Kernel: SageMath 10.7

Finding Primes

MATH 242 Modern Computational Mathematics

Primality Testing

Write a function `isPrime(n)` that accepts a positive integer `n` and determines whether it is prime. Your function should return `True` if `n` is prime and `False` otherwise.

First plan your function on paper before writing any code.

In [1]:
```
def isPrime(n):
    # loop over possible divisors 2, 3, ..., n-1
    for i in range(2,n):
        # does i divide n?
        if n % i == 0:
            # found a divisor!
            return False

    # if we get here, then we didn't find a divisor
    return True
```

In [2]:
```
isPrime(5)
```

Out[2]:
```
True
```

In [4]:
```
isPrime(63)
```

Out[4]:
```
False
```

Listing Primes

Use your function `isPrime` to make a list of all primes up to some number N.

How fast is this?

How long would it take to list all primes up to one billion?

Can you think of a faster way to list the primes?

In [7]:
```
primes = []
nMax = 1000
for i in range(2, nMax):
    if isPrime(i):
        primes.append(i)
print(primes)
```

Out[7]:
```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71,
73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151,
157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233,
239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317,
331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419,
421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503,
509, 521, 523, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599, 601, 607,
613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691, 701,
709, 719, 727, 733, 739, 743, 751, 757, 761, 769, 773, 787, 797, 809, 811,
821, 823, 827, 829, 839, 853, 857, 859, 863, 877, 881, 883, 887, 907, 911,
919, 929, 937, 941, 947, 953, 967, 971, 977, 983, 991, 997]
```

In [0]: