# Prime Number Algorithms

**Input:** ‘n’ a number to check that it is prime or not

**Output:** a boolean that indicates ‘n’ is prime or not

## Version 1

Check the divisibility of n by k for all k = 2, 3, …, (n-1)

### Method

for (k = 2; k < n; k = k + 1)

if (n mod k == 0)

return false;

return true;

### Analysis

Best case occurs when the number is even and worst case occurs when the number is prime

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Time | Best Case | Worst Case |
| k = 2 | 1 | 1 | 1 |
| k < n | 1 | 1 | n - 1 |
| k = k + 1 | 2 | 0 | n - 2 |
| n mod k == 0 | 2 | 1 | n - 2 |
| return false | 1 | 1 | 0 |
| return true | 1 | 0 | 1 |
| Total | | 5 | 5n - 7 |

## Version 2

If n is prime then n is odd

If n is odd then it is not divisible by k = 2, 4, ..., 8

### Method

if (n mod 2 == 0)

return false;

for (k = 3; k < n; k = k + 2)

if (n mod k == 0)

return false;

return true;

### Analysis

Best case occurs when the number is even and worst case occurs when the number is prime

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Time | Best Case | Worst Case |
| n mod 2 == 0 | 2 | 1 | 1 |
| return false | 1 | 1 | 0 |
| k = 3 | 1 | 0 | 1 |
| k < n | 1 | 0 | n/2 - 1 |
| k = k + 2 | 2 | 0 | n/2 - 2 |
| n mod k == 0 | 2 | 0 | n/2 - 2 |
| return false | 1 | 0 | 0 |
| return true | 1 | 0 | 1 |
| Total | | 3 | 2.5n - 5 |

## Version 3

If at all n is divisible it should be divisible by a number not exceeding n/2

### Method

if (n mod 2 == 0)

return false;

end = ceil (n / 2);

for (k = 3; k < end; k = k + 2)

if (n mod k == 0)

return false;

return true;

### Analysis

Best case occurs when the number is even and worst case occurs when the number is prime

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Time | Best Case | Worst Case |
| n mod 2 == 0 | 2 | 1 | 1 |
| return false | 1 | 1 | 0 |
| end = ceil (n / 2) | 3 | 0 | 1 |
| k = 3 | 1 | 0 | 1 |
| k < end | 1 | 0 | n/4 - 1 |
| k = k + 2 | 2 | 0 | n/4 - 2 |
| n mod k == 0 | 2 | 0 | n/4 - 2 |
| return false | 1 | 0 | 0 |
| return true | 1 | 0 | 1 |
| Total | | 3 | 1.25n - 2 |

## Version 4

Dividing n by k for k = 2 to sqrt(n)

### Method

if (n mod 2 == 0)

return false;

end = ceil(sqrt(n));

for (k = 3; k < end; k = k + 2)

if (n mod k == 0)

return false;

return true;

### Analysis

Best case occurs when the number is even and worst case occurs when the number is prime

|  |  |  |  |
| --- | --- | --- | --- |
| Step | Time | Best Case | Worst Case |
| n mod 2 == 0 | 2 | 1 | 1 |
| return false | 1 | 1 | 0 |
| end = ceil(sqrt(n)) | 3 | 0 | 1 |
| k = 3 | 1 | 0 | 1 |
| k < end | 1 | 0 | sqrt(n)/2 - 1 |
| k = k + 2 | 2 | 0 | sqrt(n)/2 - 2 |
| n mod k == 0 | 2 | 0 | sqrt(n)/2 - 2 |
| return false | 1 | 0 | 0 |
| return true | 1 | 0 | 1 |
| Total | | 3 | 2.5sqrt(n) - 2 |

## Average Case

For this case first we divide a range of numbers between 105 and 108 to 10 parts, then in each part, we pick 100 random samples and measure the average of execution time for them.