|  |  |  |
| --- | --- | --- |
|  | **Carátula para entrega de prácticas** | |
| Facultad de Ingeniería | | Laboratorio de docencia |

Laboratorios de computación

salas A y B

|  |  |
| --- | --- |
| *Profesor:* | Marco Antonio Martínez Quintana |
| *Asignatura:* | Fundamentos de Programación |
| *Grupo:* | 3 |
| *No de Práctica(s):* | 10 |
| *Integrante(s):* | Camacho Garduño Miguel Angel |
| *No. de Equipo de cómputo empleado:* | No aplica |
| *No. de Lista o Brigada:* | 4 |
| *Semestre:* | 2021-1 |
| *Fecha de entrega:* | 13 de diciembre de 2020 |
| *Observaciones:* |  |
|  |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_

Guía práctica de estudio 10:

Depuración de programas

# Objetivo:

Aprender las técnicas básicas de depuración de programas en C para revisar de manera precisa en flujo de ejecución de un programa y el valor de las variables; en su cado, corregir posibles errores

# Actividades:

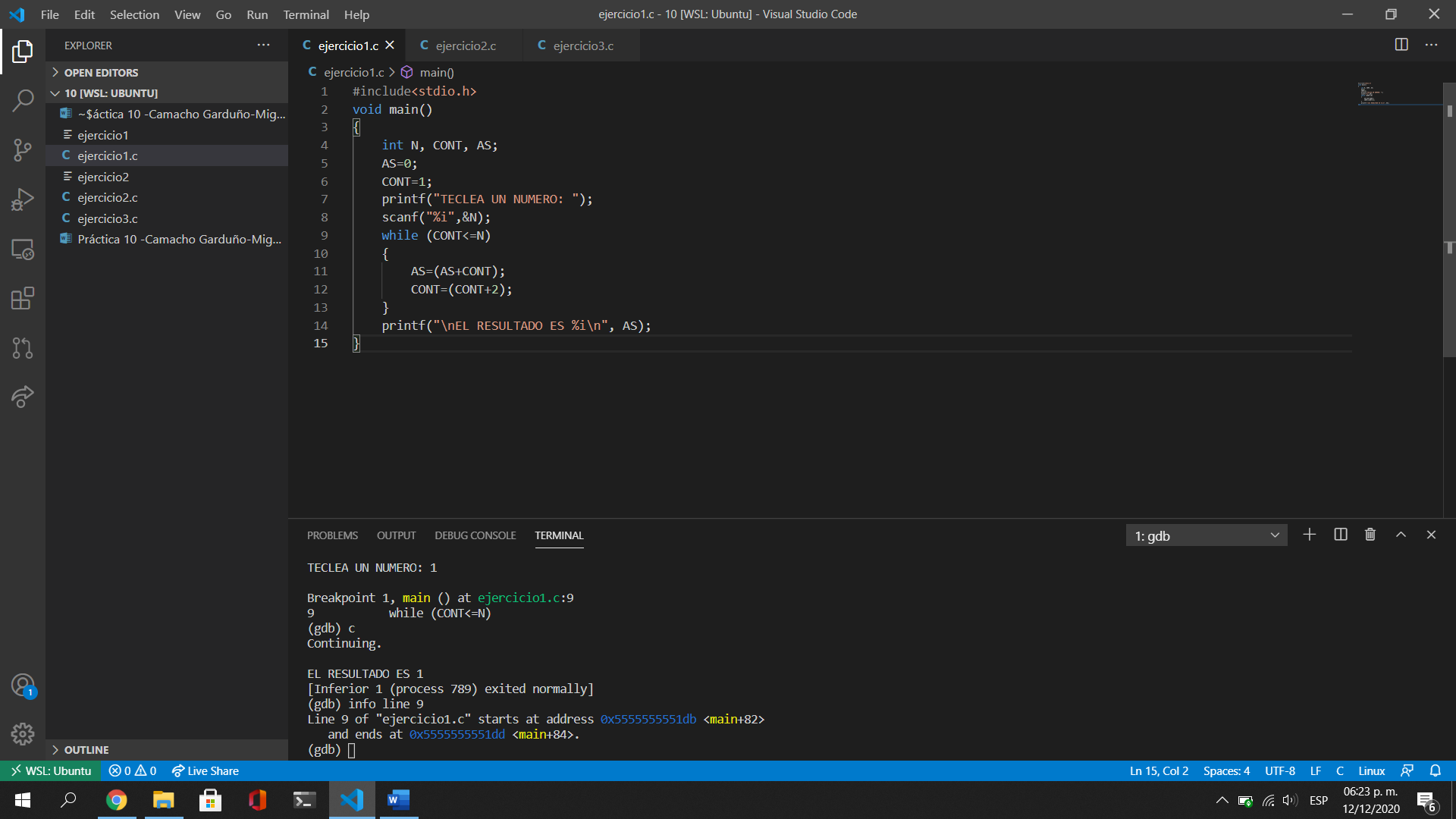
* Revisar, a través de un depurador, los valores que va tomando una variable en un programa escrito en C, al momento de ejecutarse
* Utilizando un depurador, revisar el flujo de instrucciones que se están ejecutando en un programa en C, cuando el flujo depende de los datos de entrada

# Introducción:

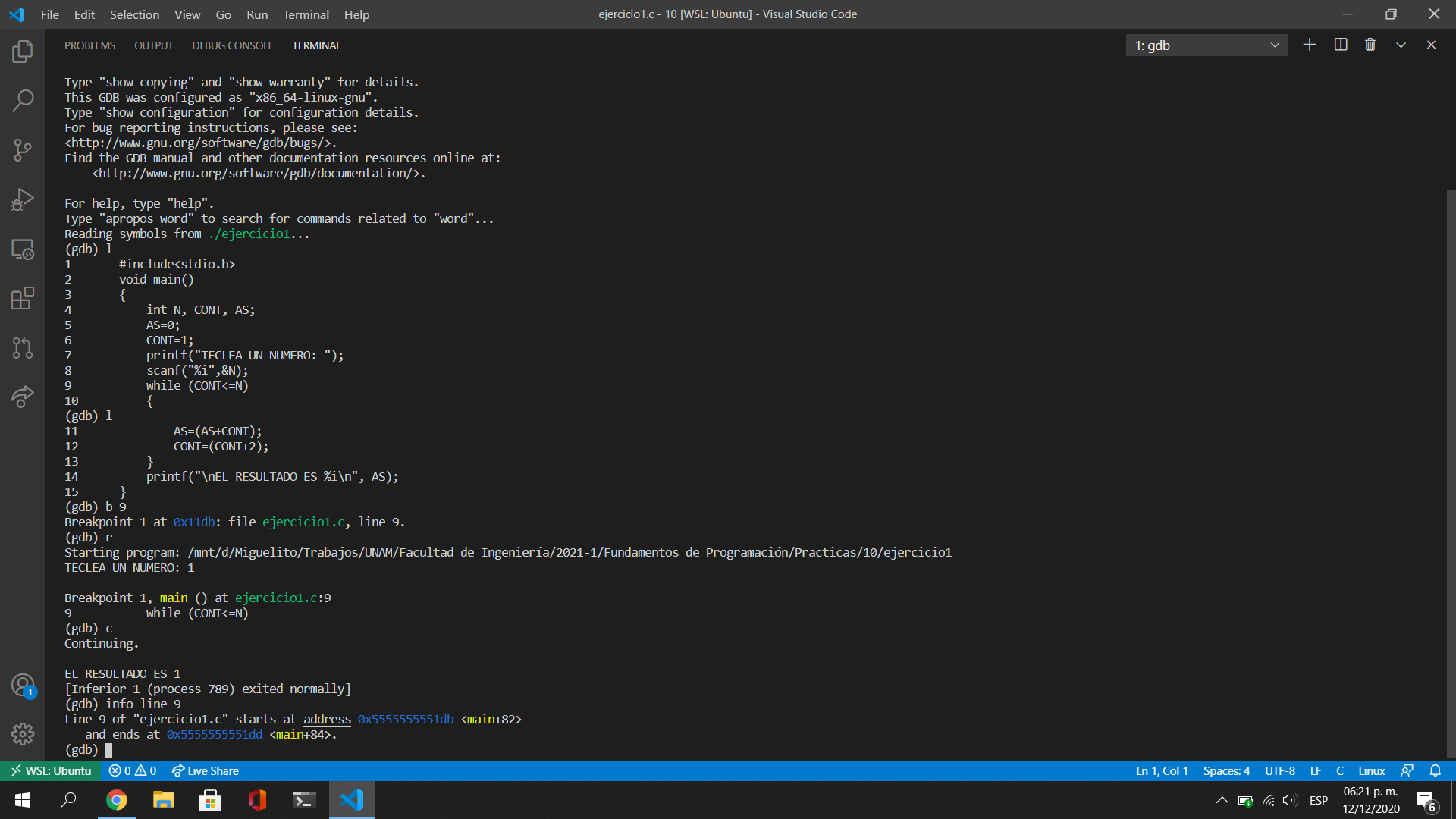
Depurar un programa significa someterlo a un ambiente de ejecución controlado por medio de herramientas dedicadas a ello. Este ambiente sirve para conocer el flujo de ejecución del programa, el valor que las variables adquieren, la pila de llamadas a funciones entre otros aspectos.

# Actividades

## Ejercicio 1

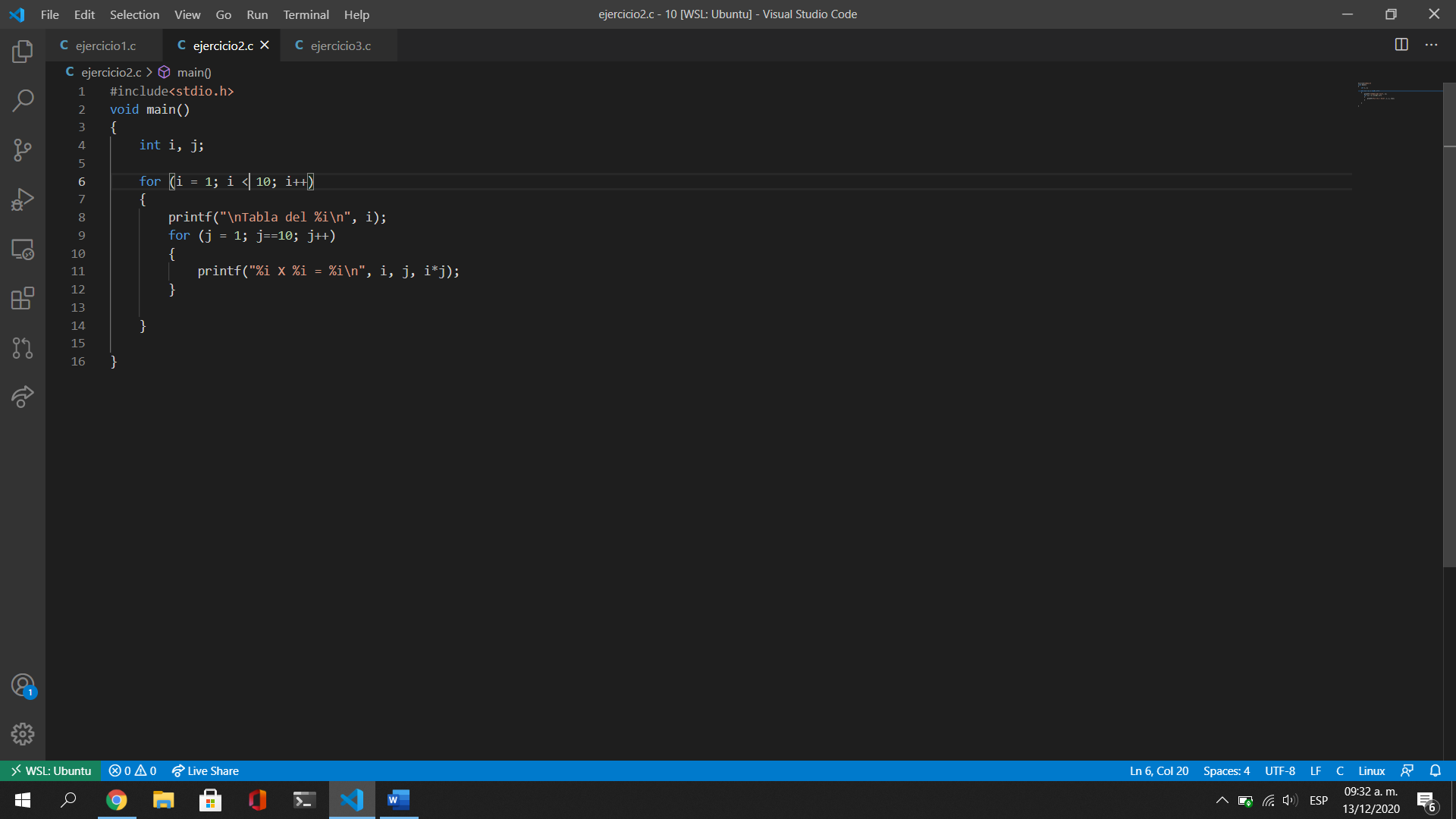


Transcribí el primer ejercicio en “ejercicio1.c” dentro de una terminal remota de Ubuntu en VSCode

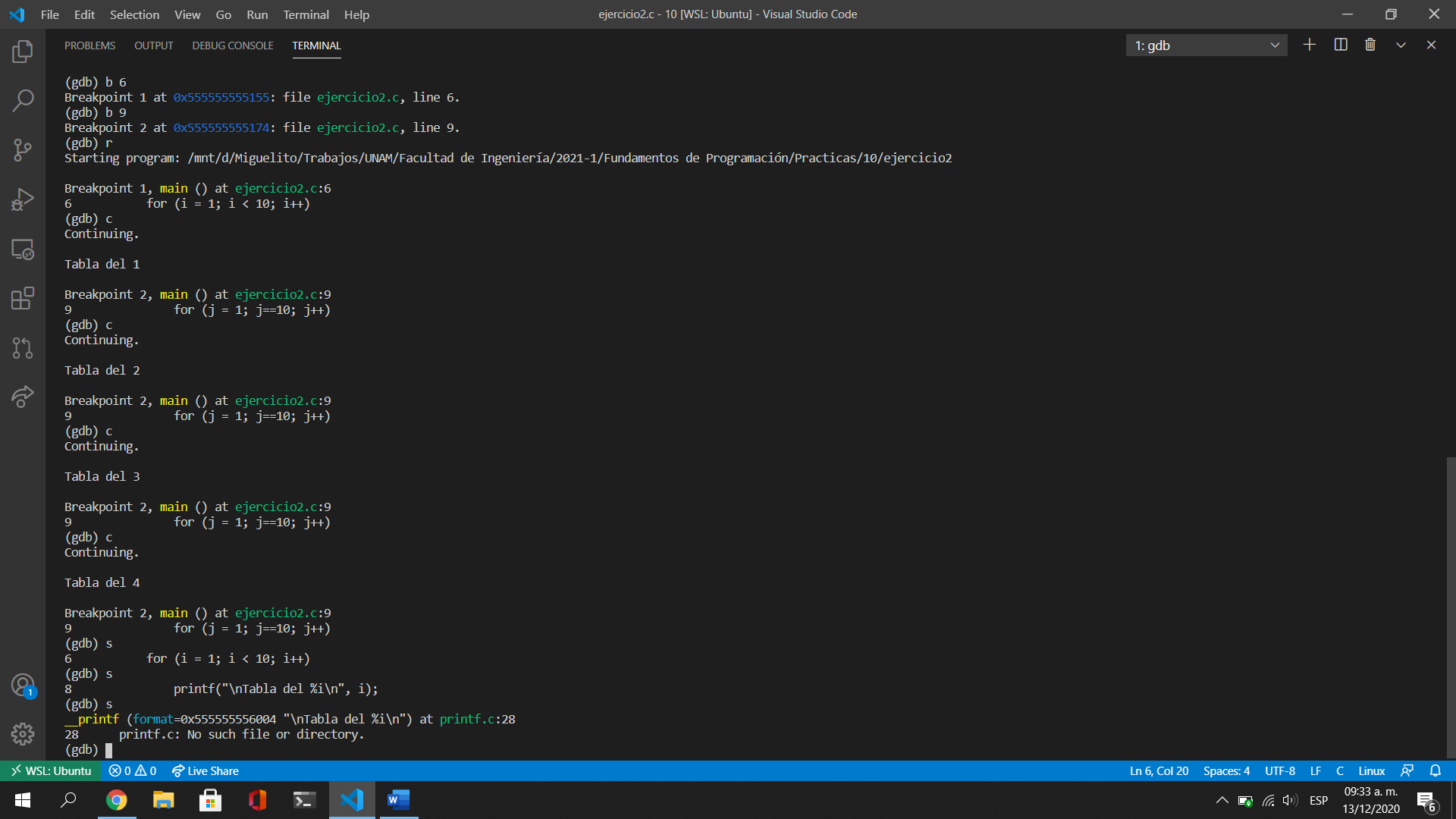


Usando gdb, agregué un punto de ruptura en la línea 9 que corresponde a “while” y al ejecutar el programa dentro de gdb, me dice que al un punto de ruptura en la línea 9 y al darle “c” continúa la ejecución del programa.

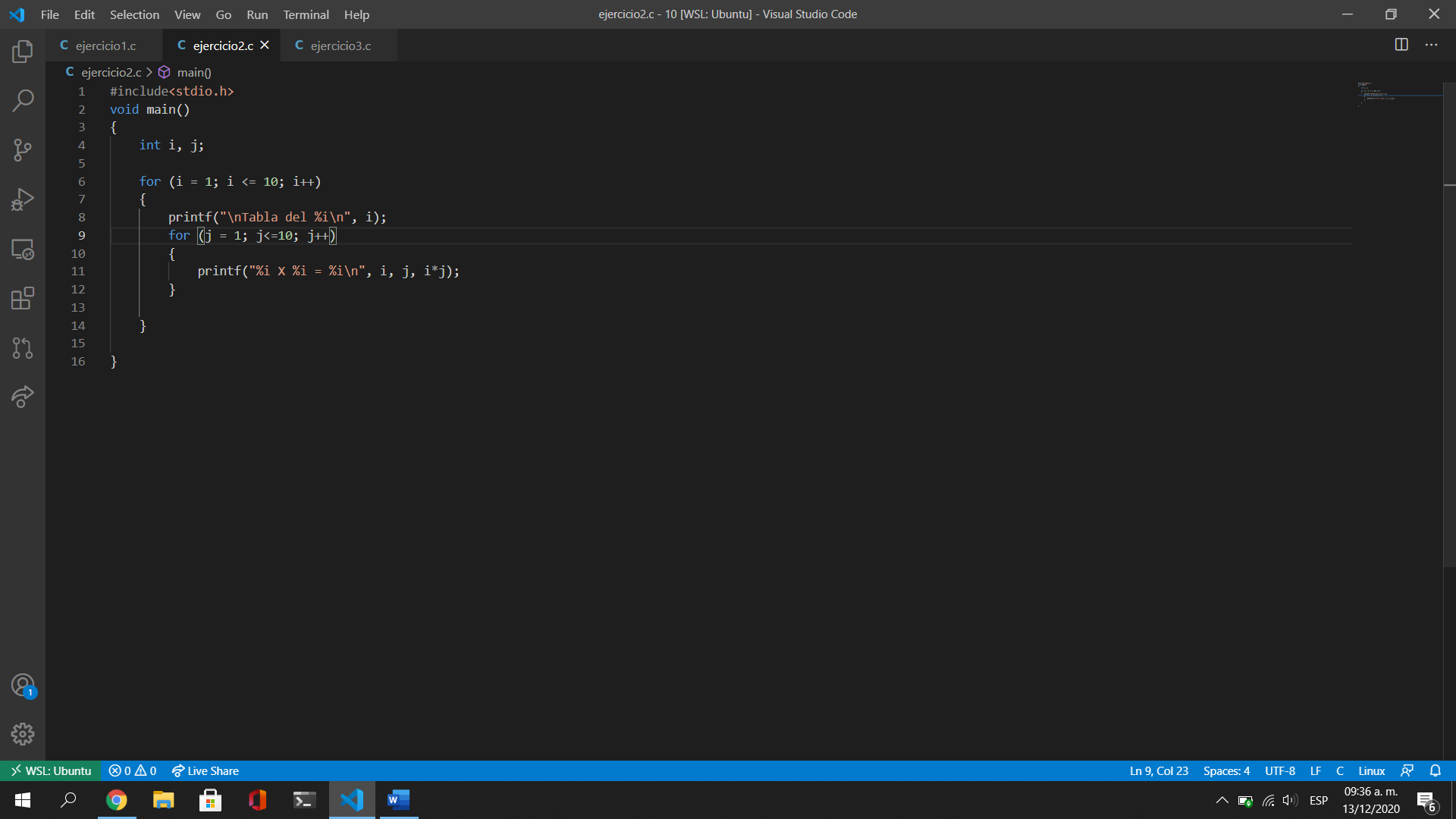
## Ejercicio 2



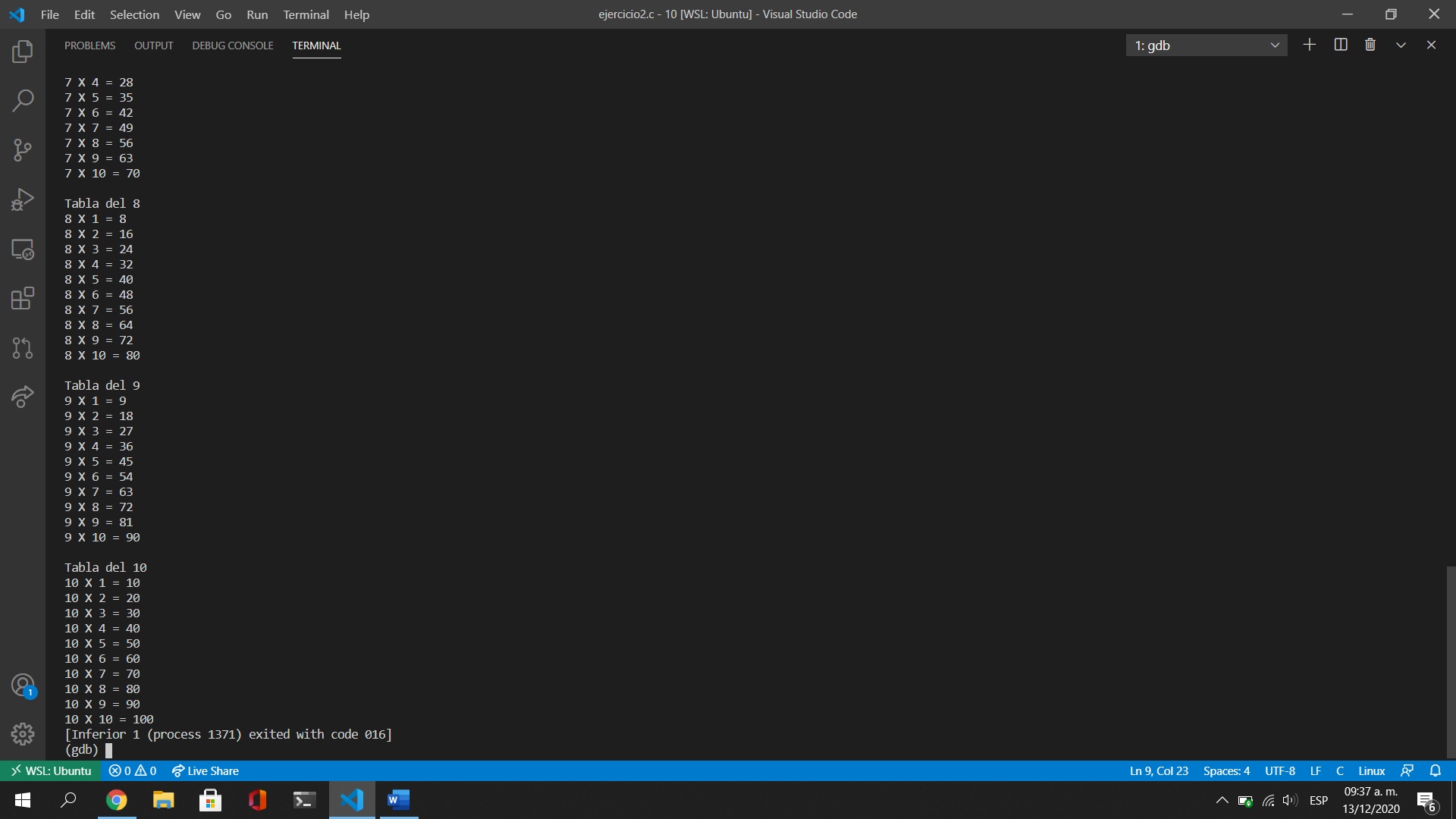
Transcribí el segundo ejercicio en “ejercicio2.c” dentro de una terminal remota de Ubuntu en VSCode



Al ejecutar gdb en la terminal, puse dos breakdows, uno por cada for en el código para determinar el porqué el código no funciona correctamente y es que porque ambos for estaban mal argumentados

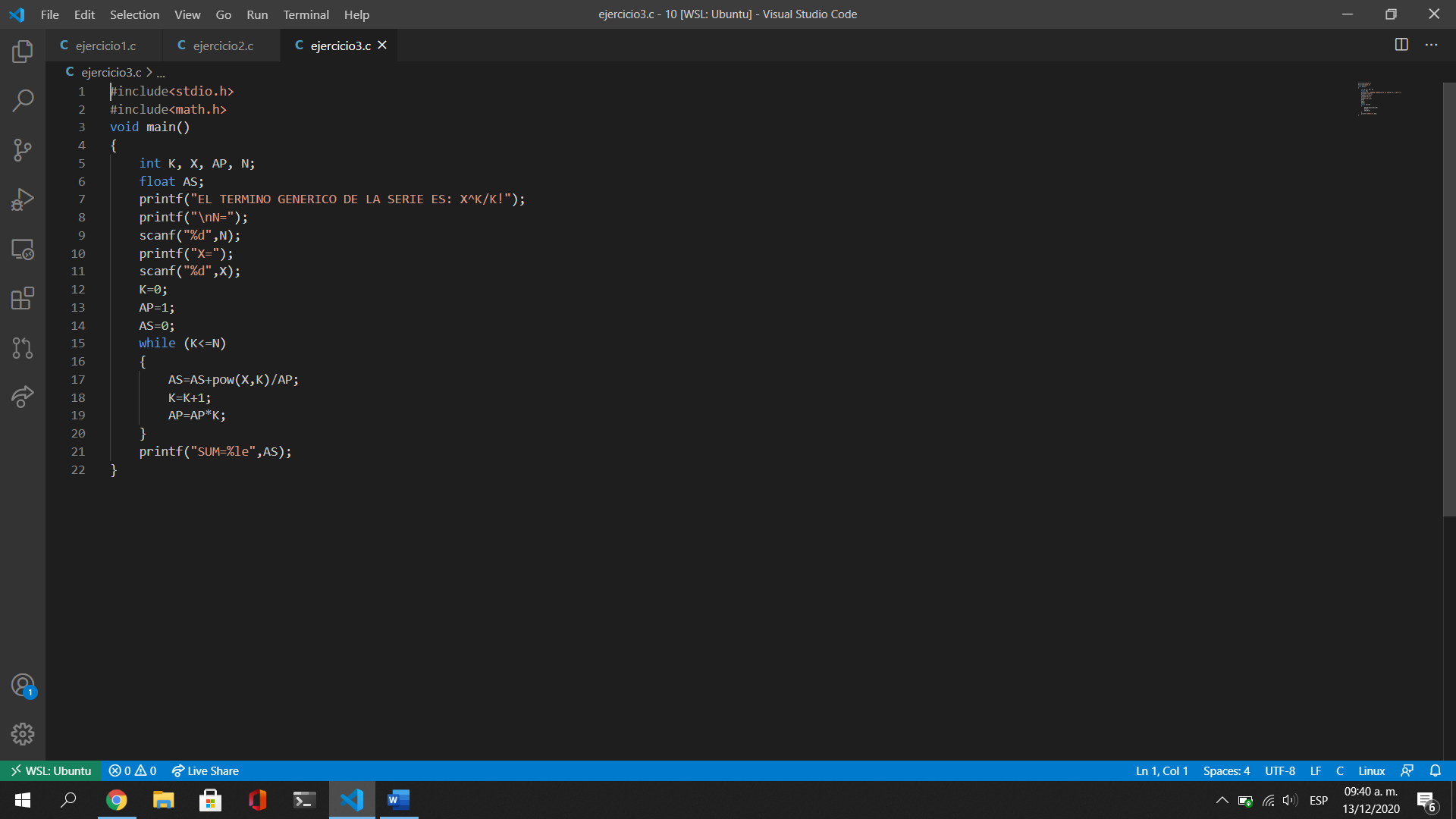


Cambié “i < 10” por un “i<=10” y el “j==10” por un “j<=10”

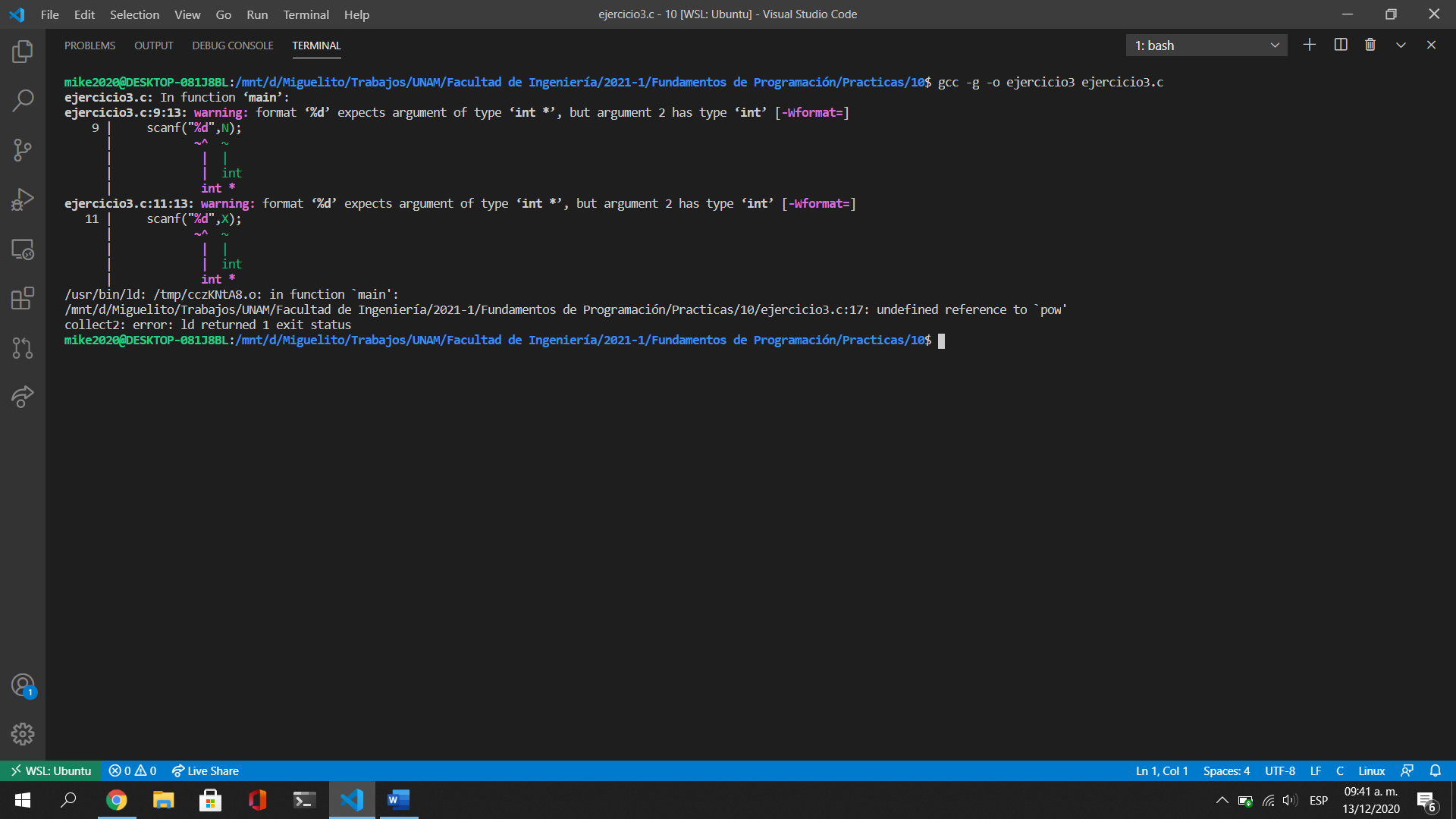


Al correr el programa dentro de gdb, ya me muestra las tablas de multiplicar correctamente, incluido la del 10

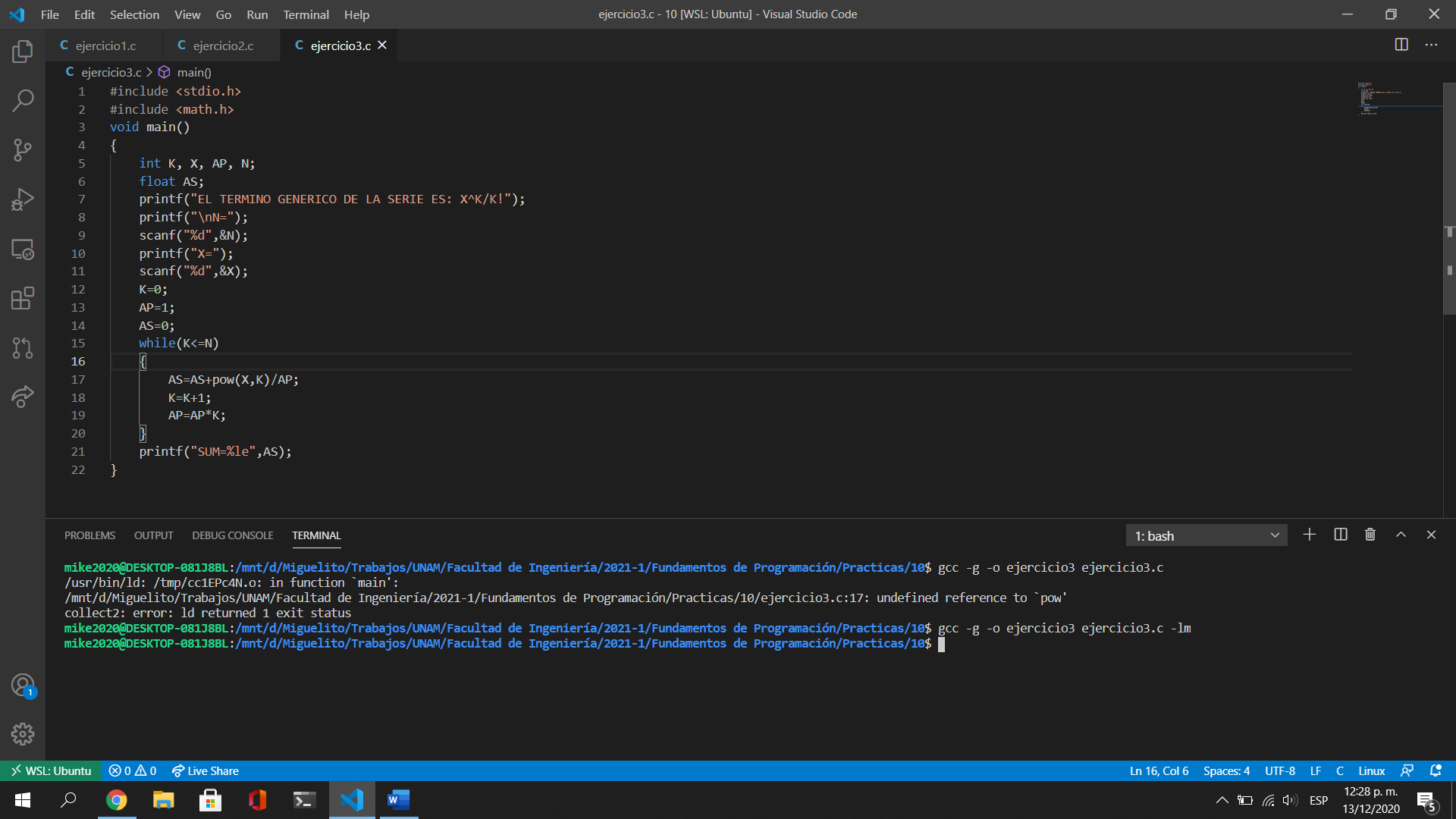
## Ejercicio 3



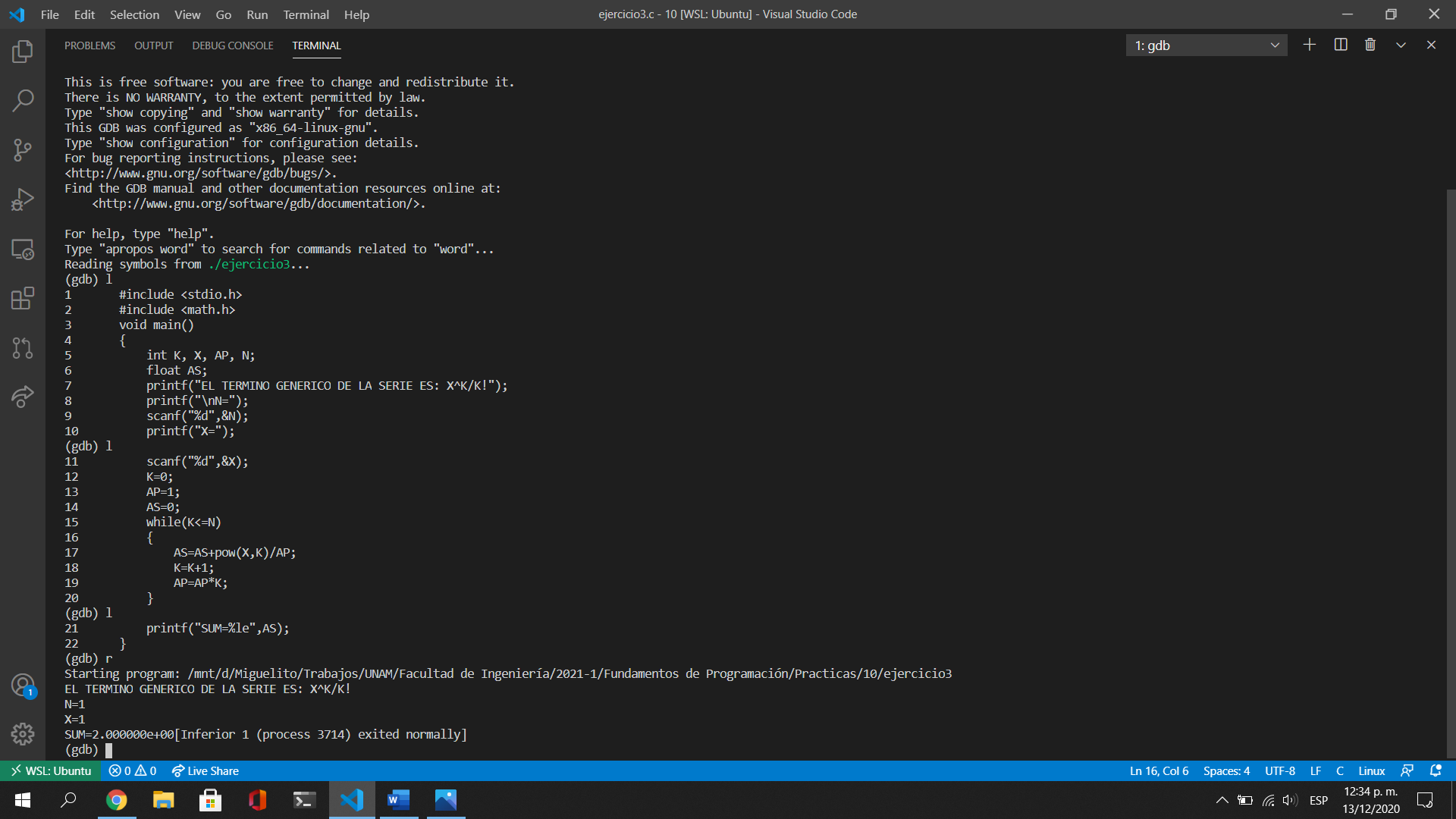
Transcribí el tercer ejercicio en “ejercicio3.c” dentro de una terminal remota de Ubuntu en VSCode



Al intentar compilar el programa, me marca que N y X no están declarados correctamente al leer los datos de entrada



Después de agregar “&” a X y N en scanf, intenté compilar el programa, pero me marcaba “undefined reference to `pow' collect2: error: ld returned 1 exit status”. Lo que hice fue agregar “-lm” para que pueda incluirse la librería de <math.h> al compilar el código



Al listar el código del programa y ejecutarlo dentro de gdb, se ejecuta correctamente

# Conclusiones

Me gustó realizar la práctica, aunque en un principio me costó trabajo entender cómo funciona lo de gdb para realizar lo que pide cada ejercicio propuesto.

# Referencias

Solano Gálvez, J. A., García Cano, E. E., Sandoval Montaño, L., Nakayama Cervantes, A., Arteaga Ricci, T. I., & Castañeda Perdomo, M. (6 de abril de 2018). *Manual de prácticas de Fundamentos de Programación MADO-17\_FP.* Recuperado el 7 de diciembre de 2020, de Fundamentos de Programación: http://lcp02.fi-b.unam.mx/