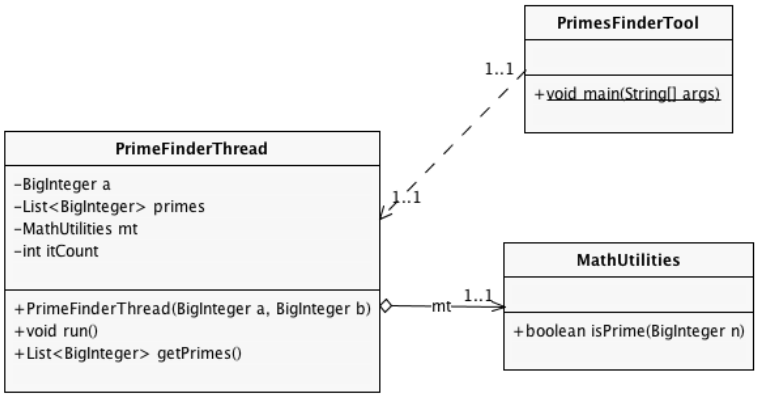
Escuela Colombiana de Ingeniería
Arquitecturas de Software – ARSW
Parcial Primer Tercio – Parte práctica.

Para una plataforma de procesamiento criptográfico de datos se requiere de una herramienta que genere un gran volumen de números primos (básicos para esquemas de seguridad de llave pública). Por ahora, la herramienta funciona sólo localmente (en un solo computador), pero igual se espera que la misma:

- Haga uso de los tiempos 'ociosos' del computador para realizar la búsqueda de los números primos. Es decir, que los cálculos sólo se hagan mientras no haya una persona usando la máquina.
- Aproveche la arquitectura de múltiples núcleos del computador (en este caso 4) a través de la paralelización de las tareas de cálculo.
- Permita fácilmente la incorporación de nuevos algoritmos de cálculo de números primos.

Por ahora, la herramienta se basa en el siguiente modelo de clases:



Donde básicamente se crean N hilos (de la clase PrimeFinderThread), y cada uno de éstos busca dentro de un intervalo diferente su correspondiente conjunto de números primos. Por ahora se usa el método de verificación de primos provisto por la clase MathUtilities, basado en el antiguo algoritmo conocido como la 'Criba de Eratóstenes'.

1. Corrija lo que haga falta para que el programa muestre correctamente los resultados (por ahora muestra un resultado diferente cada vez que se ejecuta). Adicionalmente, haga los ajustes necesarios para que el método de verificación de primos quede desacoplado de los hilos como tal, y que

permita –por ahora- intercambiar fácilmente entre el algoritmo de la clase *MathUtilities* y el de la clase *MathTools* (que usa un algoritmo alternativo).

2. Se quiere tener la posibilidad de incorporar un mecanismo de 'perfilación' (medición de desempeño) de la configuración actual de la herramienta. Para esto, haga un aspecto que –cuanto esté activado- calcule continuamente el tiempo promedio que toman los cálculos. Dicho tiempo promedio debe mostrarse por pantalla SÓLO al terminar de ejecutarse el proceso. Con esto, debe ser posible comparar el desempeño de cualquiera de los dos algoritmos utilizados.

3. Se debe controlar la ejecución de los hilos de cálculo de primos, de manera que éstos sólo trabajen mientras el equipo NO esté siendo utilizado por una persona (ya que éstos consumen mucha CPU y harían el equipo inutilizable). Por ahora, se va a asumir que el equipo estará 'en uso' cuando hayan transcurrido menos de 10 segundos desde la última vez que se movió el Mouse, en cuyo caso el procesamiento debería pausarse (es decir, los hilos deberían 'dormirse'). De la misma manera, en cuanto se detecte que el Mouse NO se ha movido recientemente (más de 10 segundos), los cálculos deberían reanudarse justo en el punto en el que fueron pausados. Para esto, se le ha suministrado la clase MouseMovementMonitor (junto con un ejemplo de uso en la clase MouseMovementMonitorExample), la cual permite consultar el tiempo transcurrido –en milisegundos- desde la última vez que se movió el Mouse.
   Nota: para probar esta funcionalidad, aumente el tamaño de los intervalos (por ejemplo [1..1000000000]), de manera que haya tiempo de verificar que se han correctamente las pausas.