Addition and subtraction in vedic maths

Vedic maths Addition

Usually if we want to add to numbers say 52 and 66 we would add the unit digits. and if there is any remainder we will bring it to the tens digit and atlast we will add the tens digit.

Now we will do it in the different way.

Let us take the same number 52 and 66.

The first step is to add the first number and the unit digit of the second number

ie. 52 + 6 = 58

The second step is to jump tens digit time.

our tens digit is 6.

so the answer is 118

Vedic Maths Subtraction

This Vedic Maths Subtraction method found as sutra in ancient vedas, is given below is very useful for such subtractions.

For example 1000 – 357 = ?

We simply take each figure in 357 from 9 and the last figure from 10.

Step 1. 9-3 = 6

Step 2. 9-5 = 4

Step 3. 10-7 = 3

So the answer is 1000 – 357 = 643

PSCNotes free sample: first 5 of 446 pages. Full PDF at pscnotes.com

This always works for subtractions from numbers consisting of a 1 followed by

noughts: 100; 1000; 10,000 etc.

Similarly 10,000 – 1049 = 8951 (subtraction from 10000)

9-1 = 8

9-0 = 9

9-4 = 5

10-9 = 1

So answer is 8951

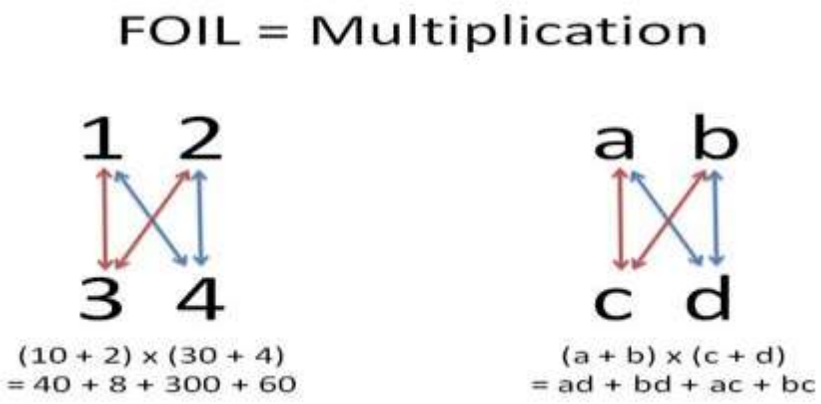
For 1000 – 83, in which we have more zeros than figures in the numbers being subtracted, we simply suppose 83 is 083.

So 1000 – 83 becomes 1000 – 083 = 917

Vedic maths multiplication

The Vedic method (the general method, at least) is based on the Urdhva-Tiryagbhyam sutra. A very terse sutra, it simply translates in English to say "vertically and crosswise". The sutra is rather vague, so the technique, as well as an algebraic analysis of the technique, is presented in the following steps:

An Algebraic Perspective



PSCNotes free sample: first 5 of 446 pages. Full PDF at pscnotes.com

All numbers n the base 10 number system (and number systems of any other base, for that matter) consist of a number of digits. Each digit represents a multiple times a power of 10 (or whatever the number system's base is). So, for example, given a number like 52, we could rewrite it as 5*10+2.

Algebraically speaking, we can express any 3-digit number as: ax+b (where a, and b are integers).

So, suppose we wanted to multiply 2 2-digit numbers. We can express them in polynomial form. Then, by foiling:

$$(ax+b)(cx+d) = acx^2+(ad+bc)x+bd$$

Algebraic Multiplication for Higher Numbers of Digits

So, for brief review, 2-digit by 2-digit algebraic multiplication goes as follows (x in all of the following examples is the base of the number system being used, which is usually 10):

$(ax+b)(cx+d) = acx^2+(ad+bc)x+bd$

Expanding to 3-digit by 3-digit algebraic multiplication:

$(ax^2+bx+c)(dx^2+ex+f) = adx^4+(ae+bd)x^3+(af+be+cd)x^2+(bf+ec)x+cf$

Now, 4-digitby 4-digit algebraic multiplication $(ax^3+bx^2+cx+d)(ex^3+fx^2+gx+h)=aex^6+(af+be)x^5+(ag+bf+ce)x^4+(ah+bg+cf+de)x^3+(bh+cg+df)x^2+(ch+dg)x+dh$

Algorithm method - meaning and quality, indefinate, definite

In mathematics, computing, linguistics and related subjects, an algorithm is a sequence of finite instructions, often used for calculation and data processing. It is formally a type

PSCNotes free sample: first 5 of 446 pages. Full PDF at pscnotes.com

of effective method in which a list of well-defined instructions for completing a task will, when given an initial state, proceed through a well-defined series of successive states, eventually terminating in an end-state. The transition from one state to the next is not necessarily deterministic; some algorithms, known as probabilistic algorithms, incorporate randomness. A step-by-step method of solving a problem or making decisions, as in making a diagnosis. An established mechanical procedure for solving certain mathematical problems.

Properties of algorithm

Input

- The input is the data to be transformed during the computation to produce the output.
- Input precision requires that you know what kind of data, how much and what form the data should be.

Output

- The output is the data resulting from the computation.
- Output precision also requires that you know what kind of data, how much and what form the output should be (or even if there will **be** any output at all!)

Finiteness

Algorithms must terminate after a finite number of steps.

Definiteness

Each instruction of the algorithm should be clear and unambiguous.

Effectiveness

PSCNotes free sample: first 5 of 446 pages. Full PDF at pscnotes.com

Every instruction must be basic enough to be carried out theoretically or by using paper and pencil.

Example of an algorithm: Design an algorithm to add two numbers and display the result.

```
Step 1 – START
Step 2 – declare three integers a, b & c
Step 3 – define values of a & b
Step 4 – add values of a & b
Step 5 – store output of step 4 to c
Step 6 – print c
Step 7 – STOP
```

Algorithms tell the programmers how to code the program. Alternatively, the algorithm can be written as :

```
Step 1 – START ADD
Step 2 – get values of a & b
Step 3 – c ← a + b
Step 4 – display c
Step 5 – STOP
```

Algorithm Analysis Efficiency of an algorithm can be analyzed at two different stages, before implementation and after implementation. They are the following:

A Priori Analysis

This is a theoretical analysis of an algorithm. Efficiency of an algorithm is measured by assuming that all other factors, for example, processor speed, are constant and have no effect on the implementation.

A Posterior Analysis

This is an empirical analysis of an algorithm. The selected algorithm is implemented using programming language. This is then executed on target computer machine. In this analysis, actual statistics like running time and space required, are collected.

PSCNotes free sample: first 5 of 446 pages. Full PDF at pscnotes.com