CSE101

OPERATORS

- In this lecture we will study
  - Operators
  - Types of Operators

Operators

- Operator is the symbol which performs some operations on the operands.

5+5=10

+ and = are the operator and
5 and 10 are operands

Types of Operators

- **Types of operators are:**
  1. Arithmetic operator
     - » Unary operator
     - » Binary operator
  2. Relational operator
  3. Logical operator
  4. Assignment operator
  5. Conditional operator
  6. Bitwise operator
  7. Special operator

Description of Operators

➢ Arithmetic Operators

These are binary operators i.e. expression requires two operands

Operator	Description	Example (a=4 and b=2)
+	Addition of two operands	a + b = 6
-	Subtraction of two operands	a – b = 2
*	Multiplication of two operands	a * b = 8
/	Division of two operands	a / b = 2
%	Modulus gives the remainder after division of two operands	a % b = 0

Arithmetic Operators

If the radius of car wheel is 15inch then what will the diameter and calculate distance traveled after one rotation of that wheel?

Arithmetic Operators

Sol:

r = 15

diameter = r + r = 2 * r = 2 * 15 = 30

dist_travelled = pi * d

dist_travelled = pi * diameter

= 3.14 * 30 = 94.2

Arithmetic Operators

To get the remainder of the integer v

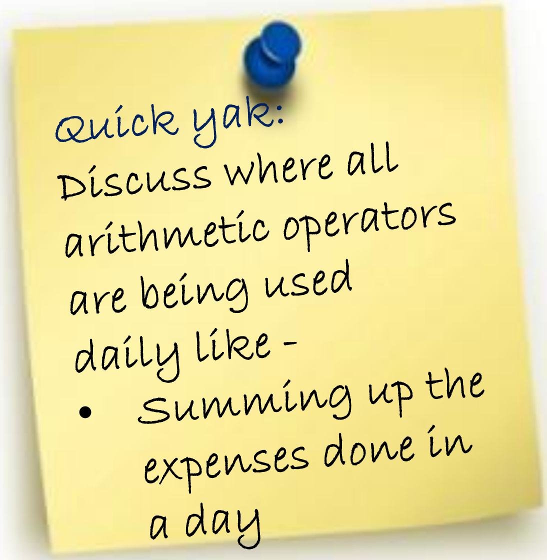
Eg:

14 mod 3 = 2

17 mod 2 = 1

190 mod 3 = 1

```
3)14(4
   12
    2
```



Quick yak:
Discuss where all arithmetic operators are being used daily like -

- Summing up the expenses done in a day

Q:Suppose we have to distribute 10 c
equally then after equal distributi be
left?

Sol: 10 mod 3 = 1

So 1 chocolate will be left as all 3 students will have 3 chocolates each.

➢Unary Operator

These operator requires only one operand.

Operator	Description	Example(count=1)
+	unary plus is used to show positive value	+count; value is 1
-	unary minus negates the value of operand	-count; value is -1
++	Increment operator is used to increase the operand value by 1	++count; value is 2 count++; value is 2
--	Decrement operator is used to decrease the operand value by 1	--count; value is 1 count--; value is 1

++count increments count by 1 and then uses its value as the value of the expression. This is known a **prefix operator**.
count++ uses count as the value of the expression and then increments count by 1. This is known as **postfix operator**.

Unary Operators

Q: In an exam there was 10 question each carry 1 mark for right answer and 0.50 marks were deducted for wrong answer. A student attempted 6 questions and out of that 5 questions were wrong. So what is the score of the student out of 10?

Sol: No. of questions attempted = 6

Marks deducted = 5 * 0.50 = 2.5

Marks for right answer = 1

Total marks = 1 – 2.5 = -1.5

Unary Minus indicates that value is negative.

Unary Operators

Q: Suppose 3 friends went for shopping. All of them took a toothbrush for themselves.

++counter

So the counter(no. of toothbrush) = 3 $\rightarrow$ ++counter

At the time of billing cashier told them that there is one toothbrush free with the purchase of 3 toothbrush.

But before the counter = 4 the friends have paid only for 3 toothbrush.

counter = 4 $\rightarrow$ counter++

i.e before incrementing the counter they have used the value of counter to pay bill.

➢Relational Operator

It compares two operands depending upon the their relation. Expression generates zero(false) or nonzero(true) value.

Operator	Description	Example (a=10 and b=20)
<	less than, checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.	(a < b) value is 1(true)
<=	less than or equal to, checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.	(a <= b) value is 1 (true).
>	greater than, checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true.	(a > b) value is 1 (true).
>=	greater than or equal to, checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.	(a >= b) value is 1 (true).
==	equality ,checks if the value of two operands is equal or not, if yes then condition becomes true.	(a == b) value is 1 (true).
!=	inequality, checks if the value of two operands is equal or not, if values are not equal then condition becomes true.	(a != b) value is 1 (true).

Relational Operator

Q: Age of Sam is 20 and age of Tom is 19.

Verify the relationship between their age.

Sol: age of Sam : sAge = 20 age of Tom : tAge = 19

sAge < tAge → 0 (false)

sAge > tAge → 1 (true)

So, Sam is elder than Tom.

sAge == tAge → 0 (false)

Quick yak:
Where relational operators being used:

- Comparing heights
- Passing an exam
- Grades obtained

➢Logical Operator

It checks the logical relationship between two expressions and the result is zero(false) or nonzero(true).

Operator	Description	Example
&&	Logical AND operator. If both the operands are true then condition becomes true.	(5>3 && 5<10) value is 1 (true).
| |	Logical OR Operator. If any of the two operands is true then condition becomes true.	(5>3 || 5<2) value is 1 (true).
!	Logical NOT Operator. Use to reverses the logical state of its operand. If a condition is true then Logical NOT operator will make false.	!(8==8) value is 0 (false).

Logical Operator

Grade system :

If (Marks >=90 || attendance == 100)

students performance is excellent.

If (Marks <= 40 && attendance < 75)

student is detained.

➢ Bitwise Operator

A bitwise operator works on each bit of data.

Logical Table				
a	b	a & b	a | b	a ^ b
0	0	0	0	0
0	1	0	1	1
1	1	1	1	0
1	0	0	1	1

Operator	Description	Example(a=1 and b=0)
&	bitwise AND	a & b = 0
|	bitwise OR	a| b = 1
^	bitwise XOR	a ^ b = 1
~	bitwise one's complement	~a = 0, ~b=1
<<	bitwise left shift, indicates the bits are to be shifted to the left.	1101 << 1 = 1011
>>	bitwise right shift, indicates the bits are to be shifted to the right.	1101 >> 1 = 1110

➢Assignment Operator

They are used to assign the result of an expression on right side to a variable on left side.

Operator	Description	Example(a=4 and b=2)
+=	a=a+b	a+=b; a=a+b = 6
-=	a=a-b	a-=b; a=a-b = 2
*=	a=a*b	a*=b; a=a*b = 8
/=	a=a/b	a/=b; a=a/b = 2
%=	a=a%b	a%=b; a=a%b = 0
>>=	a=a>>b	a=00000100 >> 2 = 00010000
<<=	a=a<<b	A=00000100 << 2 = 00000001
&=	a=a&b	(a=0100, b=0010) a&=b; a=a&b = 0000
|=	a=a|b	(a=0100, b=0010) a|=b; a=a|b =0110
^=	a=a^b	(a=0100, b=0010) a^=b; a=a^b = 0110

Assignment Operator

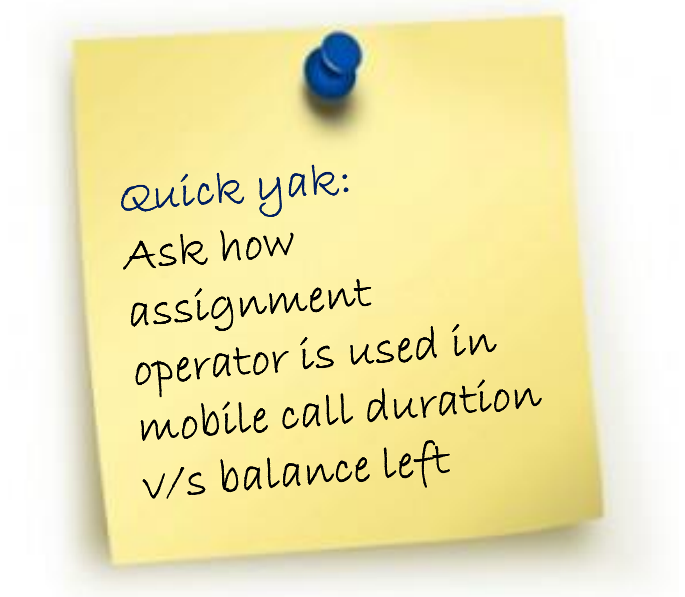
- To increase the cost of item soa

      Cost_soap = Cost_soap + 5

      or Cost_soap += 50;

- To double the quantity of wate

      Water_inBowl *= 2;



Quick yak:
Ask how assignment operator is used in mobile call duration v/s balance left

✓ Therefore assignment operator are used to store the changed value of the variable in the same variable.

➢Conditional Operator

Conditional operator contains condition followed by two statements. If the condition is true the first statement is executed otherwise the second statement.

It is also called as **ternary operator** because it requires three operands.

Operator	**Description**	**Example**
?:	conditional expression, Condition? Expression1: Expression2	(a>b)? “a is greater”: “b is greater”

Conditional Operator

- Eligibility to cast vote

    (age>=18)? “can cast vote”: “cannot cast vote”;

- In C

    (age>=18)? printf(“can cast vote”) : printf(“cannot cast vote”);

➢Some Special Operators

Operator	Description	Example
,	comma operator, can be used to link the related expressions together	int a, b, x;
sizeof ()	sizeof operator to find the size of an object.	int a; sizeof(a)=2

type	Cast operator, to change the data type of the variable	float x= 12.5;
		int a;
		a = (int) x; value of a is 12.

- Comma operator can be used like:

```
for(i=0 , j=1  ;  i>10  ;  i++ , j++)
```

- To know space occupied by variable in computer memory we use ***sizeof()* operator.**

```
char choice;
int char_sz = sizeof(choice); // 1 because char is 1byte
```

- If we are adding float number and integer number and we require output in float then integer number is converted to float using *type cast* operator.

```
int num1;
float num2, sum;
sum= (float) num1 + num2;
```

Precedence of Operators

- The precedence of operators *determine a rank for the operators.*
- The higher an operator's precedence or priority, the higher binding it has on the operands.

Example: So how the expression a * b + c will be interpreted?

(a * b) + c or a * (b + c),

here the first interpretation is the one that is used because the multiplication operator has higher precedence than addition.

Associativity of Operators

- Associativity tell us the order in which several operators with equal precedence are computed or processed in two directions, either from left to right or vice-versa.

Example: In the expression → **a * b / c,**
since <u>multiplication and division have the same precedence</u> we must use the associativity to determine the grouping.

These operators are left associative which means they are grouped left to right as if the expression was

(a * b) / c.

Operator	Associativity	Type
() [] . -> ++(postfix) - - (postfix)	left to right	Highest
+ - ++ -- ! & * ~ sizeof (type)	right to left	Unary
* / %	left to right	multiplicative
+ -	left to right	additive
<< >>	left to right	shifting
< <= > >=	left to right	relational
== !=	left to right	equality
&	left to right	bitwise AND
^	left to right	bitwise OR
|	left to right	bitwise OR
&&	left to right	logical AND
||	left to right	logical OR
?:	right to left	conditional
= += -= *= /= &= |= ^= <<= >>= %=	right to left	assignment
,	left to right	comma

Standard Library

Real life example

Header for tires

Header for oil

Header for speakers

Ferrari car as output

C code example

```
//Sample program
#include<stdio.h>  //header file for printf()
#include<conio.h> //header filr for getch()
int main()
{
   //stdio.h is providing printf() function
    printf("Car is under process");
    //conio.h is providing getch() function
    getch();
}
```

Output:

```
Car is under process
```

C Standard Library

- C programs consist of modules or pieces called **functions.**
- You can program all the functions you need to form a C program,
- C has a rich collection of existing functions called the **C Standard Library.**

C Standard Library

- Function prototype and data definitions of these functions are written in their respective header file.
- For example: If you want to use **printf()** function, the header file **<stdio.h>** should be included.
- The C Standard Library is a set of C built-in functions, constants and header files like <stdio.h>, <stdlib.h>, <math.h>, etc.

C standard Library Functions

stdio.h:	**I/O functions**
getchar()	returns the next character typed on the keyboard.
putchar()	outputs a single character to the screen.
printf()	use to output data to user.
scanf()	use to input data from user.
string.h:	**String functions**
strcat()	concatenates a copy of string2 to string1
strcmp()	compares two strings
strcpy()	copy's contents of string2 to string1
math.h:	**Mathematics functions**
cos()	returns cosine of argument
exp()	returns natural logarithmic
fabs()	returns absolute value of number
sqrt()	returns square root of number
pow()	Calculates the number raise to pow

C standard Library Functions

ctype.h:	**Character functions**
isdigit()	returns non-0 if argument is digit 0 to 9
isalpha()	returns non-0 if argument is a letter of the alphabet
isalnum()	returns non-0 if argument is a letter or digit
islower()	returns non-0 if argument is lowercase letter
isupper()	returns non-0 if argument is uppercase letter
stdlib.h:	**Miscellaneous functions**
malloc()	provides dynamic memory allocation.
rand()	returns a random value.
srand()	used to set the starting point for rand()

Next Class: Control Structures

cse101@lpu.co.in