Dale Chetney

--How it works—

Step 1: Generate primes up to height\*width

Make an array of Integers.  
Store the sqare root of the upper bound.  
Make an array of bools that’s associated with each number up to upper bound.  
Starting at i=2, and going to the square root, check if that number’s associated bool is false.  
If it is, add I to the array of numbers, and start at j=i2.  
Incrementing by I until you reach the upper bound, set j’s associate bool to true.  
Now start at I = square root and go up to upper bound.  
If I’s associate bool is false, add I to the list.

Step 2: Write a message

For each letter index i of the message, get the next prime on the list.  
get the color at position prime%width, prime/width.  
int letter <- charAt(i) – 32  
color’s red’s 1st bit <- letter’s 5th bit.  
red’s 0th bit <- letter’s 4th bit.  
green’s 1st bit <- letter’s 3rd bit  
green’s 0th bit <- letter’s 2nd bit  
blue’s 1st bit<-letter’s 1st bit  
blue’s 0th bit<-letter’s 0th bit

Step 3: Read a Message

For every prime under height\*width, get the color at prime%width, prime/width.   
declare an empty String message  
int letter <- 0.  
letter’s 5th bit<-red’ s1st bit.  
letter’s 4th bit<-red’ s0th bit.  
letter’s 3th bit<-red’ s1st bit.  
letter’s 2th bit<-red’ s0th bit.  
letter’s 1th bit<-red’s 1st bit.  
letter’s 0th bit<-red’s 0st bit.  
letter is raised by 32  
add letter to message  
continue until there are no left in the prime iterator.

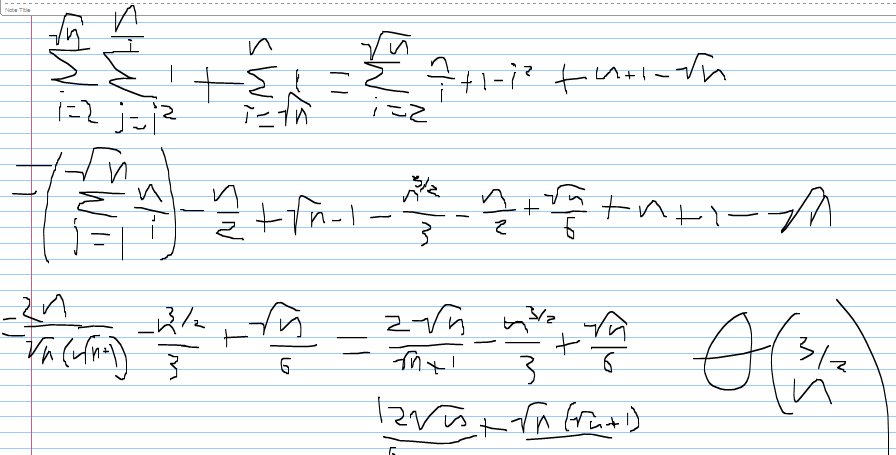
--Outside Help –

Luis Andrade suggested to me how to test my algorithm. Apart from that, I worked in isolation. It’s a fault of mine.

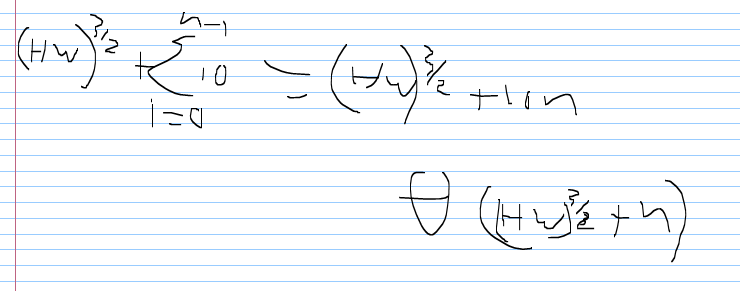
--Performace evaluation –

The number of computations where  
N = the length of the message in characters,  
H and W = Height and Width of the image,  
is:

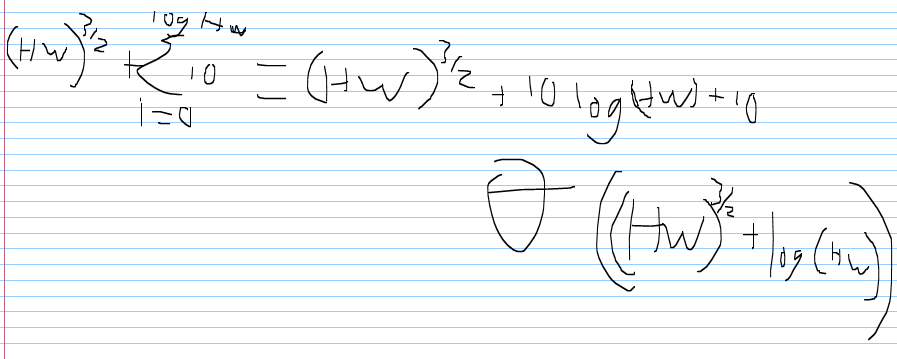
Prime Iterator:



Write:



Read:



If we disregard the PrimeIterator algorithm, our write is Θ(n) and our read is Θ(log(h\*w))

--Performance Testing—

The test is to write a message to the picture and then read it.

|  |  |  |  |
| --- | --- | --- | --- |
| Width | Height | Length | Time |
| 4 | 4 | 2 | 0 ms |
| 4 | 4 | 6 | 0 ms |
| 48 | 48 | 25 | 3 ms |
| 48 | 48 | 180 | 5 ms |
| 294 | 294 | 180 | 96 ms |
| 294 | 294 | 720 | 100 ms |