Алгоритм Левенштейна

```php
function distance($source, $dest) {
    if ($source == $dest) {
        return 0;
    }

    list($slen, $dlen) = [strlen($source), strlen($dest)];

    if ($slen == 0 || $dlen == 0) {
        return $dlen ? $dlen : $slen;
    }

    $dist = range(0, $dlen);
//Array ( [0] => 0 [1] => 1 [2] => 2 [3] => 3 [4] => 4 [5] => 5 [6]
=> 6 [7] => 7 [8] => 8 [9] => 9 [10] => 10 [11] => 11 [12] => 12 )

    for ($i = 0; $i < $slen; $i++) {
        $ dist = [$i + 1];
        for ($j = 0; $j < $dlen; $j++) {
            $cost = ($source[$i] == $dest[$j]) ? 0 : 1;
            $ dist[$j + 1] = min(
                $dist[$j + 1] + 1,    // deletion
                $ dist[$j] + 1,       // insertion
                $dist[$j] + $cost     // substitution
            );
        }
        $dist = $_dist;
    }

    return $dist[$dlen];
}

$source = 'PHP';
$dest = 'new PHP test';

echo distance($source, $dest) . "\n";
  echo levenshtein($source, $dest); // built-in PHP function
```

S1,S2 - строки длиной M,N , d(S1,S2) - расстояние Левенштейна,
тогда рекуррентное соотношение

$$\mathrm{d}(S_1, S_2) = D(M, N)\text{, где}$$

$$D(i,j) = \begin{cases} 0, & i = 0,\ j = 0 \\ i, & j = 0,\ i > 0 \\ j, & i = 0,\ j > 0 \\ \min\{ & \\ \quad D(i, j-1) + 1, & \\ \quad D(i-1, j) + 1, & j > 0,\ i > 0 \\ \quad D(i-1, j-1) + \mathrm{m}(S_1[i], S_2[j]) & \\ \} & \end{cases},$$

m(a,b)=0, если a==b, иначе 1

D(M,N)

Пусть S1 кончается на символ «a»,S2 кончается на символ «b». Есть три варианта:
1)Символ «a», на который кончается S1, в какой-то момент был стёрт. Сделаем это стирание первой операцией. Тогда мы стёрли символ «a», после чего превратили первые i-1 символов S1 в S2 (на что потребовалось D(i-1,j) операций), значит, всего потребовалось D(i-1,j)+1 операций
2)Символ «b», на который кончается S2, в какой-то момент был добавлен. Сделаем это добавление последней операцией. Мы превратили S1 в первые j-1 символов S2, после чего добавили «b». Аналогично предыдущему случаю, потребовалось D(i,j-1)+1 операци
3)Оба предыдущих утверждения неверны. Если мы добавляли символы справа от финального «a», то, чтобы сделать последним символом «b», мы должны были или в какой-то момент добавить его (но тогда утверждение 2 было бы верно), либо заменить на него один из этих добавленных символов (что тоже невозможно, потому что добавление символа с его последующей заменой неоптимально). Значит, символов справа от финального «a» мы не добавляли. Самого финального «a» мы не стирали, поскольку утверждение 1 неверно. Значит, единственный способ изменения последнего символа — его замена. Заменять его 2 или больше раз неоптимально. Значит,
1)Если a=b , мы последний символ не меняли. Поскольку мы его также не стирали и не приписывали ничего справа от него, он не влиял на наши действия, и, значит, мы выполнили D(i-1,j-1) операций.
2)Если a!=b мы последний символ меняли один раз. Сделаем эту замену первой. В дальнейшем, аналогично предыдущему случаю, мы должны выполнить D(i-1,j-1) операций, значит, всего потребуется D(i-1,j-1) +1