

**INSTITUTO POLITÉCNICO NACIONAL**

**ESCUELA SUPERIOR DE CÓMPUTO**

DESARROLLO DE SISTEMAS DISTRIBUIDOS

Carlos Pineda Guerrero

**TAREA 1**

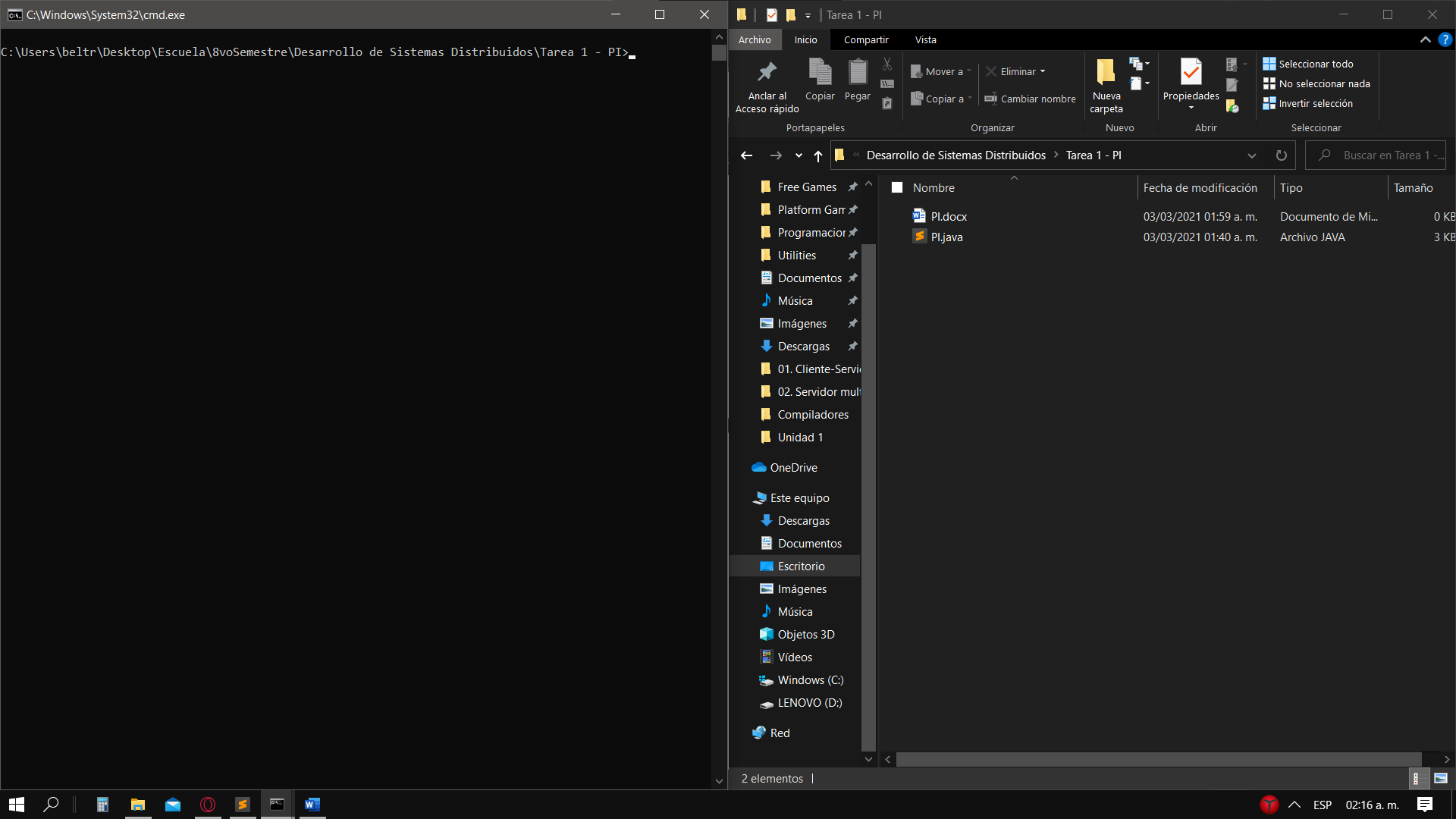
**Calculo distribuido de PI**

GRUPO: 4CV4

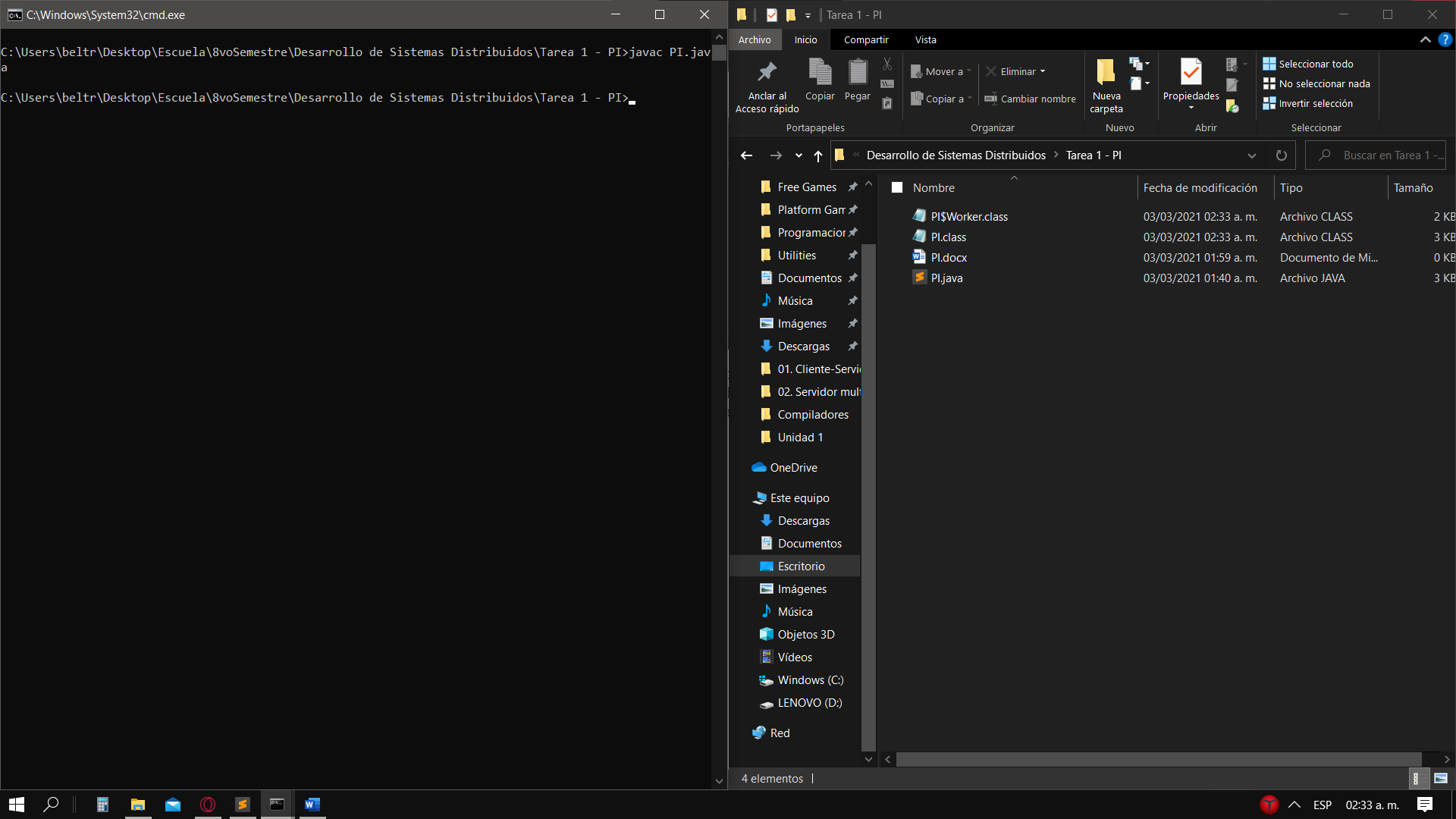
ALUMNO:

Beltrán Orozco Isaac

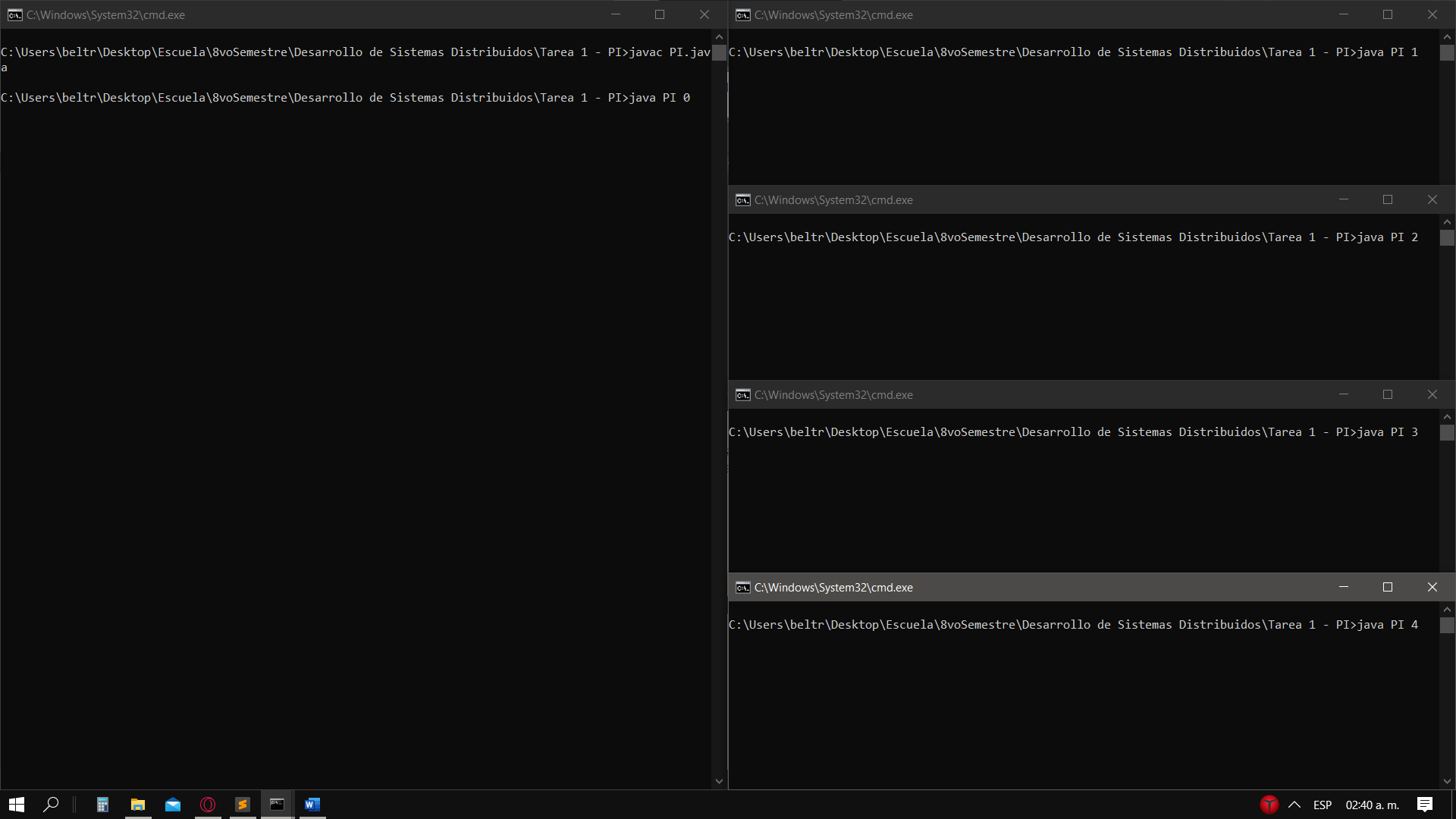
Desarrollo:



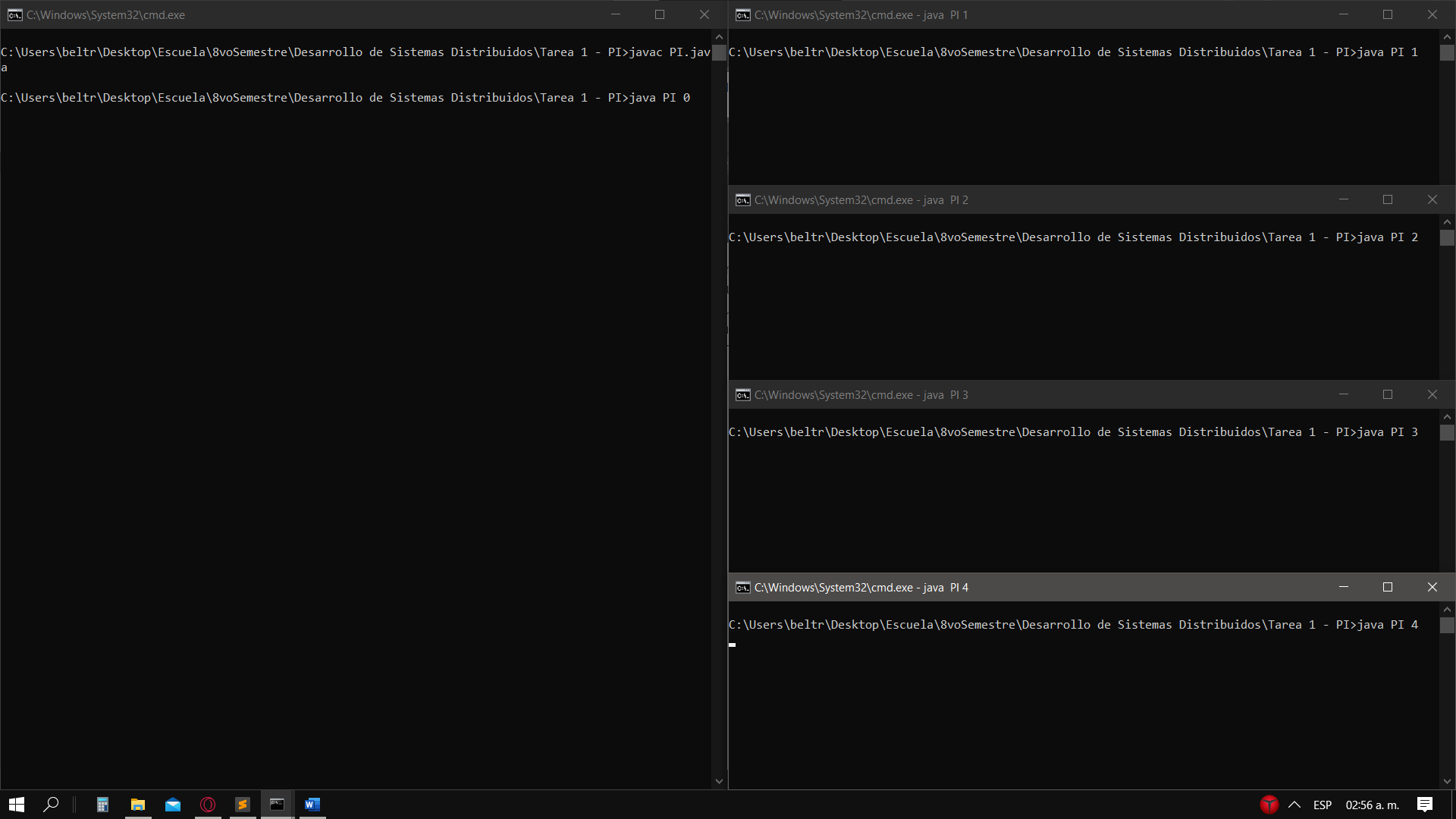
En esta captura vemos a la izquierda la línea de comandos, ubicada en la misma carpeta que se muestra a la derecha (pero esta en el administrador de archivos de Windows), en la cual podemos ver que se encuentra nuestro archivo PI.java antes de la compilación.



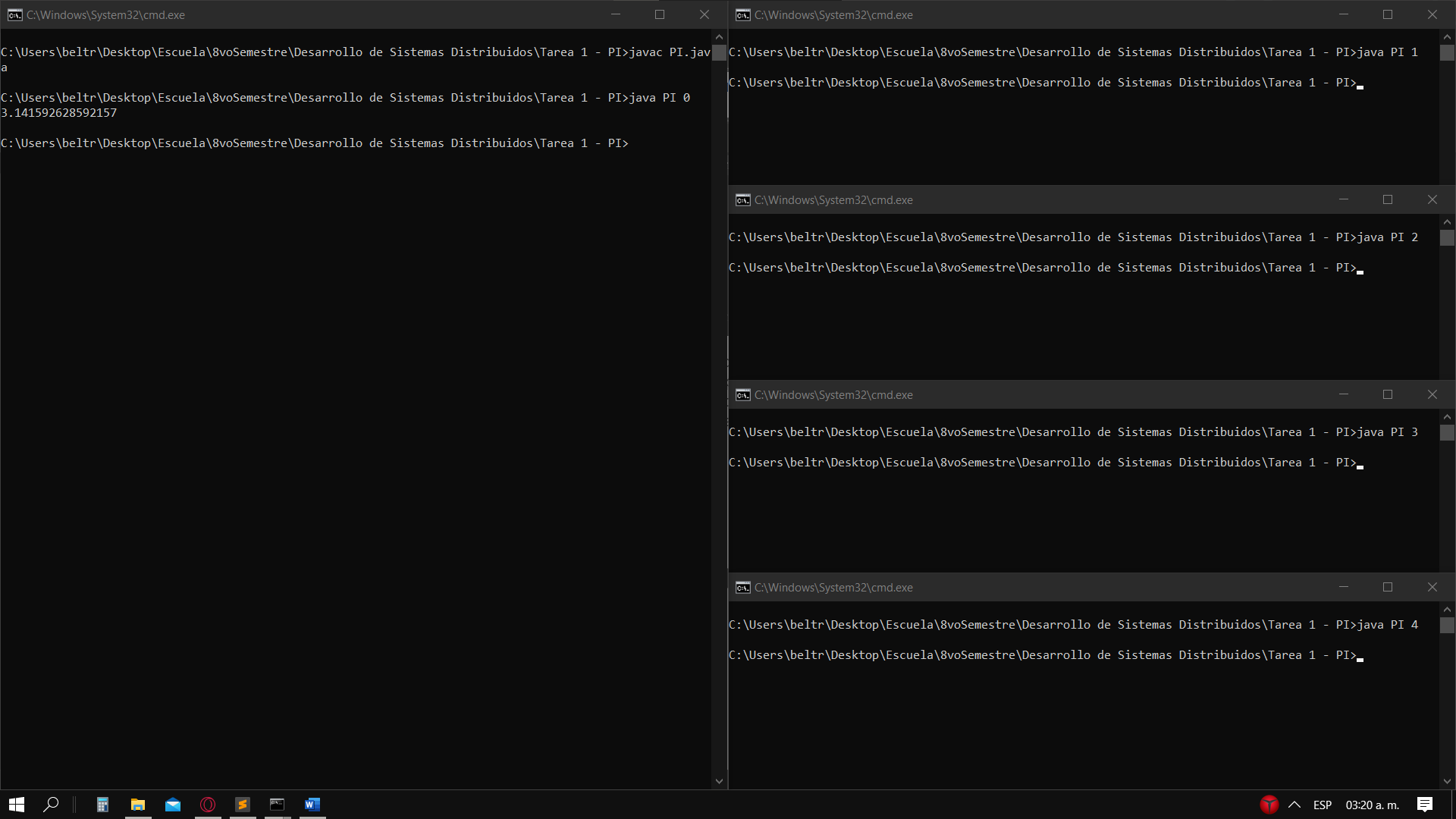
Ahora observamos el mismo panorama, pero ya después de la compilación, pues en la línea de comandos podemos apreciar que se ha ejecutado la instrucción “javac PI.java” sin ningún problema y a la derecha ya se encuentra nuestros archivos .class resultantes de dicha compilación.



En esta nueva captura, a la izquierda, seguimos teniendo la instancia original de la línea de comandos (en la que compilamos nuestro programa), esta será la que funja como nuestro servidor; pero ahora a la derecha se encuentran otras cuatro instancias de cmd.exe, las cuales serán las que actúen como nuestros clientes. En cada una de estas instancias (incluyendo en la original) se ha colocado la instrucción que respectivamente realizaran, pero aun no ejecutamos ninguna de estas instrucciones.



Ahora, como podemos apreciar en la cabecera de cada una de ellas, hemos ejecutado nuestro programa en todas las instancias de la derecha con el respectivo parámetro que habíamos anticipado. Debido a que los clientes están programados para intentar la conexión con el servidor hasta que tengan éxito, por el momento se encuentran a la espera de que el servidor se ejecute para empezar a realizar sus operaciones.



Finalmente ejecutamos nuestro servidor en la línea de comandos de la derecha, todos los clientes terminan su ejecución, y vemos como en el mismo lado derecho, debajo de la instrucción que pone nuestro servidor en marcha, aparece una aproximación bastante certera del valor de PI, seguida de la ubicación de la carpeta en la que nos encontrábamos, marcando así la finalización de PI.java de manera exitosa.

Conclusiones:

En esta tarea se puede observar lo útil y flexible que es el uso de multithread, pues vemos como la operación original, el calculo de PI por medio de la serie Gregory-Leibniz, es una tarea operacionalmente costosa, constando de un numero considerable de sumas, multiplicaciones y divisiones. Sin embargo, al haber dividido dichas operaciones entre 4 hilos, la carga para cada uno de ellos se aligera considerablemente y la velocidad con la que problema es resuelto se acorta bastante. También se pueden apreciar detalles importantes y cruciales para el correcto funcionamiento de este tipo de sistema, como lo importante que se vuelve la definición certera de los puntos críticos, o la adecuada generación y destruccion de los distintos threads.