Santiago Figueroa

Duvan García Tovar

María Alejandra Estacio

**1.Identificación del problema:**

**Contexto problemático**

Una parte fundamental de la mayoría del software producido actualmente es la seguridad de los datos, esto es, mantener la triada CID: Confidencialidad, Integridad y Disponibilidad de la información. Para esto, hay una rama de la informática que juega un papel muy importante llamada Criptografía. Ahora bien, la pequeña empresa para la que trabajamos ha decidido tomarse en serio la seguridad de sus sistemas y realizar sus propias implementaciones de algunos algoritmos de encriptación, para eso es necesario, poder contar con un programa que permita la generación de números primos que posteriormente podrán ser utilizados por los algoritmos.

**Determinación del problema**

Según el enunciado, dado un número n, el programa debe estar en capacidad de calcular y mostrar en una matriz, lo más cuadrada posible, en verde los números primos desde 1 hasta ese n y en rojo los que no lo son. Además de eso, implementar 3 algoritmos diferentes para que el usuario decida cuál usar.

Eso quiere decir, que las especificaciones a cumplir son:

1. Calcular los números primos entre 1 y un número digitado por el usuario.
2. Implementar 3 algoritmos para el cálculo de los números primos y que el usuario pueda elegir con cuál de ellos trabajar.
3. El sistema debe contar con una Interfaz Gráfica de Usuario que permita ingresar el número máximo o tope (n) para la búsqueda de los números primos.
4. Mostrar en una matriz lo más cuadrada posible de acuerdo al tope(n).
5. Visualizar el cálculo de los números primos en tiempo real, coloreando de verde los que son y de rojo los que no.

**2. Búsqueda de información necesaria:**

El antiguo matemático griego Euclides demostró que hay una cantidad infinita de números primos en su gran obra llamada Los Elementos. Desde hace muchos, muchos siglos, se ha tratado de dar con una forma de encontrar números primos. El antiguo matemático griego, Eratóstenes creó un método para determinar los números primos a partir de una lista de números, este método se le conoce como la criba de Euclides, y consiste en lo siguiente.

### Algoritmo de Eratóstenes

Si queremos conocer los números primos entre el 1 y el 50, fijémonos en esta lista. Ya sabemos que el 1 no es primo, entonces lo tachamos, de ahí le sigue el dos, el cual sabemos que es un número primo, entonces tachamos todos los múltiplos de dos, el cuatro, el ocho, el diez y así sucesivamente.

Ahora sigue el número tres que también es primo, entonces de la misma manera ahora tachamos todos los múltiplos de tres, el seis ya está, tachamos el 9, el quince, al veintiuno y seguimos sucesivamente.

Si repetimos este proceso con el cinco y con el 7, los números que quedan sin tachar son primos, los cuales son el 2, 3 ,5, 7, 11,13,17, 19, 23, 29, 31, 37, 41, 43, y 47. El problema de encontrar primos ha perdurado, a través de los siglos.

#### Test de primalidad

Un Test de Primalidad es un algoritmo que permite decidir si un número natural n es primo o compuesto

#### Primos de Mersenne

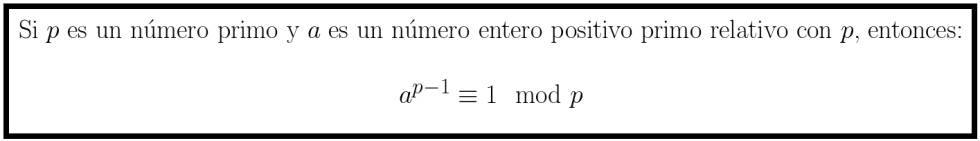
Para hallar números primos muy grandes no podemos ir probando todos los números en orden, ya que no sería computable en un tiempo razonable.

Mersenne (1588-1648) propuso que los números de la forma (2^p) −1 podrían ser primos, sabiéndose en la actualidad además que tienen propiedades muy deseables para eliminar candidatos que no sean primos.

Estos primos pueden llegar a ser un poco difíciles de manejar, pero son muy útiles y podemos usarlos para sistemas muy robustos de encriptación. Actualmente la asociación GIMPS tiene una red distribuida para hallar primos de Mersenne que tiene el récord de haber hallado el último número primo mencionado anteriormente.

#### Pequeño teorema de Fermat

Si p es un número primo y a es un número entero positivo que no tiene factores comunes con p (es decir, a y p son primos relativos), entonces el resto de la división de ap-1 entre p es 1. En términos de congruencias, este teorema queda así:



En principio, este teorema puede ser muy útil para descartar que un número es primo, ya que para que un cierto número n sea primo es necesario que se cumpla el pequeño teorema de Fermat para todo entero positivo a menor que el propio n.

#### Test de primalidad de Miller-Rabin

Su versión original fue propuesta por G. L. Miller, se trataba de un algoritmo determinista, pero basado en la no demostrada [hipótesis generalizada de Riemann](https://es.wikibooks.org/w/index.php?title=Hip%C3%B3tesis_generalizada_de_Riemann&action=edit&redlink=1); [Michael Oser Rabin](https://es.wikibooks.org/w/index.php?title=Michael_Oser_Rabin&action=edit&redlink=1) modificó la propuesta de Miller para obtener un [algoritmo probabilístico](https://es.wikibooks.org/w/index.php?title=Algoritmo_probabil%C3%ADstico&action=edit&redlink=1) que no utiliza resultados no probados.

Teorema de los números primos

El número de primos menores o iguales a x, , satisface la relación asintótica:

**Efectos prácticos.**

Proceso de encriptación

Encriptar un número es cambiarlo por otro, de forma que, aunque alguien averigüe el número codificado y conozca cómo se codifica, no sea capaz de averiguar el número original. Elevando a una potencia grande, después dividiendo esa potencia entre otro número que sirva de tope. Con eso, tenemos un número diferente y para recuperarlo solo debemos elegir bien el exponente y el divisor. Si elegimos como divisor el producto de dos números primos, las matemáticas permiten encontrar un exponente para encriptar y otro para des encriptar**,** una clave pública y una **privada** respectivamente.

Clave pública: Es un número que todo el mundo puede conocer y con el que te pueden encriptar mensajes para que luego tú los des encripta con tu clave privada.

Clave privada: Es un número que solo tú conoces y te sirve para recuperar el mensaje original después del proceso de encriptado.

Factorización de números compuestos:

Los números primos son fundamentales porque constituyen la base de la matemática, debido a que todos los demás números (compuestos) se pueden expresar como multiplicaciones de ellos. Por ejemplo:

15 = 3 \* 5.

Y de esta manera se puede realizar con cualquier otro número. Por esto, se dice que los números primos son semejantes a los átomos para las matemáticas.

**3. Búsqueda de soluciones creativas:**

Mediante una lluvia de ideas que buscaba las mejores soluciones para el problema de encontrar algoritmos que encuentren números primos obtuvimos que:

**Alternativa 1.** División por tentativa.

Consiste en dividir un número entero n, entre todo número primo (2, 3, 5, 7…) menor o igual a √n. Si al hacer la división su resto es igual a cero, entonces el número n no es primo.

**Alternativa 2.** Método de factorización de Fermat.

Este teorema afirma que si n es primo y mcd (a, n) = 1, entonces an-1 ≡ 1 (mod n).

**Alternativa 3.** Método de factorización de Euler.

Este resultado afirma que si n es primo y mcd (a, n) =1, entonces a(n-1) /2 ≡ ±1 (mod n).

**Alternativa 4.** Teorema de Wilson.

Es un resultado de teoría de números vinculada con la divisibilidad y la primabilidad de números enteros, el teorema consiste en que si p es un número primo entonces: (p−1)! ≡−1(mod p). Por lo tanto, si p es un número primo entonces.

**Alternativa 5.** Base de datos de números primos.

Tener una base de datos que contenga todos los números primos conocidos hasta ahora (o hasta un número bastante grande), junto al teorema del número primo controlar cuántos primos hemos de “hallar” y representarlos cuando un usuario use el programa.

**Alternativa 6.** Método de Eratóstenes.

La criba de Eratóstenes es un algoritmo que permite encontrar los números primos dado un número natural N.

* Se crea una lista que va desde 2 hasta el número determinado.
* Se elimina de dicha lista los múltiplos de dos.
* Después se toma el siguiente número del 2 que no fue eliminado y se eliminan sus múltiplos, y así sucesivamente.
* El proceso termina cuando el cuadrado del mayor número primo confirmado es menor que el número final de la lista.
* Los números que permanecen en la lista son primos.

**Alternativa 7.** Teorema de Proth

Números primos de Proth satisfacen el teorema de Proth, es decir, un número N de esta forma es primo si y sólo si existe un número a tal que , es congruente con -1 módulo N entonces n es un número primo llamado primo de Proth. Este test funciona en la práctica porque si n es primo, el 50% de los valores de a cumplen con la condición indicada arriba.

**4. Transición de la formulación de ideas a los diseños preliminares:**

Dadas las alternativas anteriores decidimos desistir de:

* Alternativa 4: Debido a que, para una entrada grande, calcular (p-1)! es bastante tedioso. Por lo cual, se dedujo que el Teorema de Wilson tiene gran valor teórico mas no practico.
* Alternativa 2: Debido a que la fórmula no es muy precisa porque a veces arroja un número aparentemente primo pero en realidad no lo es, Ej. 561, factor de 3,11,17 y
* Alternativa 7: Este teorema no se cumple para todos los casos. Al tener una efectividad del 50% hace que sea muy propenso a fallar. Los primeros números que se pueden calcular con esta fórmula son 3, 5, 13, 17, 41, 97, 113, 193 y como se puede observar este método se salta números primos como 19 y 23.

Con las alternativas restantes tenemos que:

* Alternativa 1: Es una buena forma de calcular los números primos hasta un valor n y mostrarlos mientras se hallan, aunque tiene el problema de que para entradas demasiado grandes es poco eficiente.
* Alternativa 3: Es una buena forma de encontrar números primos para valores n considerablemente grandes, o por lo menos, los valores donde la alternativa 1 deja de ser una opción.
* Alternativa 5: No es una sorpresa que el tiempo de consulta o de comparación es de lo más eficiente posible. además, se conocen bastantes números primos. Entonces, ¿por qué no enlistarlos? y así lo que se hace al momento de digitar el N por el usuario, solo sería buscar en esta lista los primos hasta ese valor N y mostrarlos.
* Alternativa 6: Es muy simple representar la serie de pasos de este método en el lenguaje de programación en Java lo cual facilita la implementación.

**5. Evaluación y selección de la mejor solución.**

Al iniciar con una nueva etapa del método de la ingeniería, debemos definir algunos criterios en los que nos basaremos para realizar la selección de la idea o la manera más eficiente para resolver el problema que se trata a la lo largo de este documento.

**Criterios para la selección de la mejor solución:**

● **Consumo de memoria**

○ Determina el espacio en memoria que gasta el algoritmo para generar los números primos.

■ Escala de 1 a 5 siendo: 1 (n^2) y 5 un espacio constante k.

● **Tiempo de ejecución de la solución**

○ Determina el tiempo que toma un algoritmo para generar y encontrar los números primos

■ Escala de 1 a 5 siendo: 1 (2^n) y 5 un tiempo constante *k*

● ● **Proximidad/exactitud de la solución**

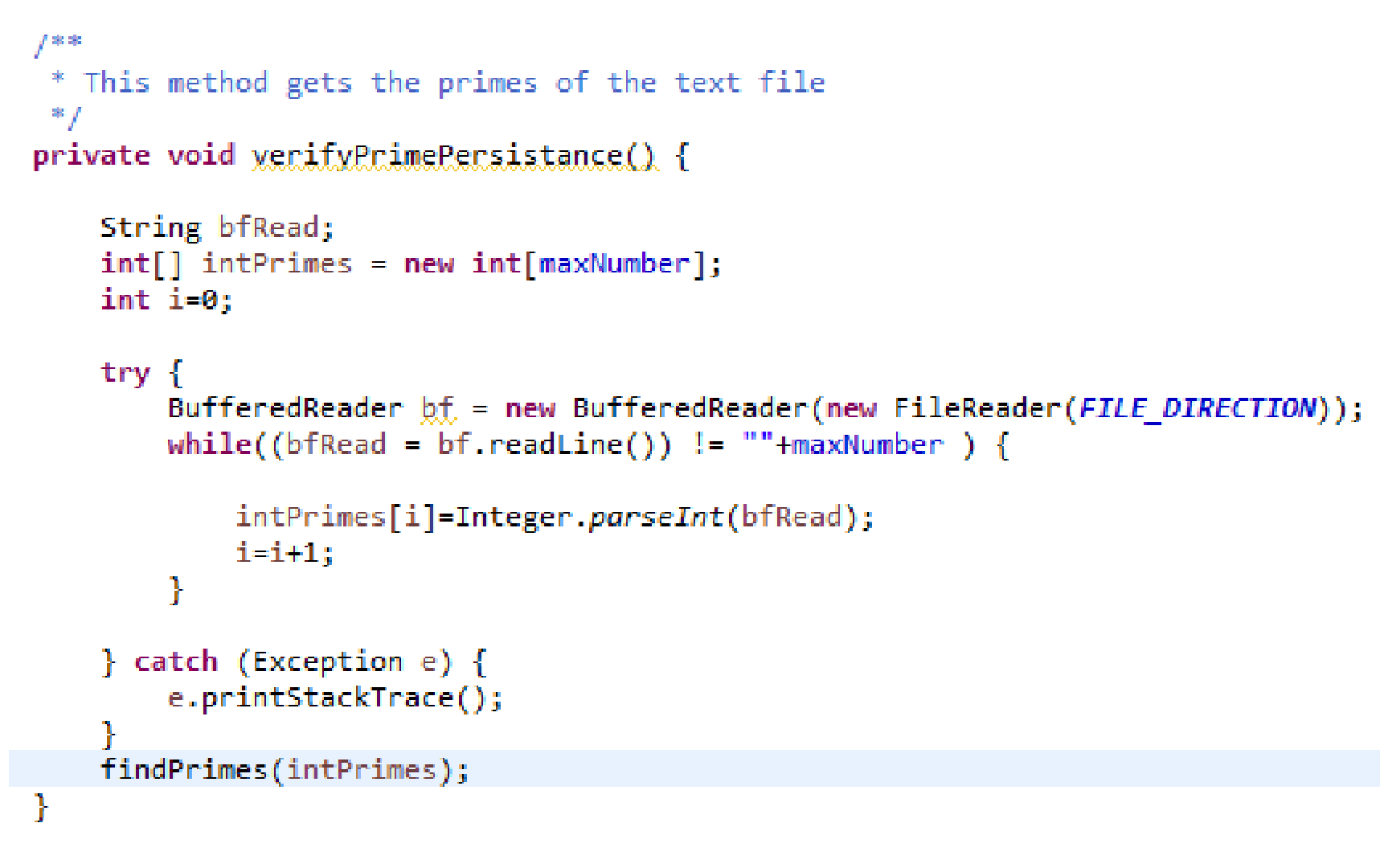
○ Determina el nivel de precisión con el que un algoritmo calcula los números primos

■ Escala de 1 a 5 siendo: 1 no determina la solución y 5 la(s) solución(es) determinada(s) es/son correcta(s)

Análisis de complejidad temporal y espacial

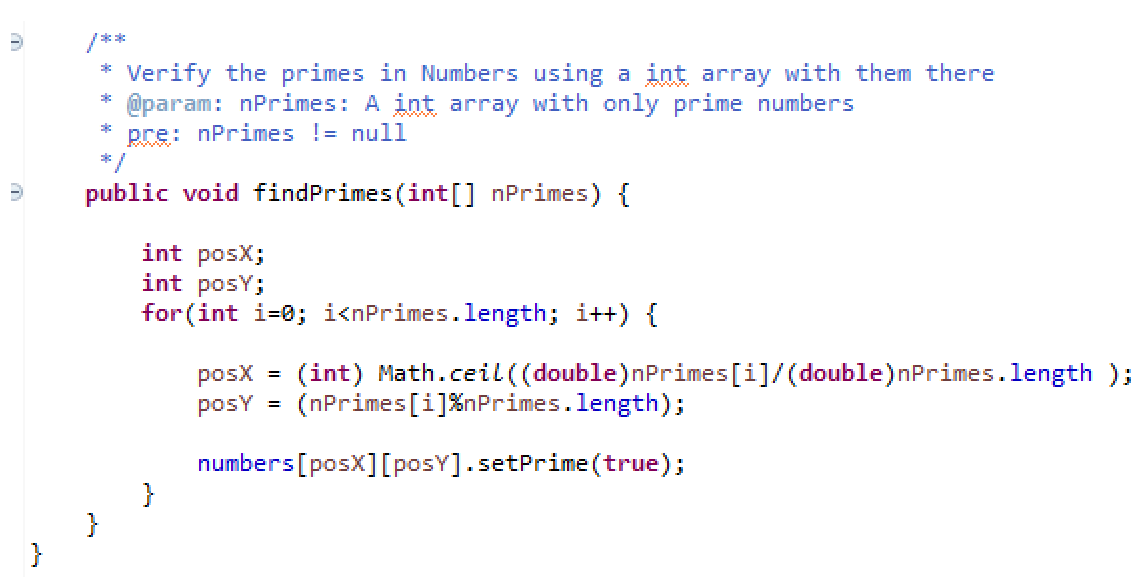
Aquí se muestra entonces el análisis de complejidad temporal de los 3 algoritmos para encontrar los números primos utilizados en el laboratorio.

1. Algoritmo por el método de persistencia:

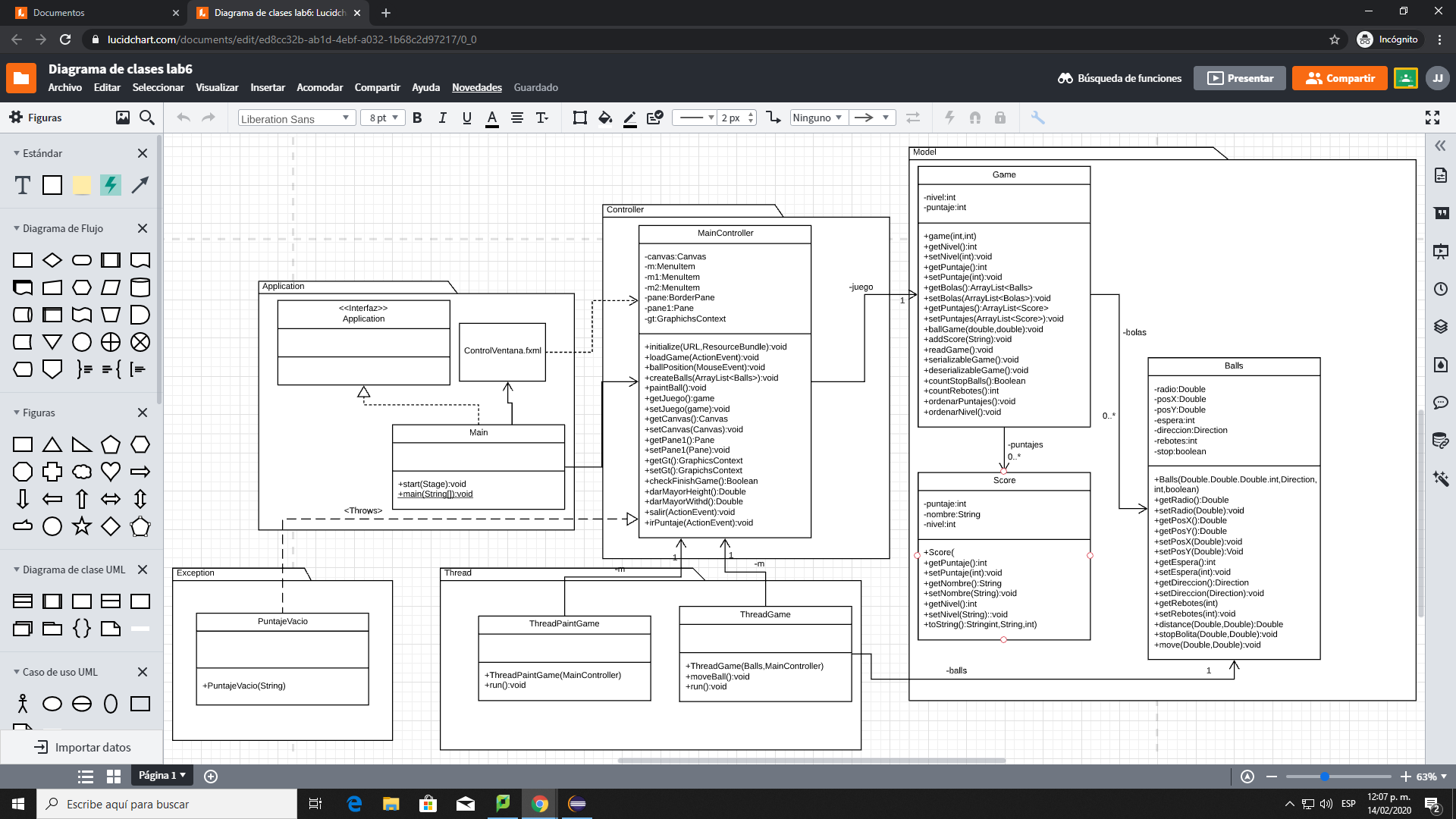


como podemos ver en el peor de los casos tenemos siendo n el número de primos que hay hasta el valor digitado por el usuario en la parte de la complejidad temporal. Ahora, la complejidad espacial es

1. Algoritmo por el método de Eratóstenes
2. Algoritmo por el método de Euler
3. Algoritmo que “colorea” los números primos:



En el peor de los casos tenemos siendo n el tamaño del arreglo que contiene solo los números primos. Ahora, la complejidad espacial es



<https://www.lucidchart.com/invitations/accept/b3970211-5630-4a34-8783-24f496322d8b>

<http://mimosa.pntic.mec.es/jgomez53/matema/conocer/primos.htm>

<https://ernestomataplata.me/divulgacion/matematicas-numeros-primos/>

<https://interferencias.tech/2017/11/07/encriptacion-numeros-primos/>

<https://www.geogebra.org/m/fjxtmmpu>

<https://es.wikibooks.org/wiki/Implementaci%C3%B3n_de_algoritmos_de_teor%C3%ADa_de_n%C3%BAmeros/Test_de_primalidad_de_Miller-Rabin>

<https://elpais.com/elpais/2017/08/24/el_aleph/1503587455_519490.html>

<http://gaceta.rsme.es/abrir.php?id=250>

<https://digital.csic.es/bitstream/10261/31196/1/Articulo49.pdf>