Kate Ross

Mr. Bailey

Computational Mathematics

3 March 2016

The Function of the Summation of Two Integers Without Using Arithmetic Operators

The first thing that comes to mind is the processing of bits. Why? We have no choice – we can not use the "+" operator. So let's sum the numbers as it is done by computers.

Now we need to figure out how the summation works. The additional tasks allow us to develop new skills, learn something interesting, and create new templates.

So let's look at the additional task. We will use the decimal number system.

To sum 759 and 674, we fold the digit [0] of both numbers, transfer 1, and then turn to the digit [1], transfer 1, etc. Similarly, we can work with the bits: sum all the digits and make transfers if they are necessary.

Is it possible to simplify the algorithm? Yes! Let's say we want to divide the "sum" and the "transfer." We have to do the following:

1. Perform the 759 + 674 operation, forgetting about the transfer. The result will be 323.
2. Perform the 759 + 674 operation, but only make transfers (without the summation of digits). The result will be 1110.
3. Now we need to sum the results of the first two operations (using the same mechanism described in steps 1 and 2): 1110 + 323 = 1433.

Now let's go back to the binary system:

1. If we sum a pair of binary numbers, excluding the transfer of the sign, the i-th summed bit can be zero only if the i-th bits of the a and b numbers coincide (both have a value of 0 or 1). This is the classic operation XOR.
2. If we sum a pair of numbers, performing only the transfer, then the i-th bit of the sum is set to 1 only if the (i-1)-th bits of both numbers (a and b) have a value of 1. This is the AND operation with an offset.
3. We need to repeat these steps until there are no more transfers.

The following code implements this algorithm:

```
public static int summarize(int a, int b)    {

    if (b == 0) return a;

    int sum = a ^ b;            // sum without the transfer

    int contain = (a & b) << 1;    // transfer without the sum

    return summarize(sum, contain);        // recursion

}
```

The problems related to the implementation of basic operations (sum, subtraction) are quite popular. To solve these problems, we need to deal with the way operations are usually implemented, and then find a way that allows us to write code with the restrictions.

Thanks for your attention!