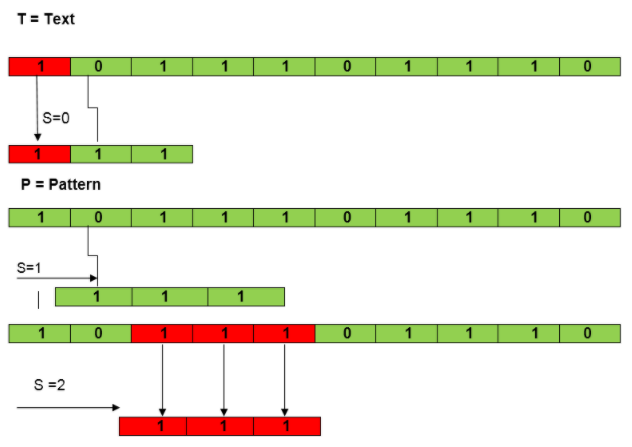
Naïve String-Matching Algorithm

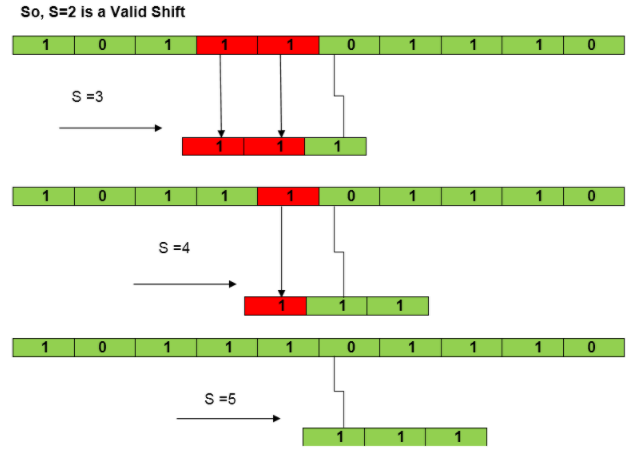
* Naïve pattern searching is the simplest method among other pattern searching algorithms. It checks for all character of the main string to the pattern.
* This algorithm is helpful for smaller texts.
* It does not need any pre-processing phases.
* The naïve approach tests all the possible placement of Pattern P [1.......m] relative to text T [1......n]. We try shift s = 0, 1.......n-m, successively and for each shift s. Compare T [s+1.......s+m] to P [1......m].

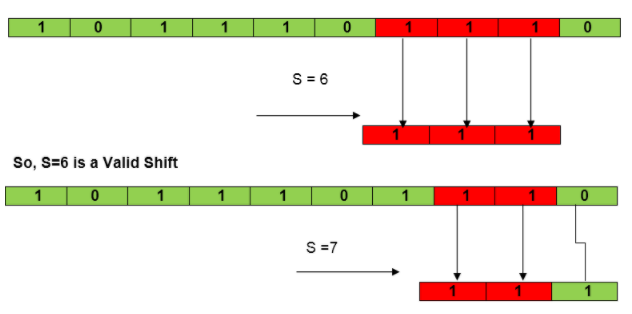
Example:

Suppose T = 1011101110

P = 111







Algorithm:

**NAIVE-STRING-MATCHER (T, P)**

1. n ← length [T]

2. m ← length [P]

3. for s ← 0 to n -m

4. do if P [1.....m] = T [s + 1....s + m]

5. then print "Pattern occurs with shift" s

Rabin-Karp Algorithm

The Rabin-Karp string matching algorithm calculates a hash value for the pattern, as well as for each M-character subsequence of text to be compared. If the hash values are unequal, the algorithm will determine the hash value for next M-character sequence. If the hash values are equal, the algorithm will analyse the pattern and the M-character sequence. In this way, there is only one comparison per text subsequence, and character matching is only required when the hash values match.

Example:

T = 31415926535.......

P = 26

Here T. Length =11 so Q = 11

And P mod Q = 26 mod 11 = 4

Now find the exact match of P mod Q...

