Add two numbers without using arithmetic operators

Write a function Add() that returns sum of two integers. The function should not use any of the arithmetic operators (+, ++, –, -, .. etc).

Sum of two bits can be obtained by performing XOR (^) of the two bits. Carry bit can be obtained by performing AND (&) of two bits.
Above is simple Half Adder logic that can be used to add 2 single bits. We can extend this logic for integers. If x and y don't have set bits at same position(s), then bitwise XOR (^) of x and y gives the sum of x and y. To incorporate common set bits also, bitwise AND (&) is used. Bitwise AND of x and y gives all carry bits. We calculate (x & y) << 1 and add it to x ^ y to get the required result.

```
#include<stdio.h>

int Add(int x, int y)
{
   // Iterate till there is no carry
   while (y != 0)
   {
      // carry now contains common set bits of x and y
      int carry = x & y;

      // Sum of bits of x and y where at least one of the bits is not set
      x = x ^ y;

      // Carry is shifted by one so that adding it to x gives the required sum
      y = carry << 1;
   }
   return x;
}

int main()
{
   printf("%d", Add(15, 32));
   return 0;
}
```

Following is recursive implementation for the same approach.

```
int Add(int x, int y)
{
   if (y == 0)
      return x;
   else
      return Add( x ^ y, (x & y) << 1);
}
```

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

@geeksforgeeks, **Some rights reserved** **Contact Us!**

Powered by **WordPress** & **MooTools**, customized by geeksforgeeks team