1. 1. BITWISE OPERATOR-

* Bitwise AND (&)- It takes two operands and does AND on every bit of two numbers. The result of AND is 1 only if boths bits are 1.
* Bitwise OR (|)- It takes two numbers as operandsand does OR on every bit of two numbers. The result of OR is 1 if any of the two bits are 1.
* Bitwise XOR(^)-It takes two numbers as operandsand does OR on every bit of two numbers. The result of OR is 1 if two bits are different.
* Bitwise NOT(~)- It takes one number and inverts all bits of it.
* Bitwise Right Shift(>>)- It takes two numbers, right shift of the bits of the first operand ,the second operand decides the number to be shifted.
* Bitwise Left Shift (<<)- It takes two numbers, left shift of the bits of the first operand ,the second operand decides the number to be shifted.

1. TERNARY OPERATOR-

* Is used to execute code based on the result of a binary condition. The input is in binary , and returns a value , like a function. It also known as conditional operator , which takes less space and helps to write the if-else statements in the shortest way.

#include<stdio.h>

int main()

{

int num1, num2;

char ch;

float result:

printf(“ Write the first no.:”);

scanf(“%d”,&num1);

printf(“ Write the second no.:”);

scanf(“%d”,&num2);

printf(“choose the operation(+,-,\*,/,%):”);

scanf(“%c”,&ch);

Switch(ch)

{

case1 ‘+’:

result=num1+num2;

break;

case2 ‘-’:

result=num1+num2;

break;

case3 ‘\*’:

result=num1\*num2;

break;

case4 ‘/’:

result=(float)num1/(float)num2;

break;

case5 ‘%’:

result=num1%num2;

break;

default:

printf(“The operation is invalid.\n”);

}

printf(“Result is: %d %c %d=%f\n”,num1,num2,ch,result”);

Return 0;

}