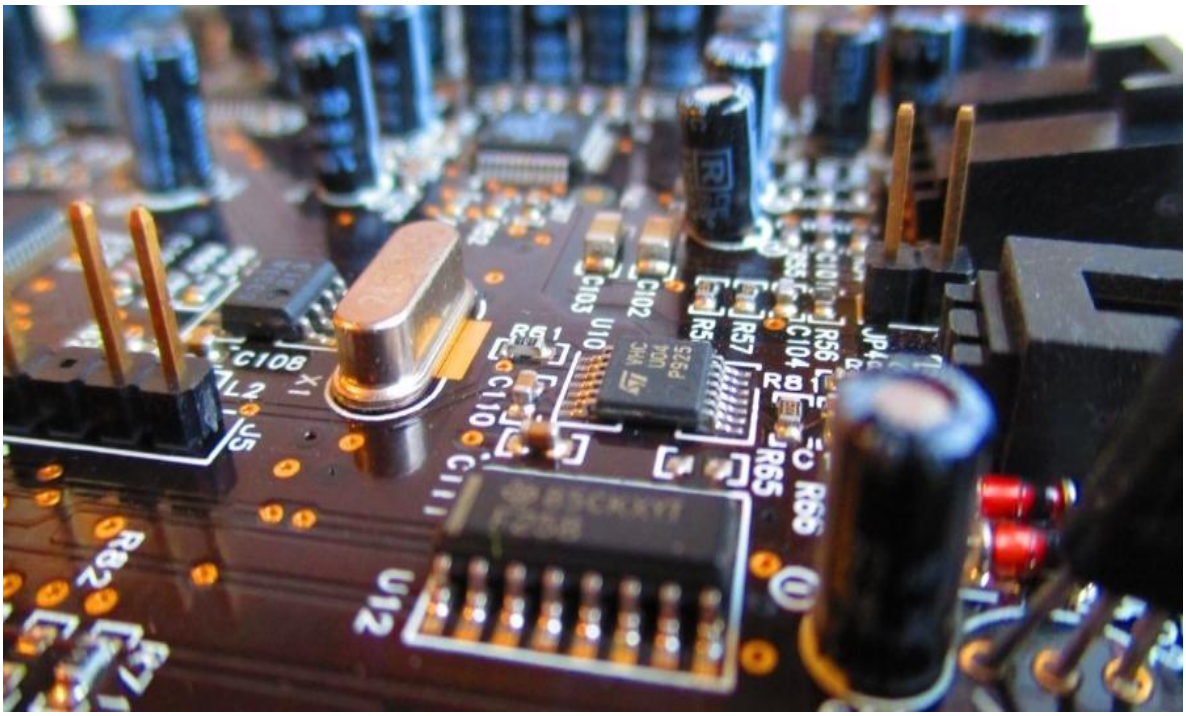
FACULTAD DE INGENIERÍA

CIUDAD UNIVERSITARIA



SISTEMAS OPERATIVOS

MAESTRO: GUNNAR EYAL WOLF ISZAEVICH

ALUMNO: YUKIOAYAX CANEK GABRIEL HERNÁNDEZ

BLOQUE: 136

GRUPO: 8

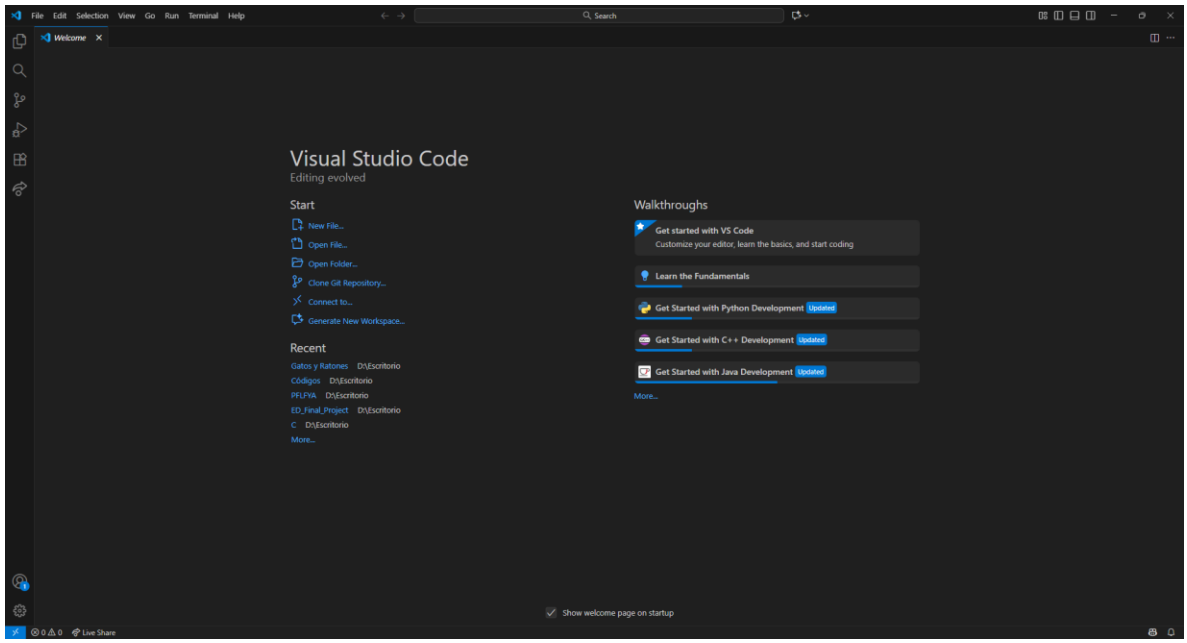
SEMESTRE 2026-1

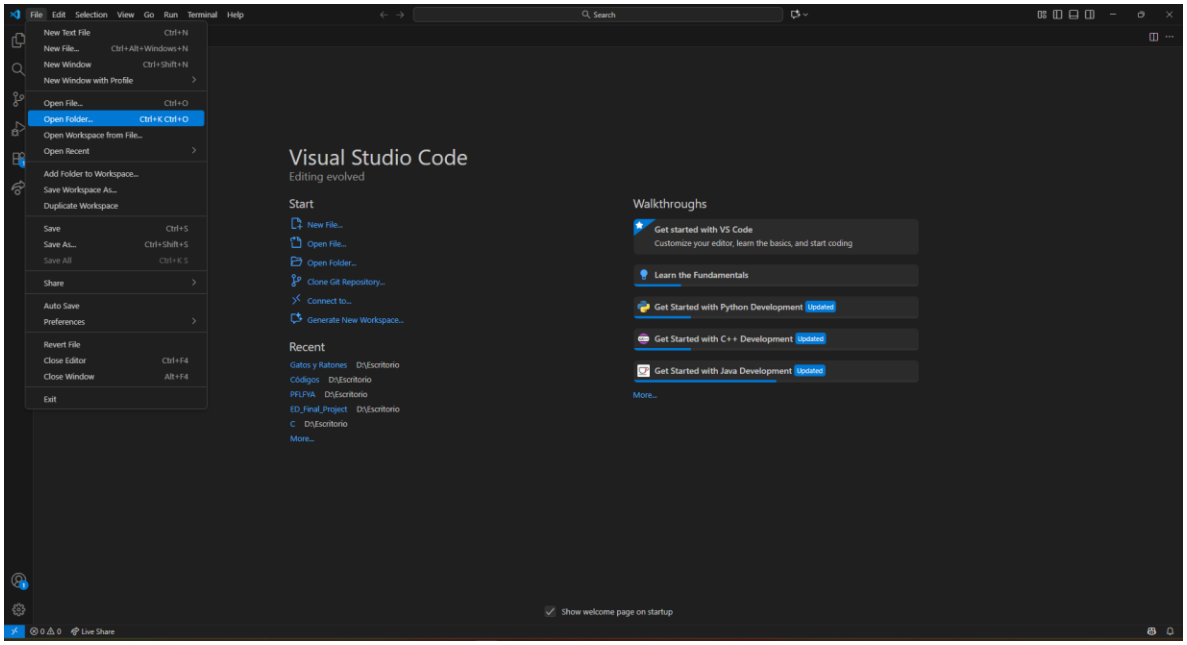
De gatos y ratones

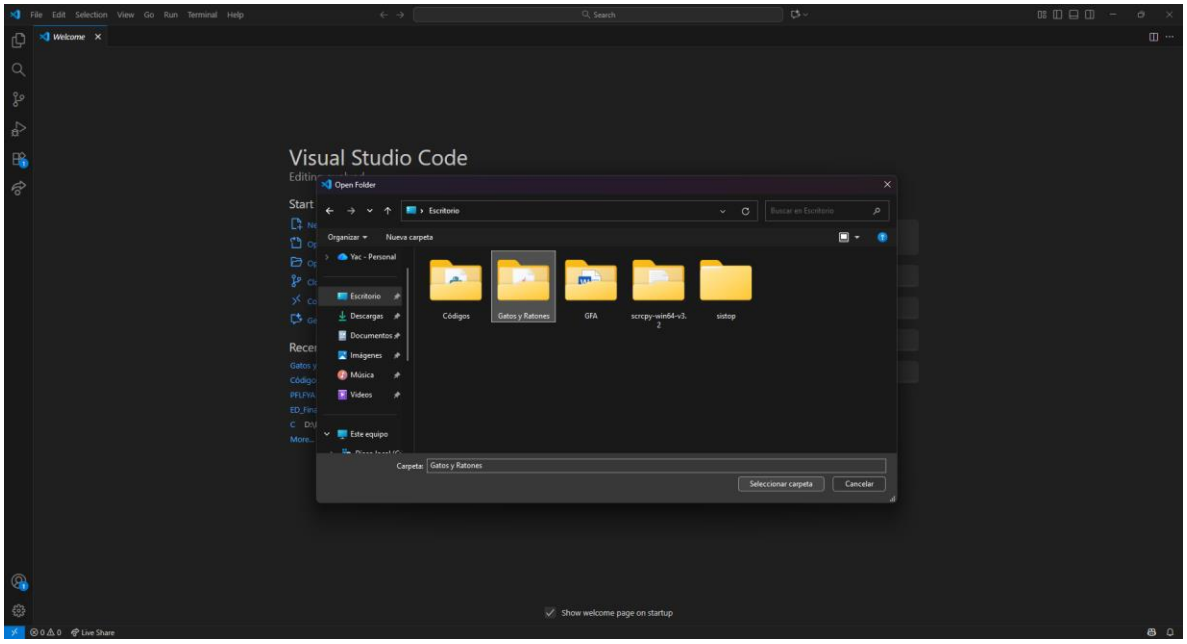
La programación concurrente permite usar varios núcleos a la vez, pero si los hilos comparten recursos sin control aparecen errores raros, para evitarlo usamos sincronización con elementos como los semáforos, locks y condiciones.
Un semáforo administra permisos, un hilo toma uno para entrar al recurso y lo devuelve al salir, con eso limitamos cuántos hilos trabajan simultáneamente y prevenimos choques; un lock da exclusión mutua, mientras un hilo tiene el lock ningún otro puede entrar a la sección crítica que protege, una condición se usa junto con lock para esperar un estado y despertar a quien deba continuar, un hilo puede esperar en la condición cuando no es su turno o no hay platos libres; al cambiar el estado otro hilo hace signal/signalAll y los que esperaban vuelven a competir por el lock, en el problema seleccionado, gatos y ratones, esto se traduce en que el lock protege el acceso a los platos y las condiciones coordinan los turnos sin inanición, la condición para que entren los gatos es que sea su turno, no haya ratones comiendo y exista al menos un plato libre mientras que la condición para que entren los ratones es análoga, ningún gato comiendo y al menos un plato libre, cuando el último del grupo que está comiendo sale y los platos quedan libres, si el otro grupo está esperando se cambia el turno y se despierta a todos los que esperaban, asegurando que ninguno quede sin oportunidad.
El problema plantea una casa con g gatos y r ratones que comparten p platos en fila, los gatos pueden comer en cualquier plato y los ratones también, pero sólo cuando no hay gatos comiendo ni mirando, si un gato ve a un ratón comiendo, el ratón corre peligro, en cada plato sólo puede haber un animal a la vez, de modo que si un gato está comiendo y un ratón empieza en otro plato, el gato lo ve, por acuerdo cuando hay ratones comiendo, los gatos se mantienen alejados, debe evitarse la inanición, ósea, ni gatos ni ratones pueden quedar indefinidamente sin comer, de estas reglas se sigue que nunca comen gatos y ratones al mismo tiempo, varios animales de la misma especie sí pueden comer en paralelo mientras existan platos libres y cuando ambos grupos esperan deben alternarse para que nadie monopolice los platos.
Se implementó un código en Java para resolver el ejercicio, para probarlo primero se necesita Java instalado, para esto se recomienda visitar el siguiente enlace (https://www.youtube.com/watch?v=Pzl41Iw4T04), además, para usar un editor cómodo, se recomienda instalar Visual Studio Code (https://www.youtube.com/watch?v=6pD7_rcFrj8), si no se desea instalar este editor, también se puede correr todo desde la terminal.
Existen 2 formas de correr el programa:

a) Visual Studio Code: Abre Visual Studio Code, si es la primera vez verás una ventana en blanco, si ya lo usaste, puede que se abra con la última carpeta que tenías, una vez que se esté situado aquí hay dos opciones: se puede arrastrar la carpeta del proyecto a la ventana o abrir la carpeta desde Archivo → Abrir carpeta…, después concede los permisos de confianza cuando VS Code te pregunte pues si no los concedes no podrás compilar ni ejecutar, con la carpeta abierta ubica el archivo GatosyRatones.java. Puedes correrlo con el botón de Run que aparece en el editor.

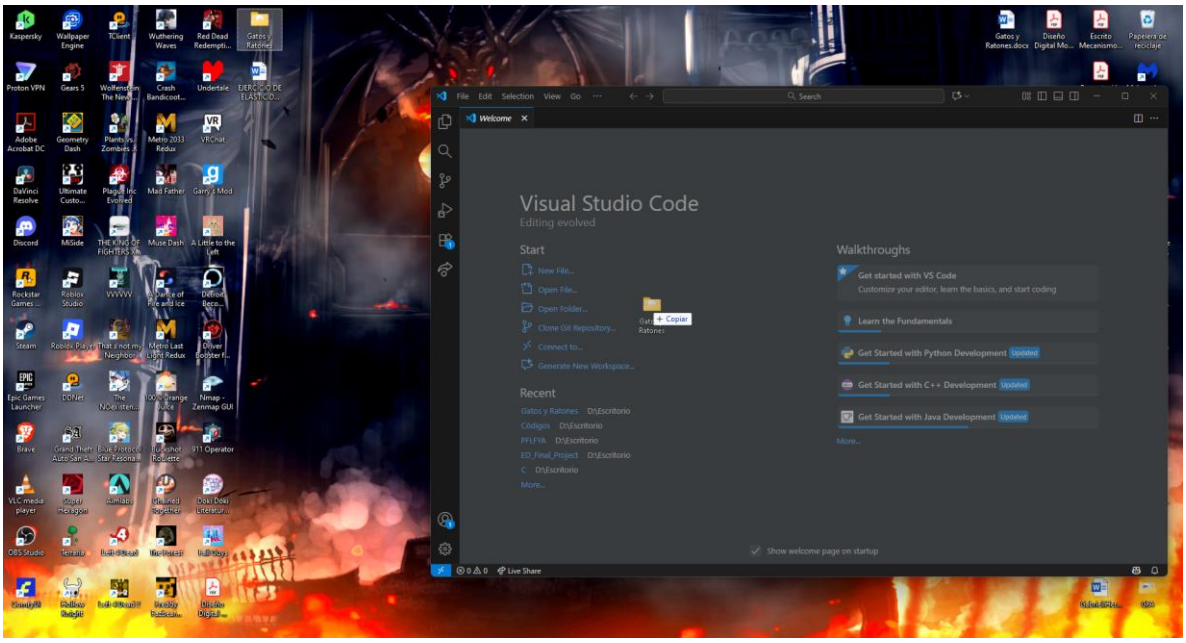
Ruta 1:


File Edit Selection View Go Run Terminal Help
Search
Welcome
Visual Studio Code
Editing evolved
Start
New File...
Open File...
Open Folder...
Clone Git Repository...
Connect to...
Generate New Workspace...
Recent
Gatos y Ratones D:\Escritorio
Códigos D:\Escritorio
PFLFYA D:\Escritorio
ED_Final_Project D:\Escritorio
C D:\Escritorio
More...
Walkthroughs
Get started with VS Code
Customize your editor, learn the basics, and start coding
Learn the Fundamentals
Get Started with Python Development Updated
Get Started with C++ Development Updated
Get Started with Java Development Updated
More...
Show welcome page on startup
Live Share

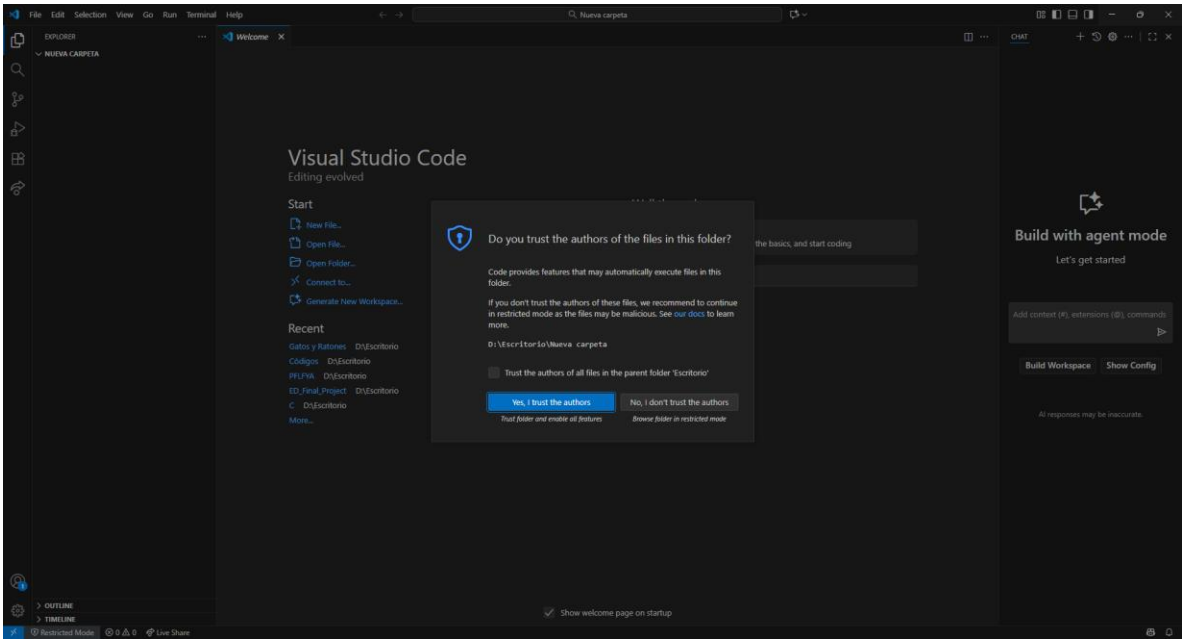

File Edit Selection View Go Run Terminal Help
Search
New Text File Ctrl+N
New File... Ctrl+Alt+Windows+N
New Window Ctrl+Shift+N
New Window with Profile
Open File... Ctrl+O
Open Folder... Ctrl+K Ctrl+O
Open Workspace from File...
Open Recent
Add Folder to Workspace...
Save Workspace As...
Duplicate Workspace
Save Ctrl+S
Save As... Ctrl+Shift+S
Save All Ctrl+K S
Share
Auto Save
Preferences
Revert File
Close Editor Ctrl+F4
Close Window Alt+F4
Exit
Visual Studio Code
Editing evolved
Start
New File...
Open File...
Open Folder...
Clone Git Repository...
Connect to...
Generate New Workspace...
Recent
Gatos y Ratones D:\Escritorio
Códigos D:\Escritorio
PFLFYA D:\Escritorio
ED_Final_Project D:\Escritorio
C D:\Escritorio
More...
Walkthroughs
Get started with VS Code
Customize your editor, learn the basics, and start coding
Learn the Fundamentals
Get Started with Python Development Updated
Get Started with C++ Development Updated
Get Started with Java Development Updated
More...
Show welcome page on startup
Live Share


Visual Studio Code
Open Folder
Escritorio
Organizer
Nueva carpeta
Códigos
Gatos y Ratones
GFA
sistop
Carpeta:
Gatos y Ratones
Seleccionar carpeta
Cancelar
Show welcome page on startup

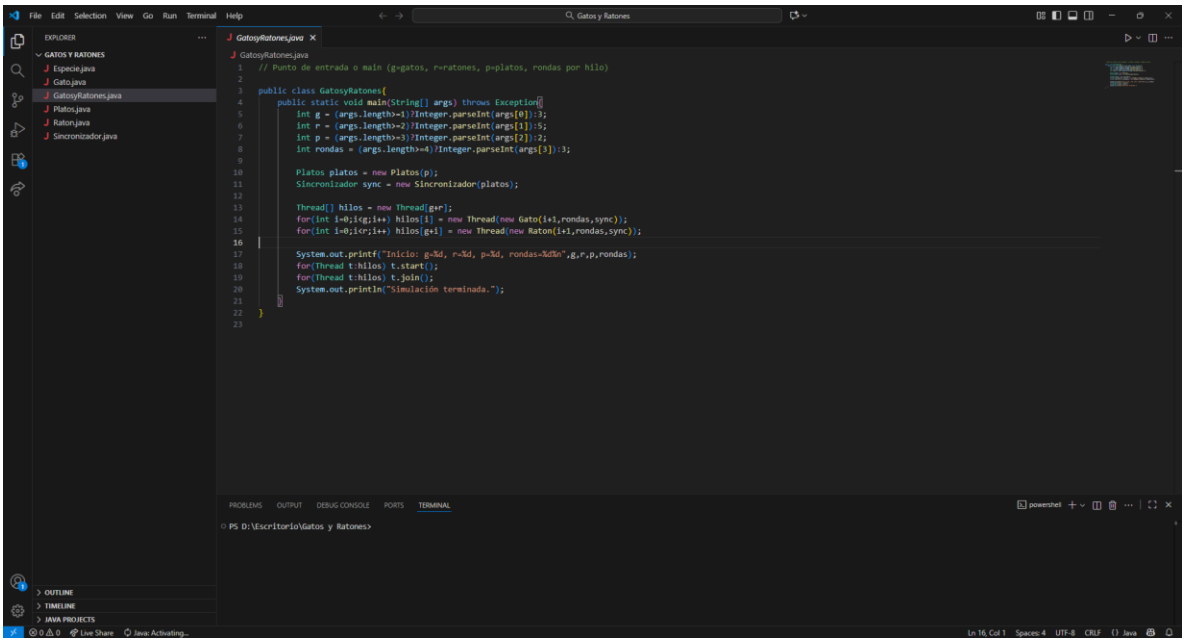
Ruta 2:


Visual Studio Code
Editing evolved
Start
New File...
Open File...
Open Folder...
Clone Git Repository...
Connect to...
Generate New Workspace...
Recent
Walkthroughs
Get started with VS Code
Learn the Fundamentals
Get Started with Python Development
Get Started with C++ Development
Get Started with Java Development
Show welcome page on startup

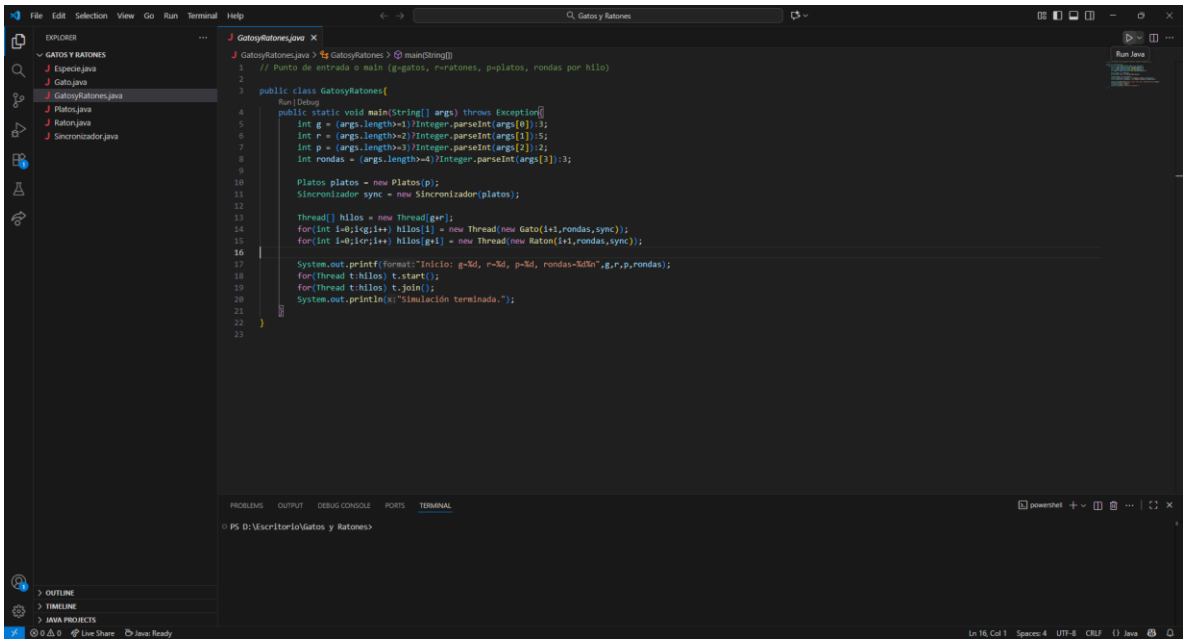
En ambas rutas aparecerá la siguiente ventana:



Pulsa el botón “Yes, I trust the authors” y sitúate en el archivo “GatosyRatones.java”.



Finalmente, presiona el botón de compilar:

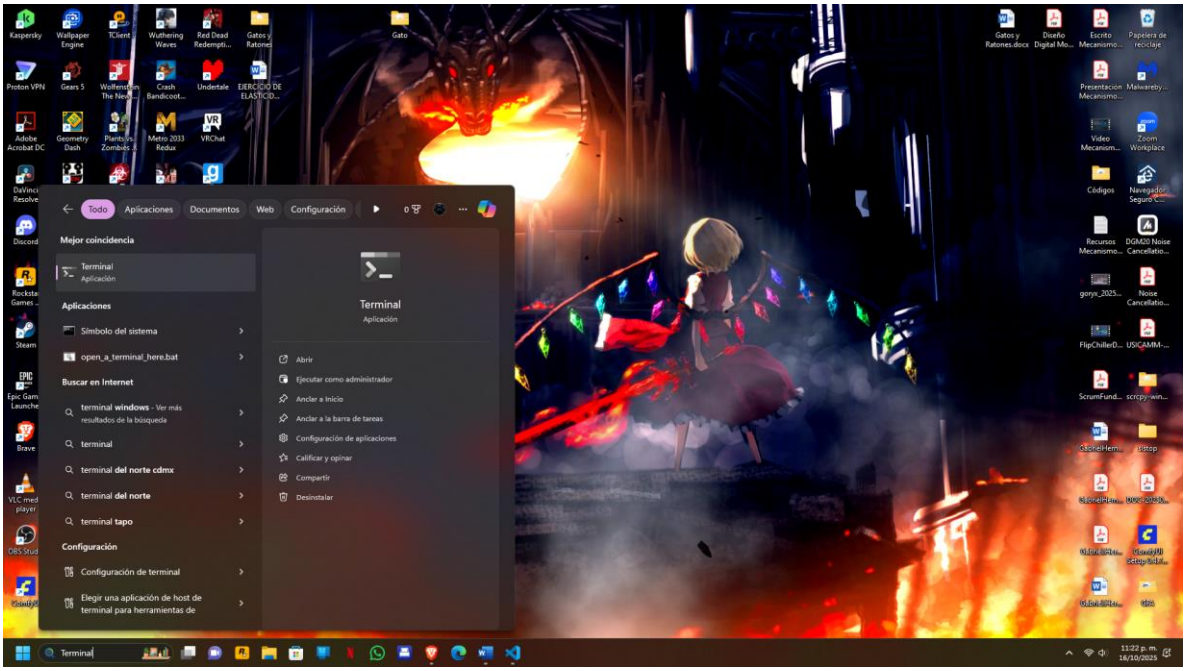


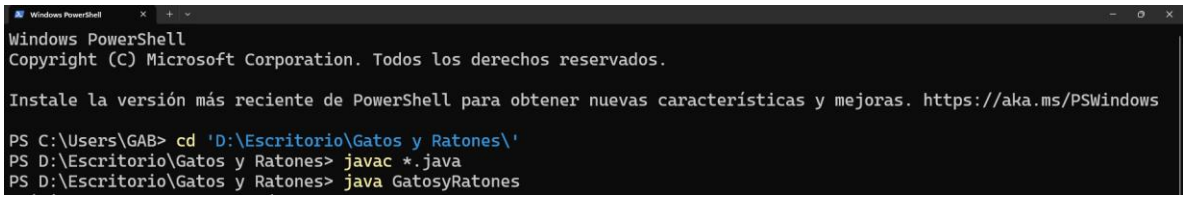
b) Terminal: Sitúate en la carpeta donde están los archivos .java (usa cd hasta esa carpeta), compila todo:

javac *.java

java GatosyRatones g r p rondas

Si no pones parámetros, se usa por defecto: g=3, r=5, p=2, rondas=3.





Listo, con eso corre y verás en consola las líneas de inicio, quién come en qué plato, y al final “Simulación terminada”.

La Opción B (Terminal) es la recomendada si se quiere probar valores distintos a los de defecto porque puedes pasar g r p rondas directamente al ejecutar, en otros sistemas operativos o distribuciones los pasos son similares.

La solución planteada se basa en dos ideas, turno y capacidad, con un lock y dos condiciones (gatos y ratones), un hilo entra sólo si es su turno, no hay del otro grupo comiendo y hay un plato libre, los p platos se cuentan como recursos pues si no hay, se espera, cuando alguien libera entra otro del mismo turno, cuando sale el último del grupo activo, si el otro grupo está esperando se cambia el turno y se despiertan sus hilos, si no, sigue el mismo grupo. Con esto nunca comen ambas especies al mismo tiempo, no se usan más platos de los que hay y nadie se queda sin pasar.

En cuanto al código, se crearon diversas clases:

- Especie.java: Etiquetas GATOS, RATONES y NADIE para indicar de forma simple quién tiene el turno o si no hay nadie, sirve para que el estado sea claro y legible.
- Platos.java: Administra los p platos con una cola de capacidad fija, reservar toma un id de plato libre, devolver lo regresa; Garantiza que nunca haya más comensales que platos.
- Sincronizador.java: Es el corazón de la coordinación, usa un lock y dos condicionales (gatos y ratones), valida tres cosas antes de entrar, que sea el turno de la especie, que no haya del otro grupo comiendo y que exista un plato libre; mantiene contadores (comiendo/esperando), alterna el turno

cuando ambos esperan y despierta al grupo que sigue, evita mezcla de especies y evita inanición.

- Gato.java: El hilo gato hace rondas, entra cuando puede, toma plato, come un momento, libera el plato y descansa, imprime mensajes breves para ver el flujo.
- Raton.java: El hilo ratón tiene el mismo ciclo que Gato (entrar–comer–salir), también con mensajes breves.
- GatosyRatones.java: Es el main o ejecutable, lee parámetros (g, r, p, rondas), crea y arranca los hilos, espera a que terminen (join) y muestra la configuración inicial y el cierre de la simulación.

En conclusión, la solución cumple el planteamiento, nunca comen gatos y ratones al mismo tiempo, varios del mismo tipo pueden comer en paralelo hasta agotar los platos y nadie se queda sin turno gracias a la alternancia, la ejecución es simple (VS Code o terminal) y la salida en consola muestra con claridad quién entra, qué plato usa y cuándo libera.

Bibliografía

S.A. (s.f.). JDK 25 Documentation. ORACLE. Recuperado el 13 de octubre de 2025 de https://docs.oracle.com/en/java/javase/25/

Mestras, J. P. (2001*). Programación Concurrente con Java.* Dep. Sistemas Informáticos y Programación. Universidad Complutense Madrid. Recuperado de https://grasia.fdi.ucm.es/jpavon/docencia/dso/programacionconcurrentejava.pdf