (a & b) << endl;  //and
8       cout << (a | b) << endl; // or
9       cout << "xor = " << (a ^ b) << endl; //xor
10      cout << (~a) << endl; // bitwise negative OR Complement operator
11      cout << (~b) << endl; // bitwise negative
12      cout << (a << 1) << endl; // left shift operator
13      cout << (a >> 2) << endl; // right shift operator
14
15      getch();
16  return 0;
17  }
```

```
C:\Users\hira\Desktop\hira.exe
1
7
xor = 6
-6
-4
10
1
```

Not Confuse

Bitwise	Logical
a & b	a && b
a | b	a || b
a ^ b	a != b
~a	!a

6. Assignment Operators (11)

- Assignment operators are used to assign a value (or) an expression
  (or) a value of a variable to another variable.
- Syntax : variable name=expression (or) value (or) variable
  - ✓ Ex : x=10;
    y=a+b;
    z=p
- i += 2 (The operator += is called an assignment operator.)

Operator	Example	Meaning
+ =	x + = y	x=x+y
- =	x - = y	x=x-y
* =	x * = y	x=x*y
/ =	x / = y	x=x/y
% =	x % = y	X=x%y

=	Assignment operator
+= -=	Addition/subtraction assignment
*= /=	Multiplication/division assignment
%= &=	Modulus and bitwise assignment
^= |=	Bitwise exclusive/inclusive OR assignment
<<= >>=	

If x *= y + 1 means → x = x * (y + 1) rather than x = x * y + 1

```
1  #include<iostream>
2  #include<conio.h>
3  using namespace std;
4  int main()
5   {
6         int a=10;
7         int b=a;
8           cout << "a = " << a <<endl;
9             cout << "b = " << b <<endl;
10         b *= a+b;
11        cout << "b = " << b;
12      getch();
13 return 0;
14 }
15
```

```
C:\Users\hira\Desktop\hira.exe
a = 10
b = 10
b = 200
```

7. Ternary or Conditional Operator (1)

- In c++, the ternary operator is a shorthand way to write simple if-else statement, It's the only operator in c++ that takes three operands.
- It is also called Conditional Operator.
- Syntax : Condition ? expression_if_true : expression_if_false;
- Example: int a=10,b=20;
    - Int max = (a>b) ? a : b ;

Explain this line

```
int max = (a>b) ? a : b ;
```

```
int max;
if(a>b)
{
  max =a;
}
else
{
  max =b;
}
```

```
#include<iostream>
#include<conio.h>
using namespace std;
int main()
{
      int a=10, b=20;
      int max_value = (a>b) ? a : b ;
      cout << "Max Value = " << max_value;
    getch();
return 0;
}
```

```
C:\Users\hira\Desktop\hira.exe
Max Value = 20
```

```
#include<iostream>
#include<conio.h>
using namespace std;
int main()
{
      int a=10, b=20;
      int max_value;
      if (a>b)
      {
          cout << "Max value = " << a;
      }
      else
      {
          cout << "Max value = " << b;
      }
    getch();
return 0;
}
```

```
C:\Users\hira\Desktop\hira.exe
Max value = 20
```

8. Special / Miscellaneous Operators

- A part from these operators, there are a few operators that do not fit in any of the above categories. These are:-

1. sizeof Operator

- Return the size (in byte) of a data type or variable.
- Syntax : sizeof(data_type / Variable_name)
- Example:-

```
#include<iostream>
#include<conio.h>
using namespace std;
int main()
 {
       int a=10;
       cout << "Size of int = " << sizeof(int) << endl;
       cout << "size of a = " << sizeof(a);
       getch();
return 0;
}
```

```
C:\Users\hira\Desktop\hira.exe
Size of int = 4
size of a = 4
```

2. Comma Operator(,)

- Evaluates multiple expressions and returns the value of the last expression.
- Example : int a=5,b=10;

  Int c = (a++,b++,a+b);

  Cout << c; // c = 17

3. Address-of-operator (&)

- Return the memory address of a variable.
- Example :

  Int a=10;

  Cout << &a; //ask address of operator

```
#include<iostream>
#include<conio.h>
using namespace std;
int main()
 {
      int a=10;
      cout << "a = " << a << endl;
      cout << "address of a = " << &a;
       getch();
return 0;
}
```

```
C:\Users\hira\Desktop\hira.exe
a = 10
address of a = 0xaed03ff9cc
```

4. Derefernce Operator (*)

- Accesses the value at a memory address (opposite of &)

```cpp
#include<iostream>
#include<conio.h>
using namespace std;
int a=100;
int main()
 {
      int a=10;
      int *ptr = &a;
      cout << "ptr = " << *ptr;
       getch();
return 0;
}
```

```
C:\Users\hira\Desktop\hira.exe
ptr = 10
```

5. Scope Resolution Operator (::)

- Accesses globle variable, namespace members, or class static members.

```cpp
#include<iostream>
#include<conio.h>
using namespace std;
int a=100;
int main()
 {
      int a=10;
      cout << " Local value of a = " << a << endl;
      cout << " Globle value of a = " << ::a ;
       getch();
return 0;
}
```

```
C:\Users\hira\Desktop\hira.exe
Local value of a = 10
Globle value of a = 100
```

6. Dot operator (.)

- Accesses members of an object/struct.

```cpp
#include<iostream>
#include<conio.h>
using namespace std;
struct student
{
    string name;
    int age;
};
int main()
 {
      student sl;
      sl.name ="Hira"; // dot operator
      sl.age = 90;
      cout << "Name = " << sl.name << " & Age = " << sl.age;
       getch();
return 0;
}
```

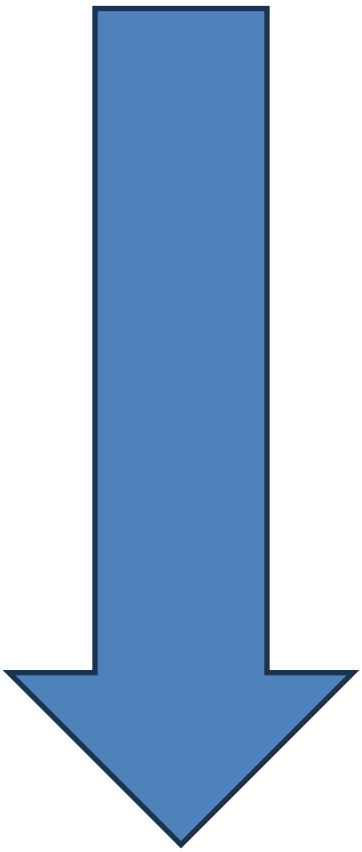
```
C:\Users\hira\Desktop\hira.exe
Name = Hira & Age = 90
```

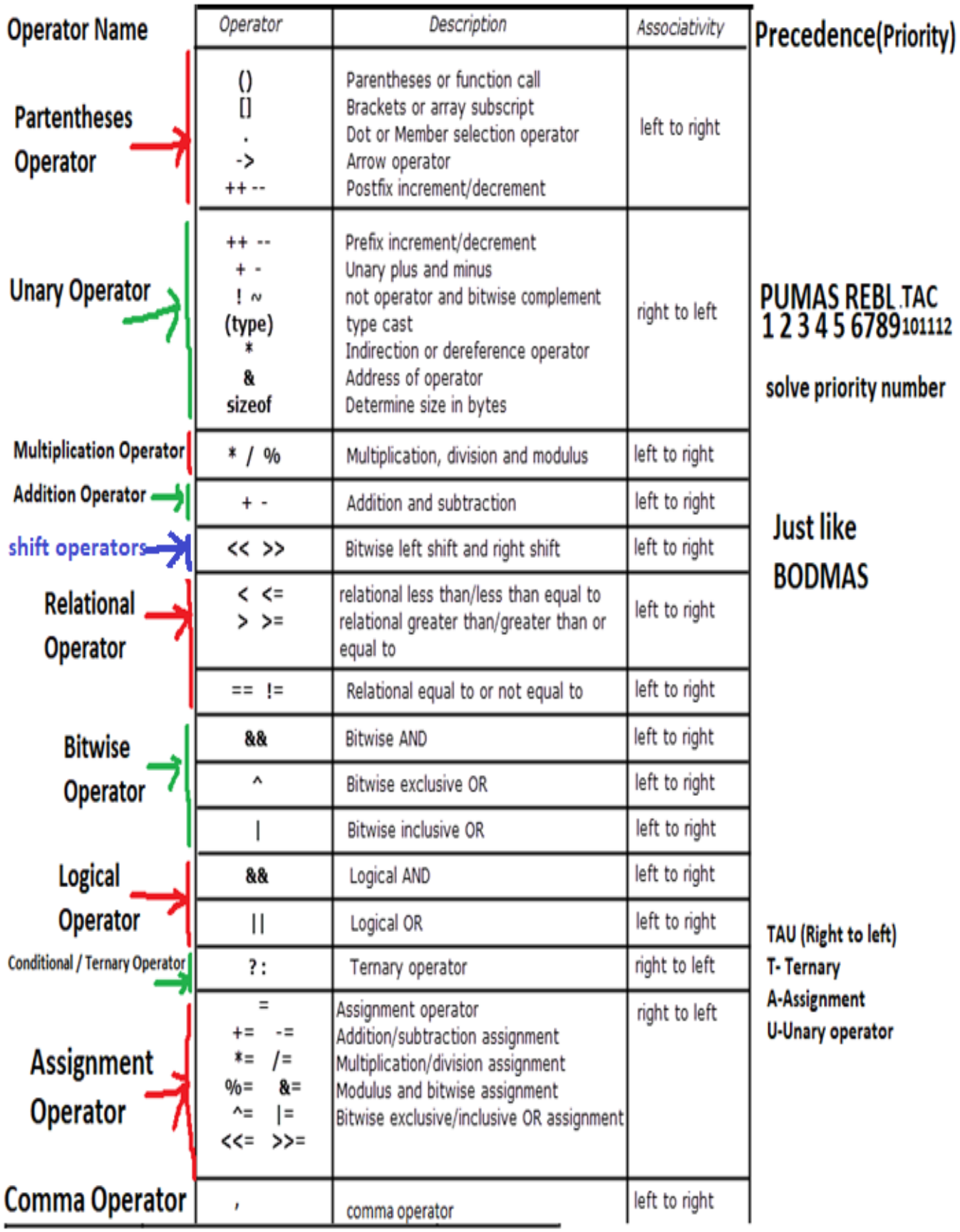
7. Arrow Operator (->)

- Accesses members through a pointer.

```
#include<iostream>
#include<conio.h>
using namespace std;
struct student
{
    string name;
    int age;
};
int main()
 {
     student sl;
     student *ptr =&sl;
     ptr -> name ="Hira";
     ptr -> age =100;
     cout << "Name = " << sl.name << " & Age = " << sl.age;
      getch();
return 0;
}
```

```
C:\Users\hira\Desktop\hira.exe
Name = Hira & Age = 100
```





==========HIRA KUMAR==========

HIRA KUMAR