PRÁCTICA 1.1

## ¿Calcular cuántos años más podremos seguir utilizando esta forma de contar? Explica el razonamiento seguido para realizar el cálculo.

En 64 bits se pueden representar 2^64 números, 1.8446744e+19 milisegundos, lo que es igual a 5.849.424.150.177.575 años. Si le restamos los 56 años que han pasado desde el 70 obtenemos la solución.

5,8 \*10^8 – 56 años hasta que llegue al tope la cuenta.

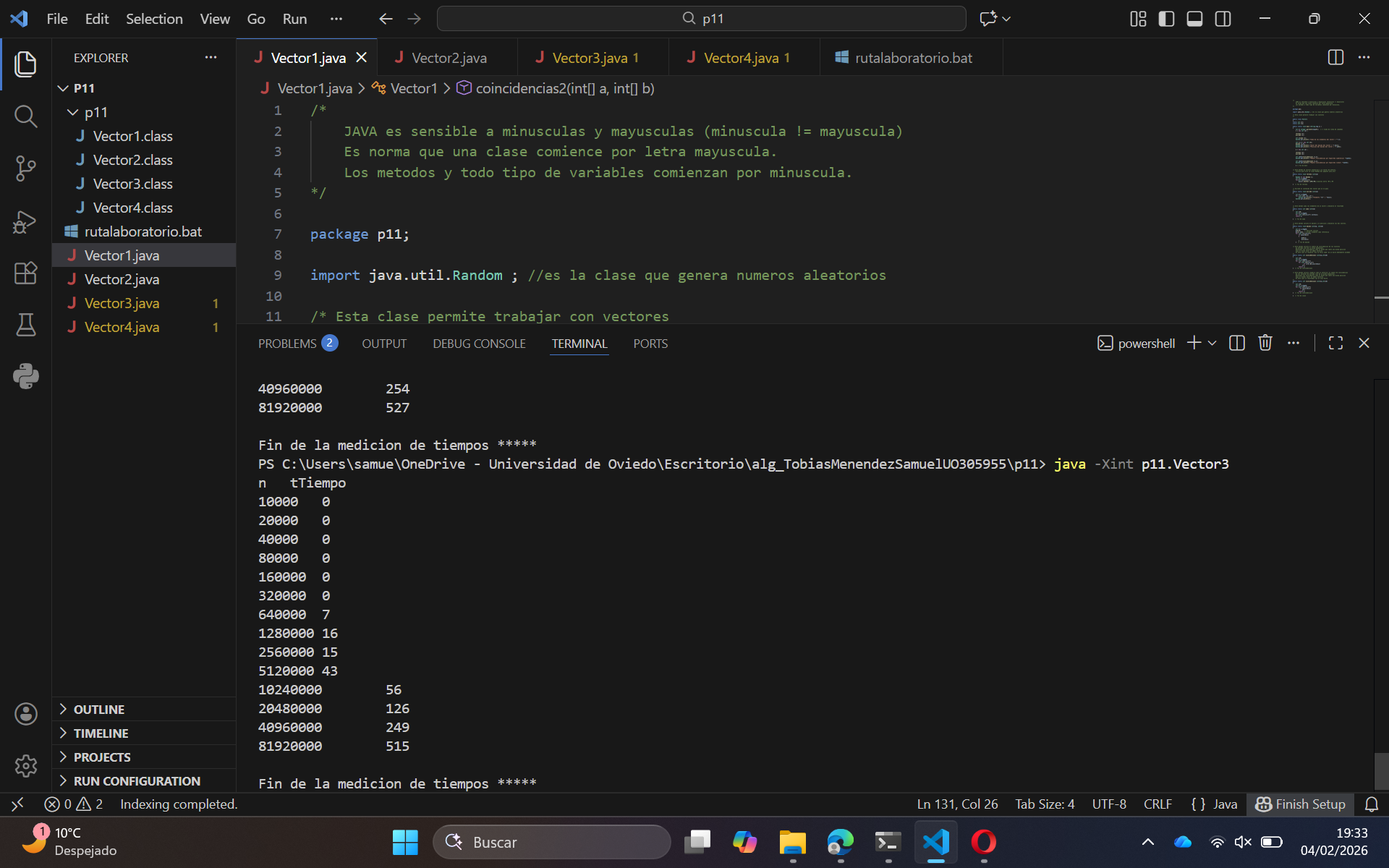
## ¿Por qué a veces el tiempo medido sale 0? ¿A partir de qué tamaño de problema (n) empezamos a obtener tiempos fiables?

Aunque en mis cálculos no pasa, a veces puede salir tiempos muy inferiores o 0 debido a usar la optimización. La optimización contiene el JIT que no interpreta el código línea a línea y debido a esto salta líneas innecesarias para la salida.

Si consideramos tiempos por debajo de 50ms despreciables, a partir de los 10mill ya son tiempos fiables.

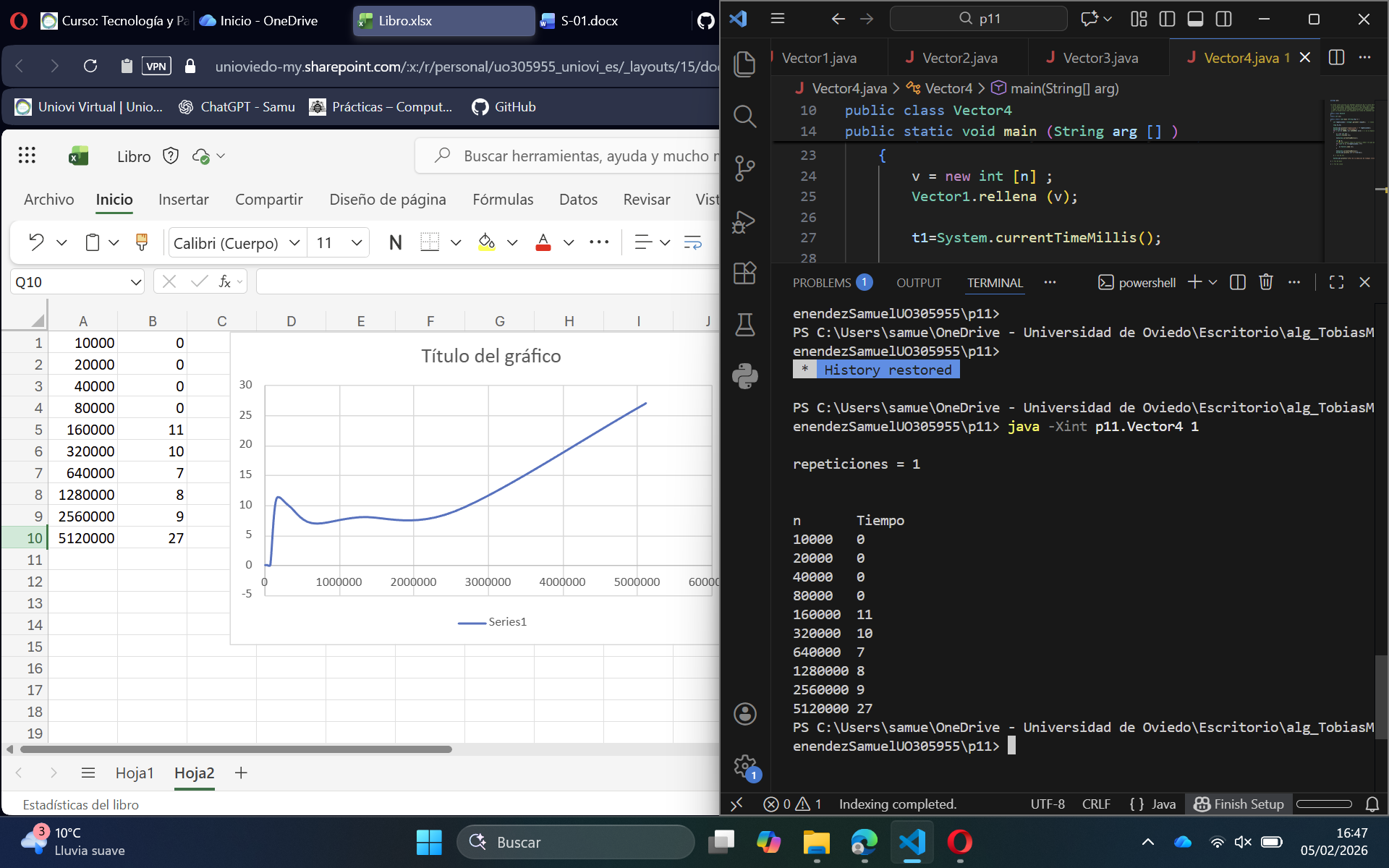
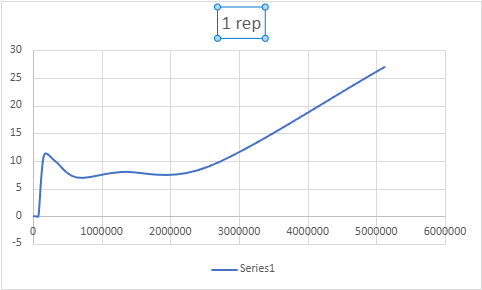
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Repeticiones | 1mill | 10mill | 20mill | 50mill | 100mill | 200mill |
| Tiempo (ms) | 8 | 74 | 153 | 312 | 653 | 1469 |

### Vector 3

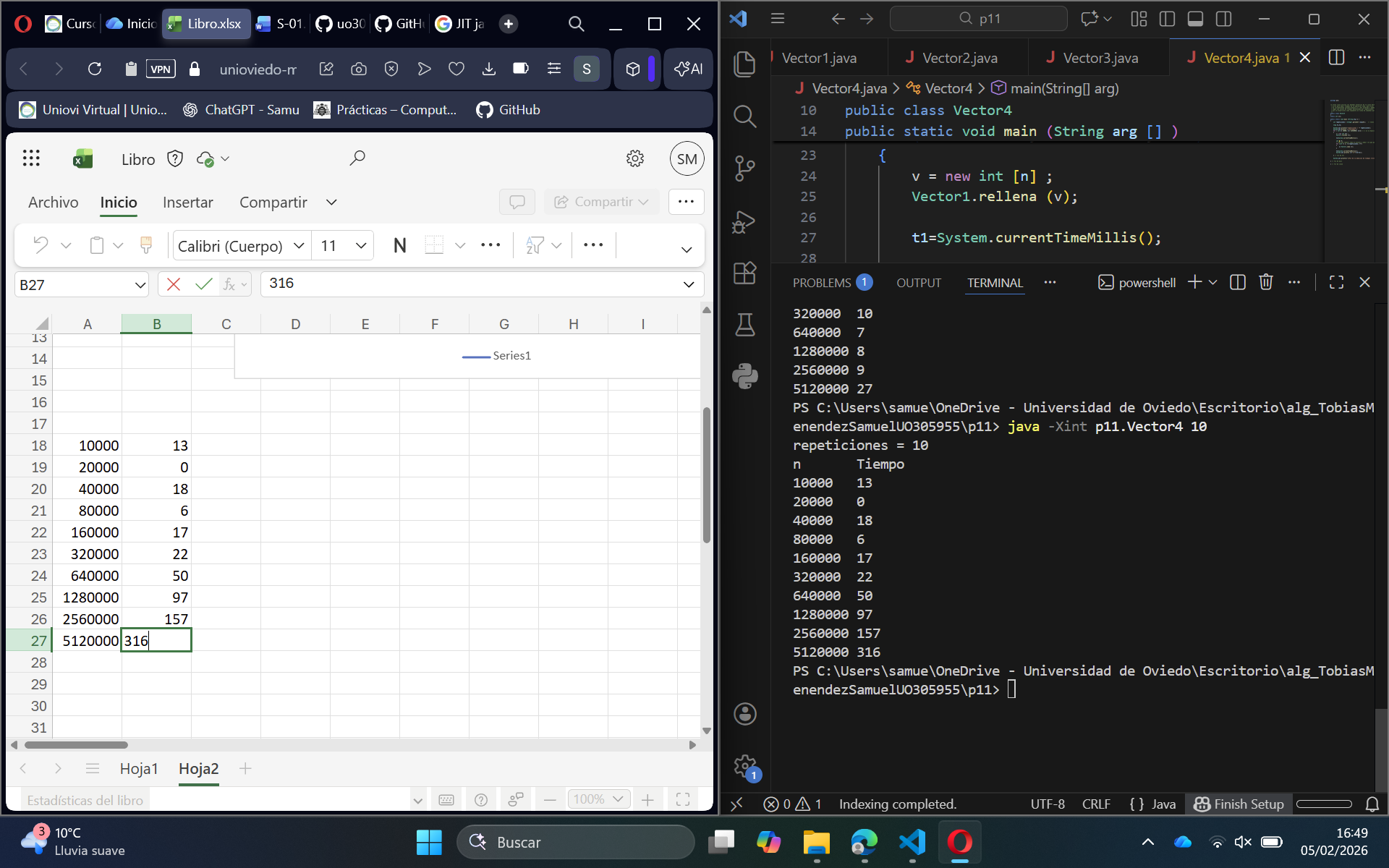


## Vector 4

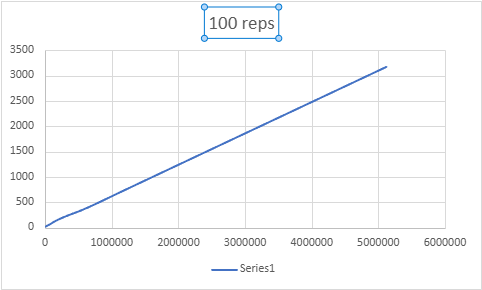
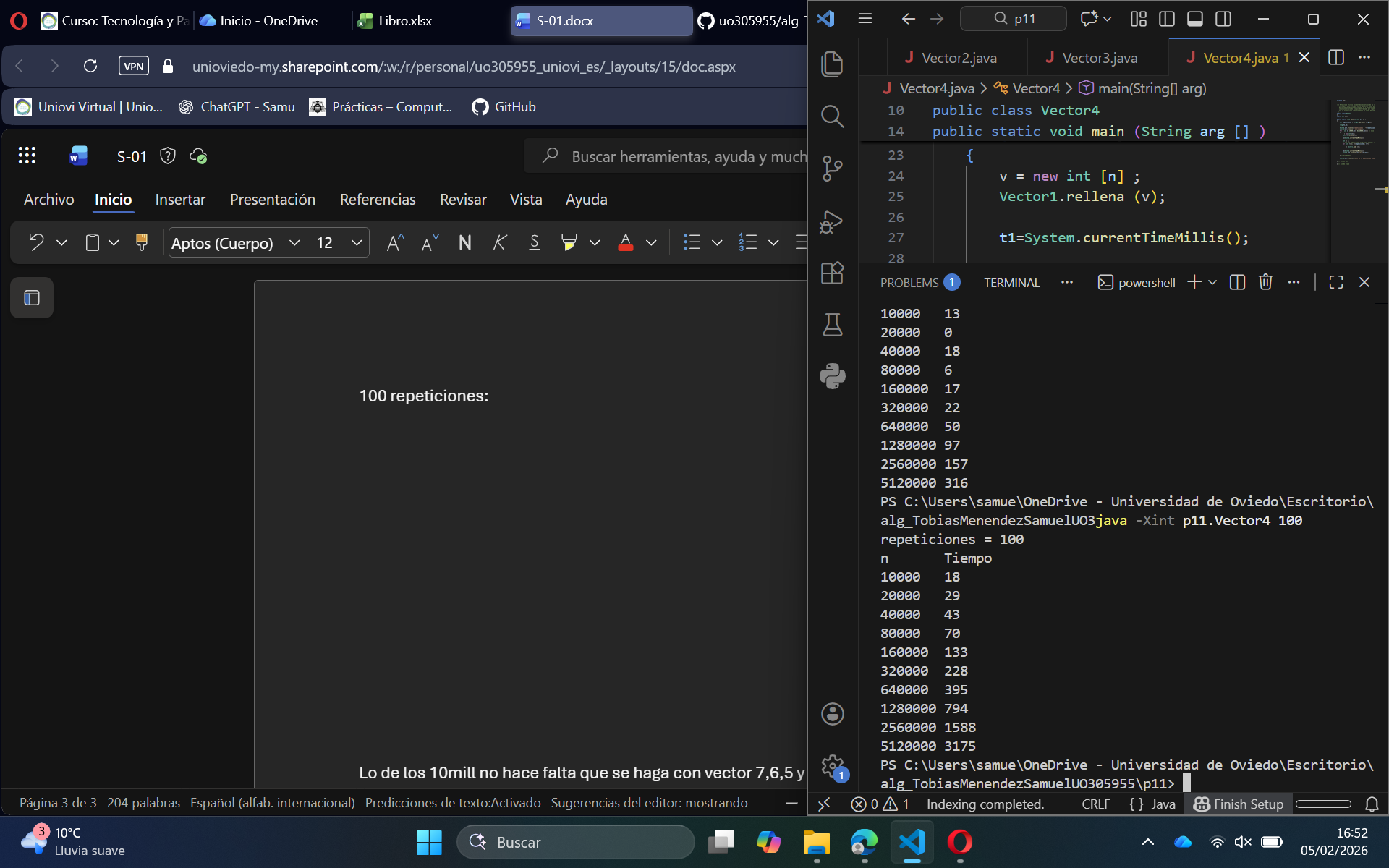
1 repetición:



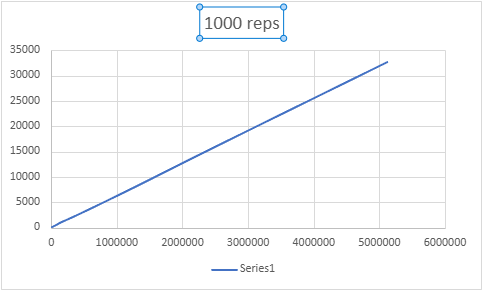
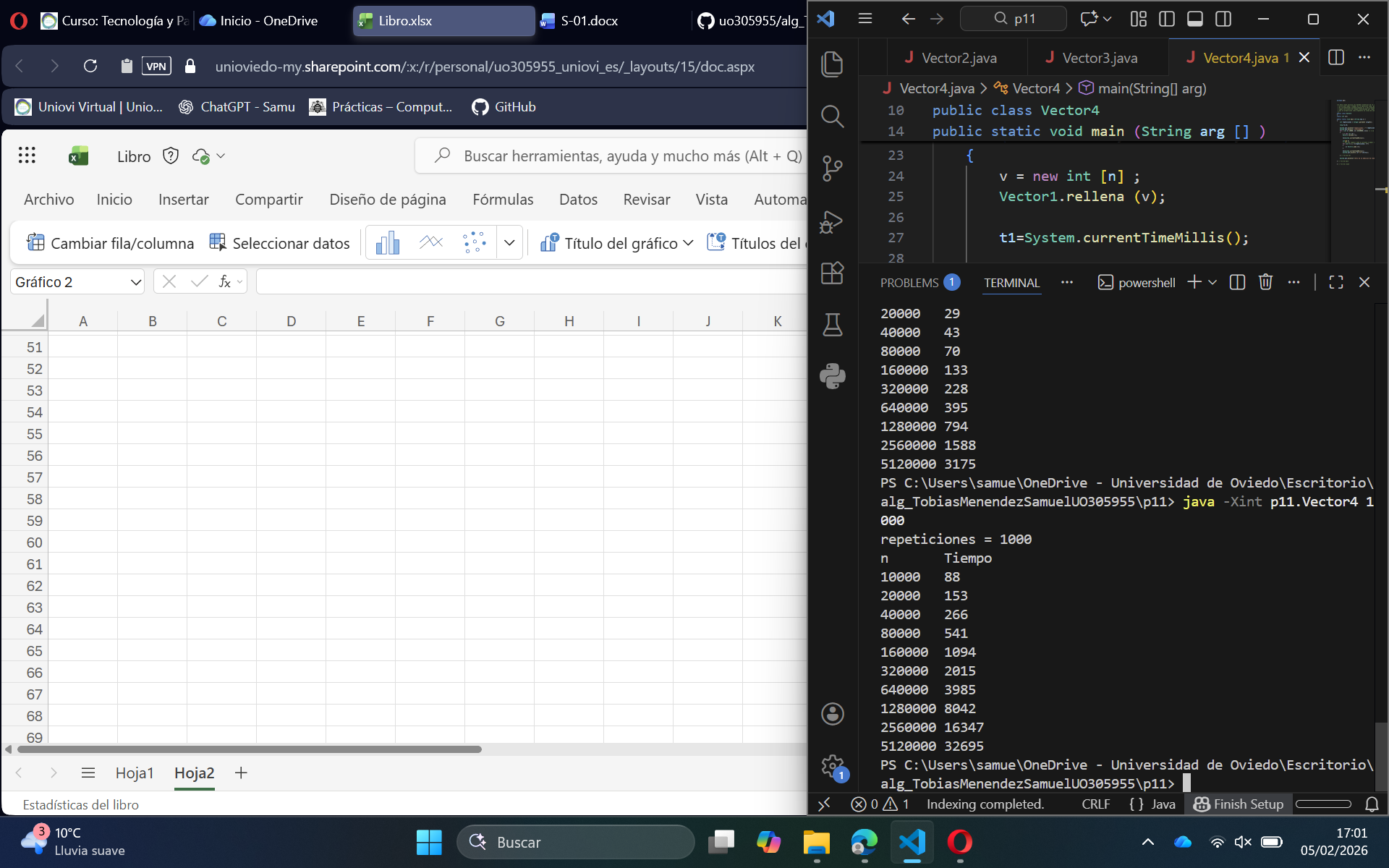
10 repeticiones:



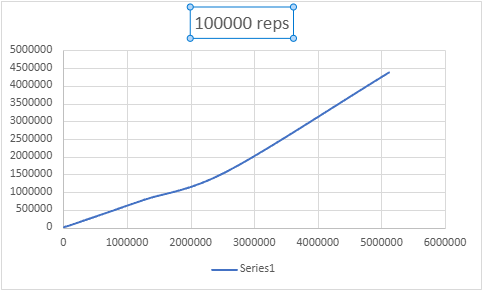
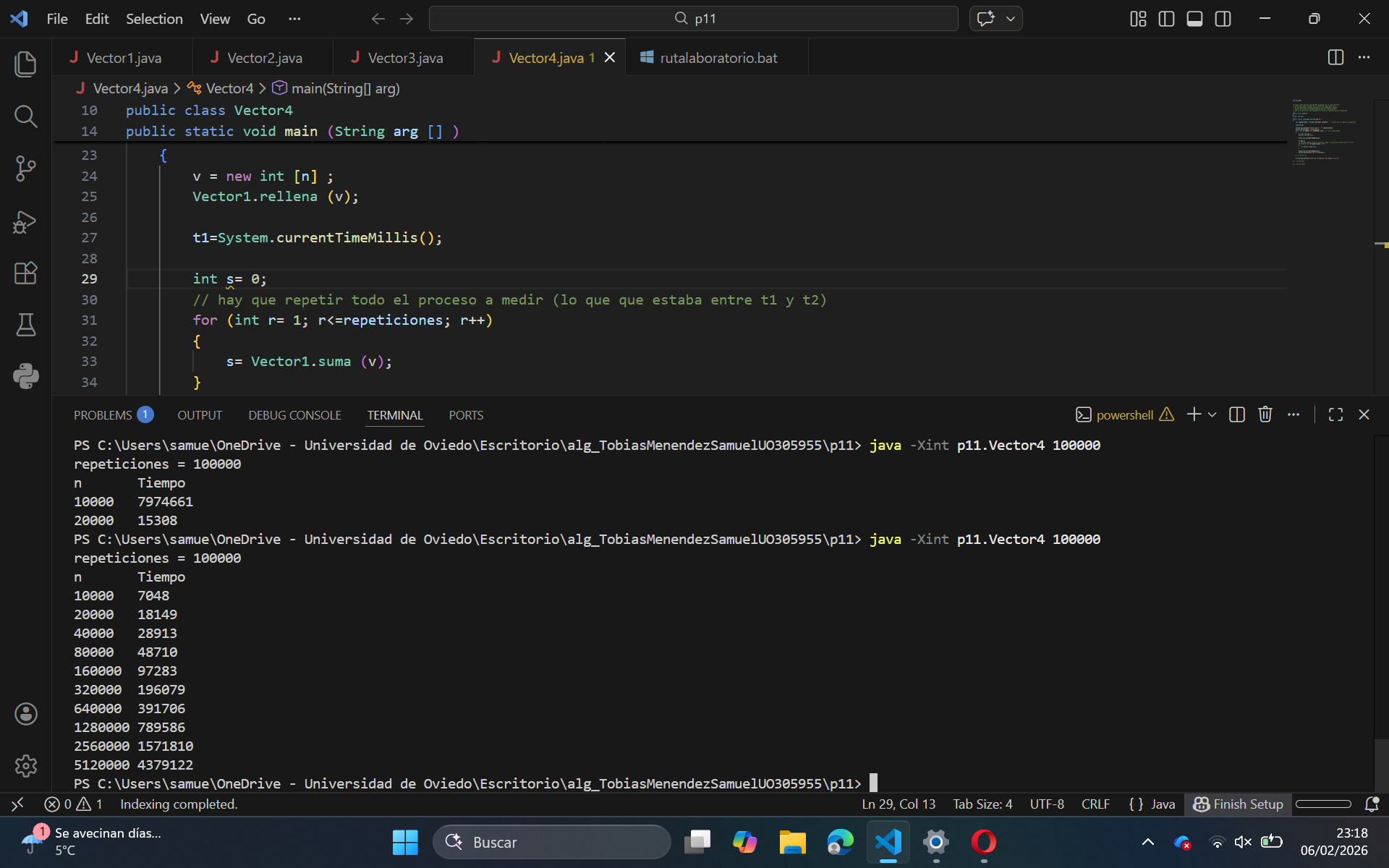
100 repeticiones:



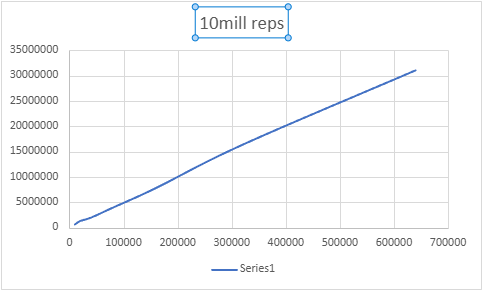
1000 repeticiones:

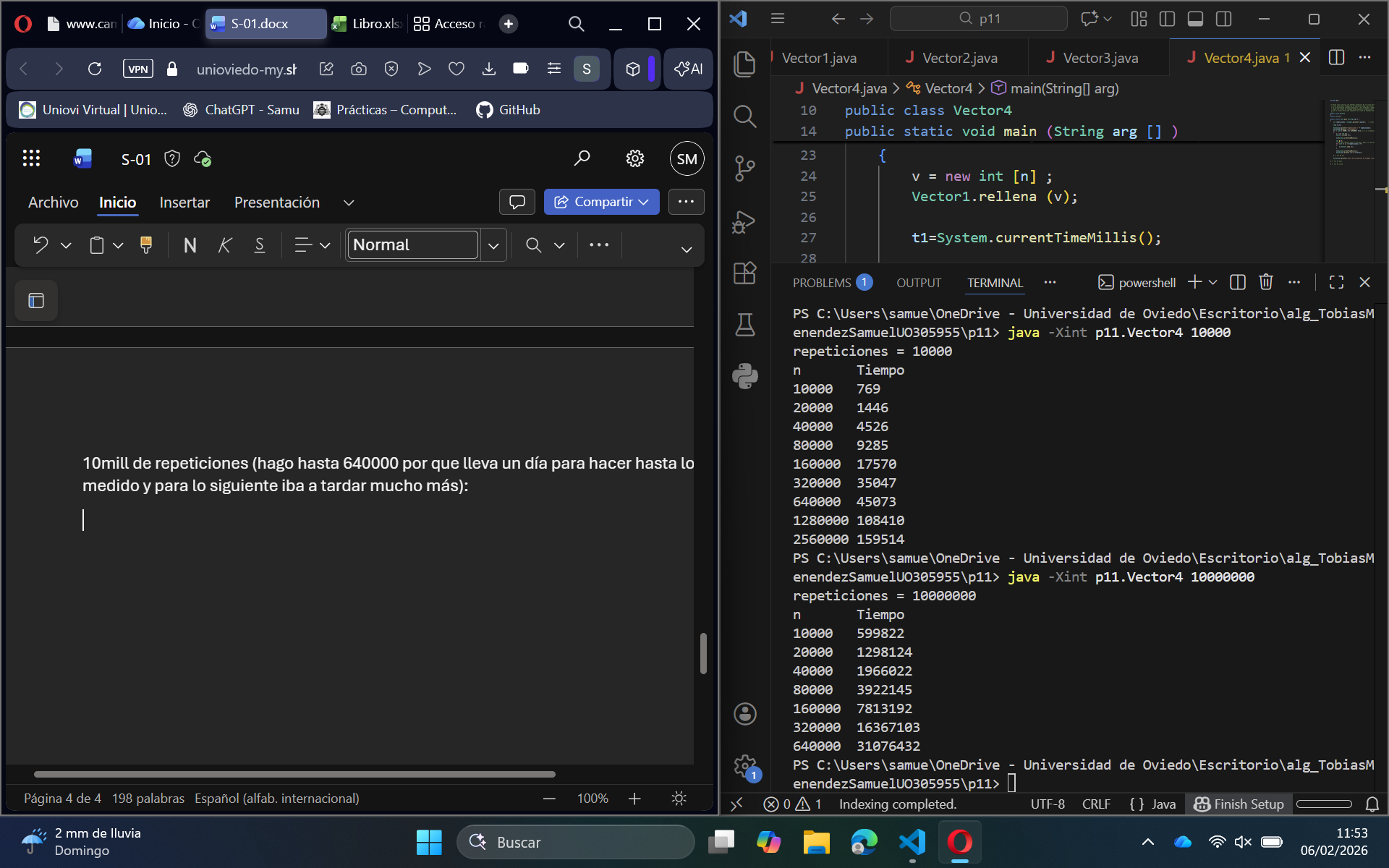


100000 repeticiones:

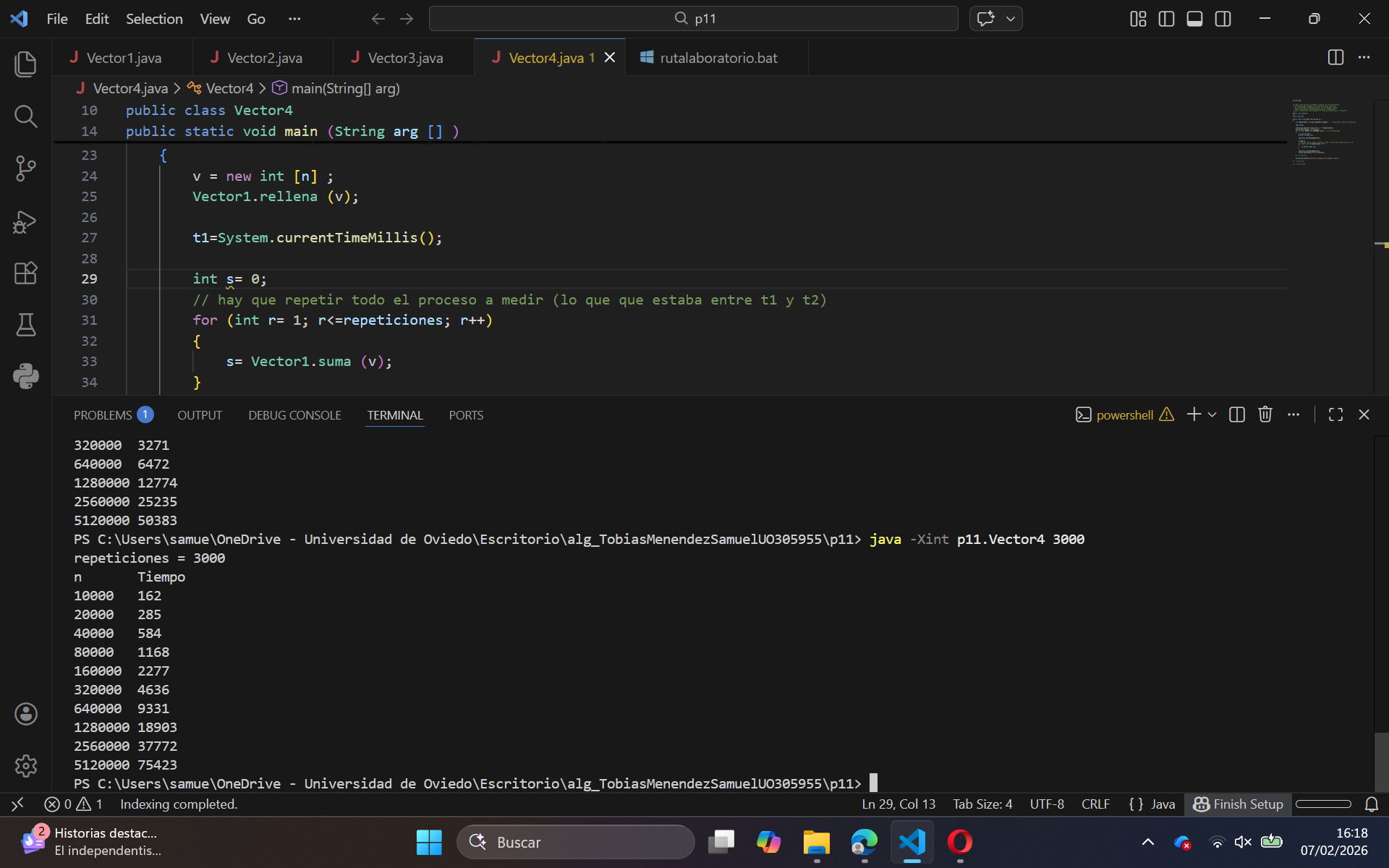


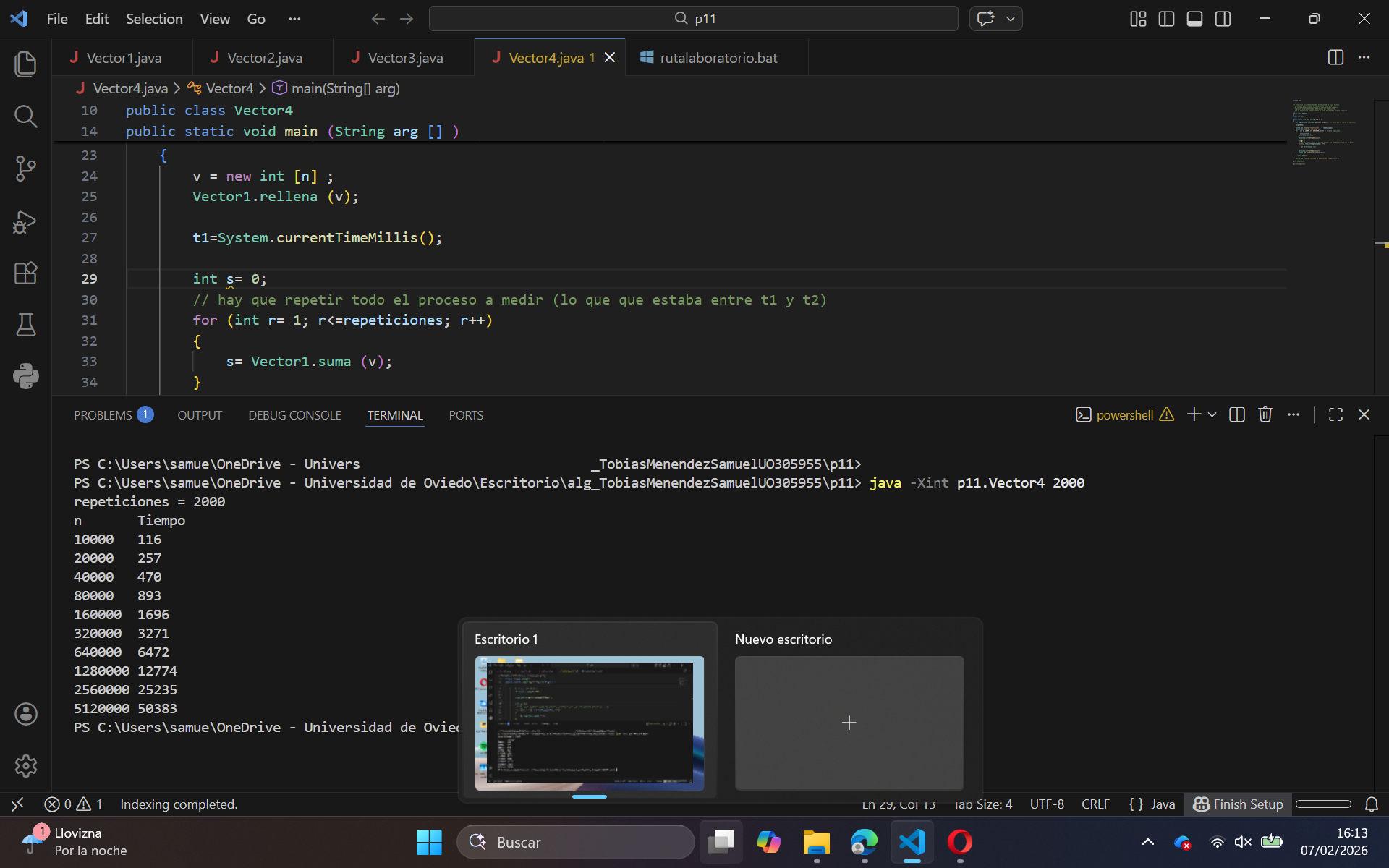
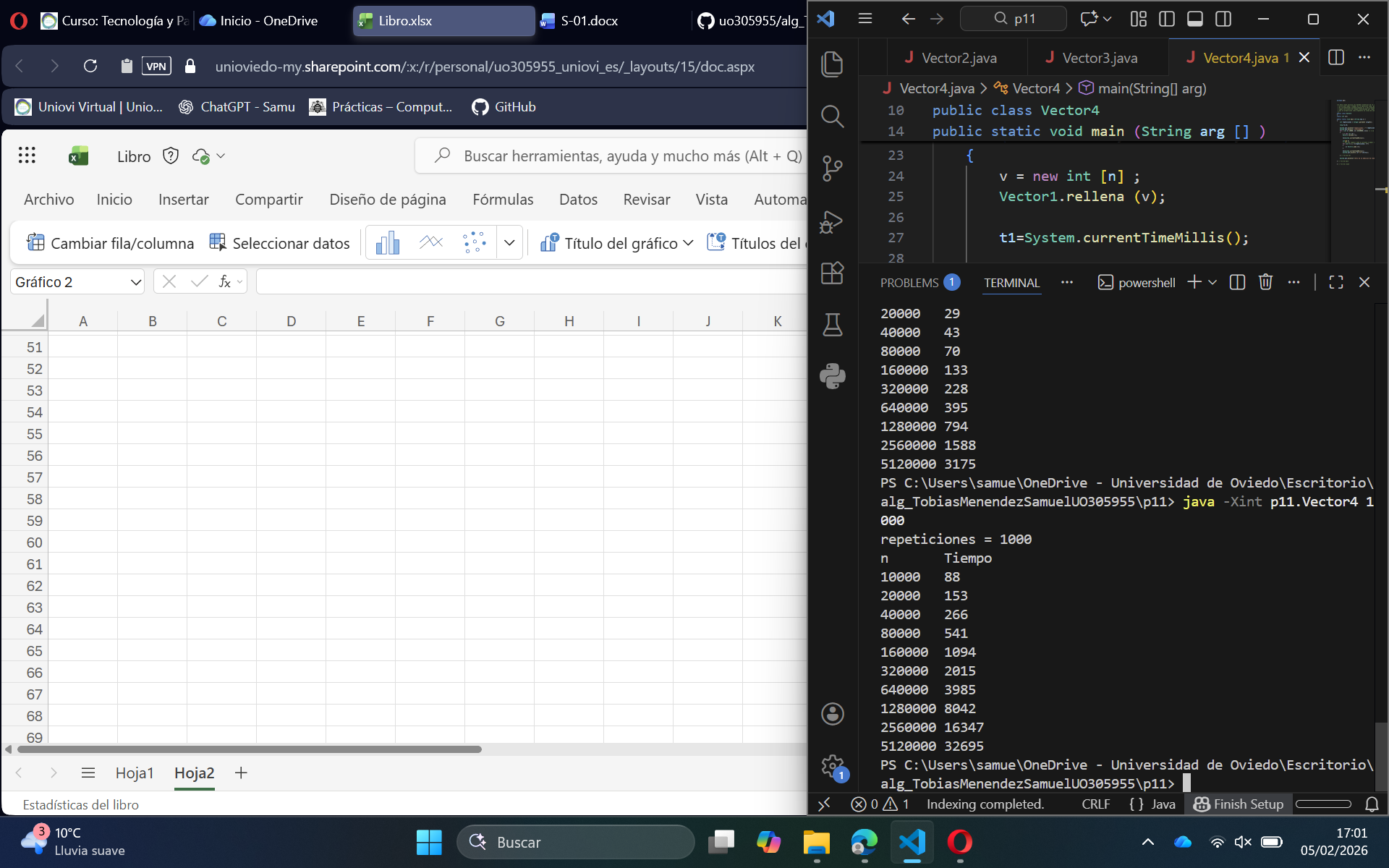
10mill de repeticiones (hago hasta 640000 por que lleva un día para hacer hasta lo medido y para lo siguiente iba a tardar mucho más):

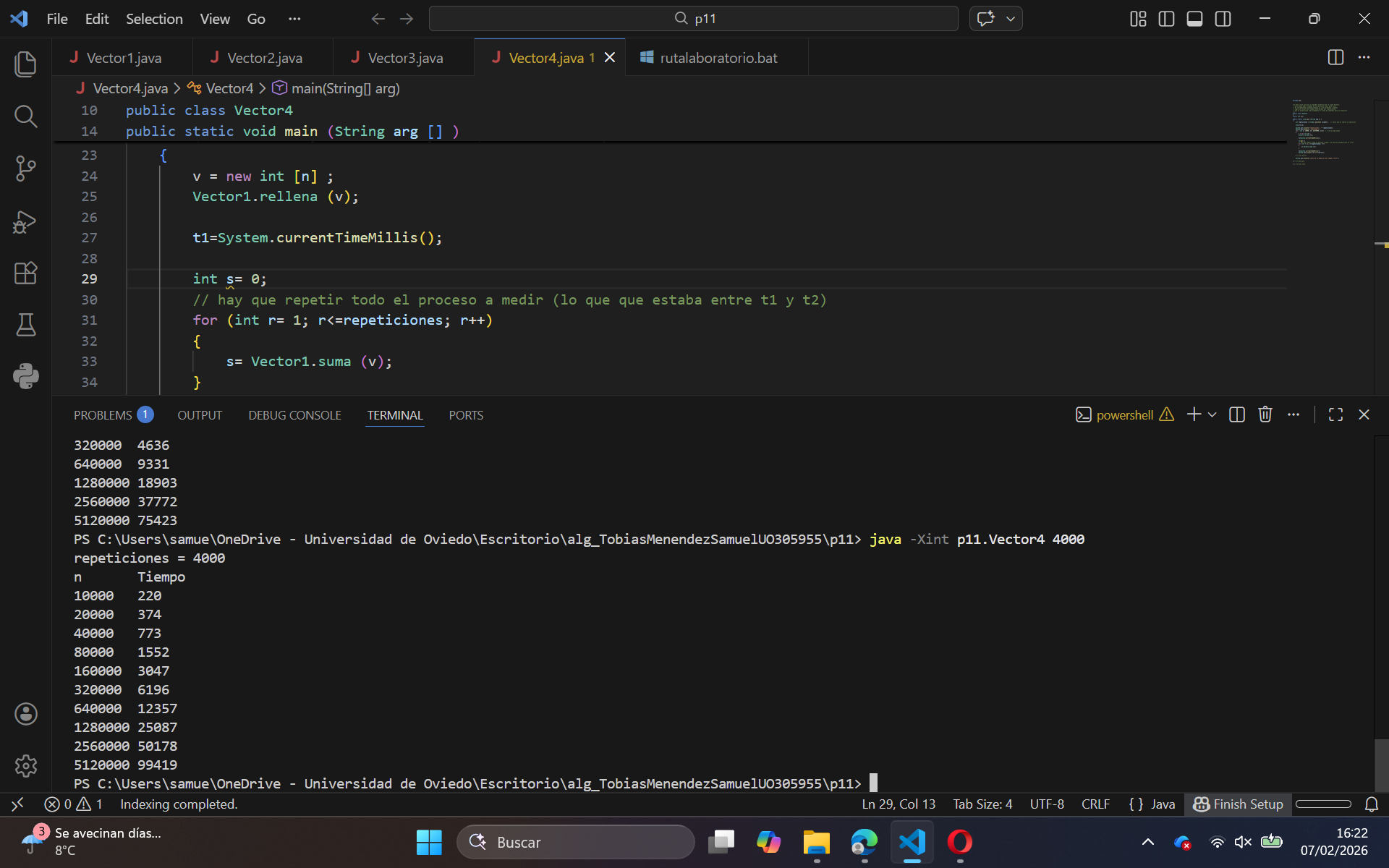




## ¿Qué pasa con el tiempo si el tamaño del problema se multiplica por 2? ¿Qué pasa con el tiempo si el tamaño del problema se multiplica por otro k que no sea 2? (Pruebe, por ejemplo, para k=3 y k=4 y compruebe los tiempos obtenidos.)

Para dar respuesta a esta pregunta voy a usar los tiempos obtenidos con 1000 repeticiones como referencia.

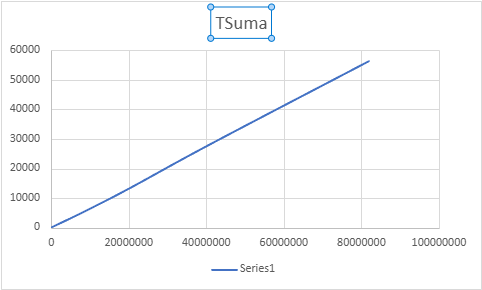


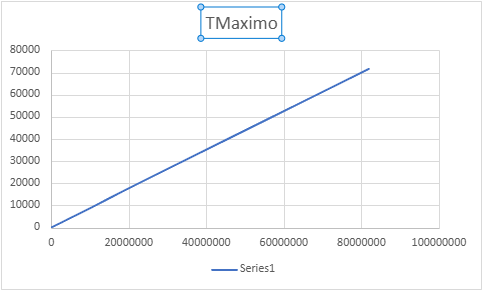


Aproximadamente, cuando se multiplican las repeticiones por una constante, el tiempo también lo hace.

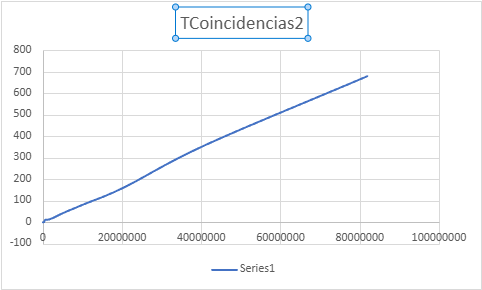
## A partir de lo visto en Vector4.java midiendo los tiempos de suma, realice las tres siguientes Clases: Vector5.java para medir los tiempos del máximo, Vector6.java para medir los tiempos de coincidencias1 y Vector7.java para medir los tiempos de coincidencias2.

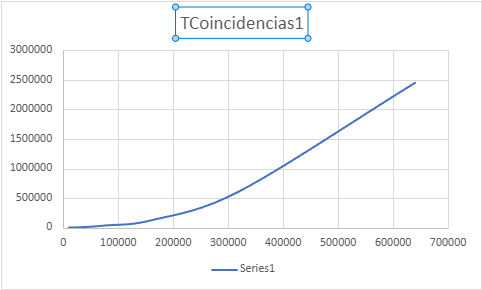
|  |  |  |
| --- | --- | --- |
| n | TSuma | TMaximo |
| 10000 | 14 | 20 |
| 20000 | 18 | 29 |
| 40000 | 33 | 49 |
| 80000 | 62 | 93 |
| 160000 | 128 | 167 |
| 320000 | 213 | 288 |
| 640000 | 405 | 553 |
| 1280000 | 817 | 1129 |
| 2560000 | 1620 | 2213 |
| 5120000 | 3216 | 4469 |
| 10240000 | 6529 | 8935 |
| 20480000 | 13489 | 18159 |
| 40960000 | 28183 | 36030 |
| 81920000 | 56353 | 71658 |





|  |  |  |
| --- | --- | --- |
| n | TCoincidencias1 | TCoincidencias2 |
| 10000 | 675 | 0 |
| 20000 | 2420 | 0 |
| 40000 | 9582 | 0 |
| 80000 | 38282 | 0 |
| 160000 | 123081 | 0 |
| 320000 | 612996 | 0 |
| 640000 | 2452972 | 12 |
| 1280000 | Fdt | 12 |
| 2560000 | Fdt | 20 |
| 5120000 | Fdt | 43 |
| 10240000 | Fdt | 83 |
| 20480000 | Fdt | 162 |
| 40960000 | Fdt | 359 |
| 81920000 | Fdt | 680 |





(Aunque la práctica especifique que más de un minuto se considera fuera de tiempo, he decidido coger más valores para que se vean mejor los datos obtenidos).

## Una vez rellenadas ambas tablas, concluya si los tiempos obtenidos sintonizan con lo esperado, según la complejidad temporal computacional de las diferentes operaciones.

Los resultados concuerdan con lo esperado. Tmáximo, TSuma, TCoincidencias2 son O(n) mientras que TCoincidencias1 es O(n^2).

## Indique las características principales (procesador, memoria) del ordenador donde se han medido tiempos.

