**Prime numbers1:** a positive integer number P is prime, P >1, if the only positive integers that evenly divide P are P and 1.e.g., 2, 3, 5, 7, 11, 13, ...

(The source of this discussion about prime numbers is the following: [Kenneth Rosen](https://www.amazon.com/s/ref=dp_byline_sr_book_1?ie=UTF8&text=Kenneth+Rosen&search-alias=books&field-author=Kenneth+Rosen&sort=relevancerank), *Discrete Mathematics and Its Applications*, 7th Edition, McGraw-Hill, 2011)

**Check if a given number P is prime:**

* **Basic** algorithm

If P = 2, then P is prime.

Otherwise, if **none** of the integers in the interval (1, P-1] divides P, then P is prime.

* **Better algorithm**

If P = 2, then P is prime.

Otherwise, if **none** of the integers in the interval (1, ] divide P, then P is prime.

**Note: Why considering integers up to** ?

if P is not prime, it can be factored into two factors a and b such that P = a \* b.

If both a and b > , then a\*b > P. So, either a or b < .

Therefore, we only need to check for values less than or equal to the square root.

* **Even Better algorithm**

If P = 2, then P is prime.

Otherwise, if 2 divides P, then P is **not** prime.

Otherwise, if **none** of the **odd** integers in the interval (1, ] divides P, then P is prime.

* **Even2 Better algorithm**

If P = 2, then P is prime.

Otherwise, if **none** of the **prime** integers in the interval (1, ] divides P, then P is prime.

**consider IsPrime2T.py and write the function is\_prime to check whether an input, n, is prime.**

**input:** positive integer

**output:** true or false whether the number is prime or not

**process:** check if the number is 2, if it is, return true

otherwise, check if the number is 1 or divisible by 2, then return false

otherwise, try to divide n by the numbers from 3 to sqrt(n)+1 with a step of 2 (to only consider odd numbers)

% python .\IsPrime2.py

% python

Python 3.6.7 .......

>>> import IsPrime2

>>> IsPrime2.is\_prime(2)

True

>>>