Determine the value of *x*, a two-digit number such that the expression 10x – x when evaluated gives a sum whose digits can be added to yield 300.

Determining the sum of all digits in a decimal number can be done using the modulo operator ‘%’ and the division operator ‘/’ successively. Because we assume the given number *x* is a signed integer, division will always truncate any floating points. For example, 11/10 will yield 1, assuming 11 and 10 are integers. This algorithm completes in linear time and has an O(*n*) time complexity, where n is the number of digits contained in any number *x*. A simple implementation is shown below.

|  |
| --- |
| x = 245  sum = 0  while x / 10 ≥ 1  sum = sum + x % 10  x = x / 10 |

This problem can be efficiently solved through the use of a binary search, a method of determining a desired output by strategically eliminating half of all values in each search iteration. This algorithm has O(log *n*), time complexity. An algorithm implementing both of these sub algorithms completes in logarithmic time. We can assume that x can be given by {x | 0 ≤ x ≤ 99}, as x has two digits. A completed algorithm is shown below.

|  |
| --- |
| x = 50  found = false;  while found = false  y = 10x – x    sum = 0;  while y / 10 >= 1  sum = sum + x % 10  y = y / 10  if sum = 300  found = true  else if sum > 300  x = x + x / 2  else  x = x – x / 2 |

The output of this algorithm yields 34.