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

Laboratorios de computación

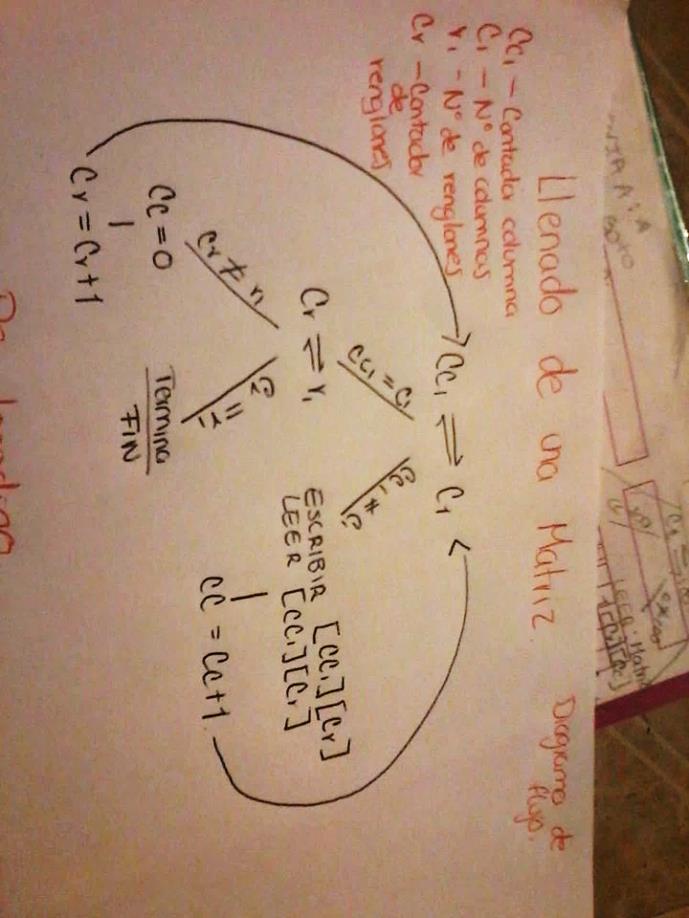
salas A y B

|  |  |
| --- | --- |
| *Profesor:* | Cruz Carlon Juan Alfredo M.C. |
| *Asignatura:* | Fundamentos de programación |
| *Grupo:* | 1107 |
| *N° de Práctica(s):* | Prácticas 8 y 9 |
| *Integrante(s):* | Silverio Letras Irving, Norman Eduardo, Brenda Juarez y Lizbeth Benitez |
|  |
|  |
| *Semestre:* | 2018-I |
| *Fecha de entrega:* | 03-Noviembre-2017 |

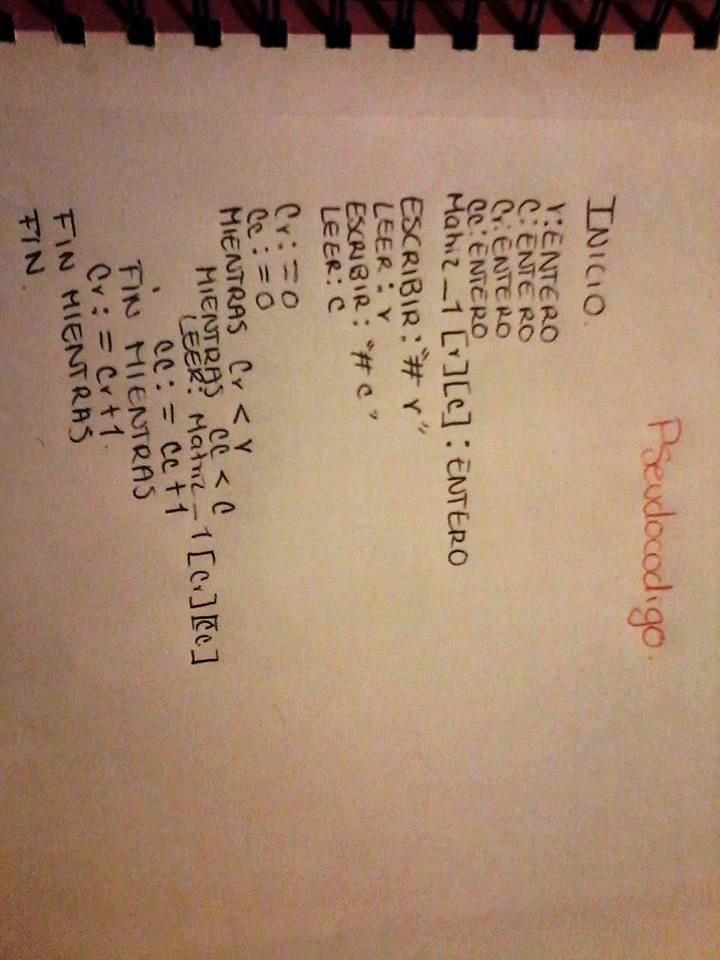
CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_

**LLENADO DE MATRICES**

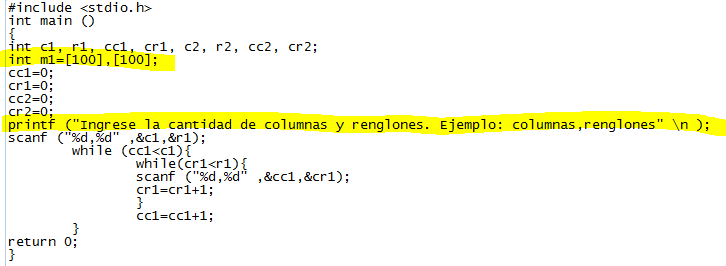
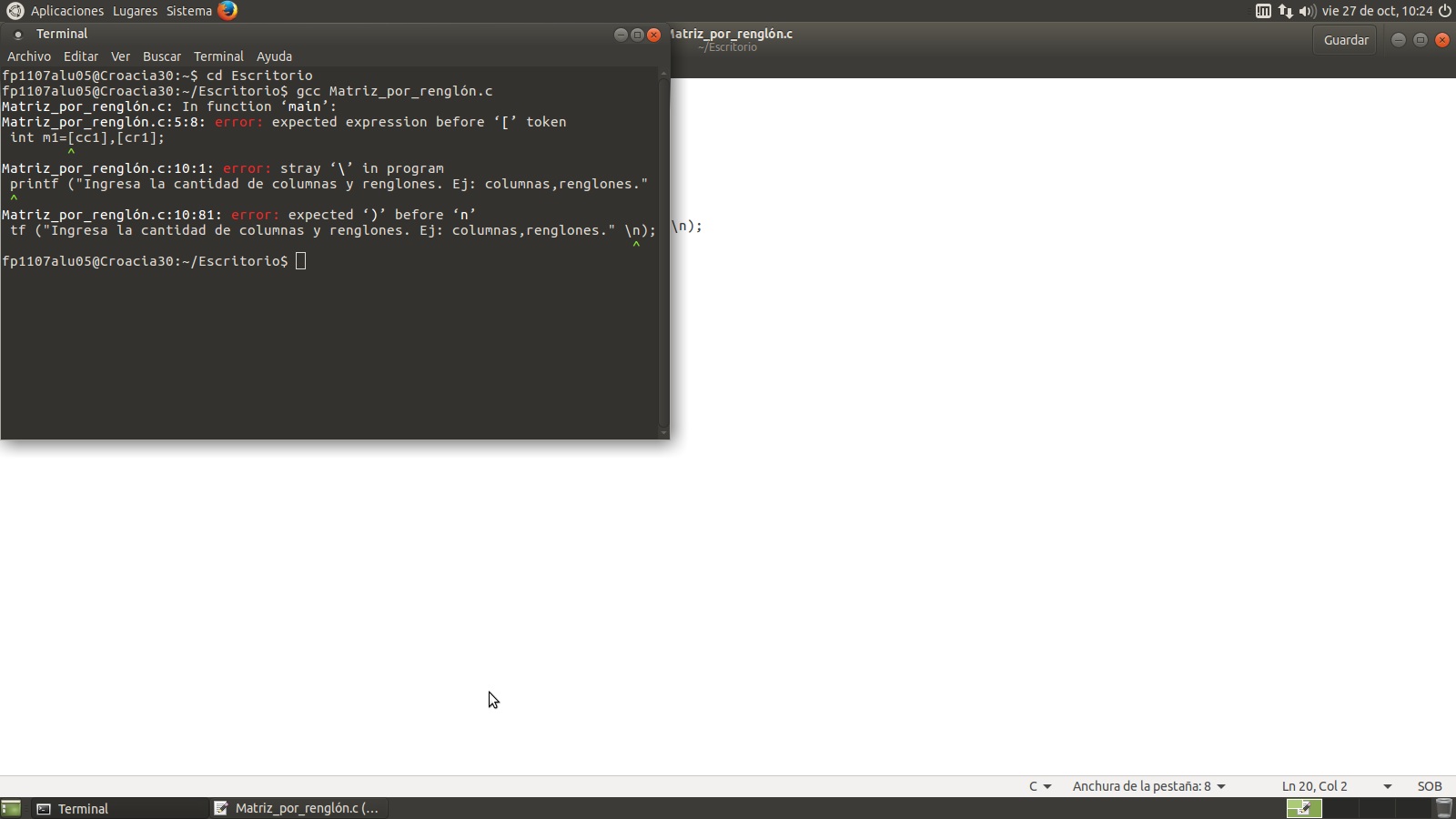
**Diagrama de flujo**



**Pseucódigo**



*Intentos previos al código*

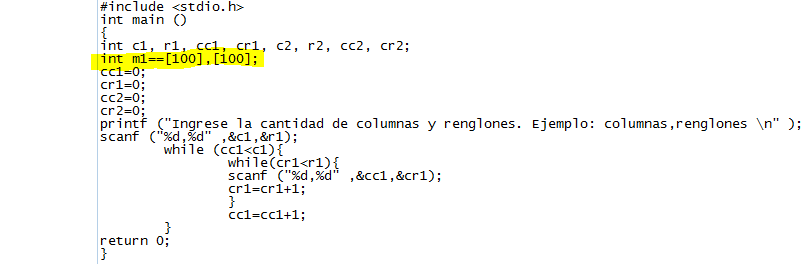
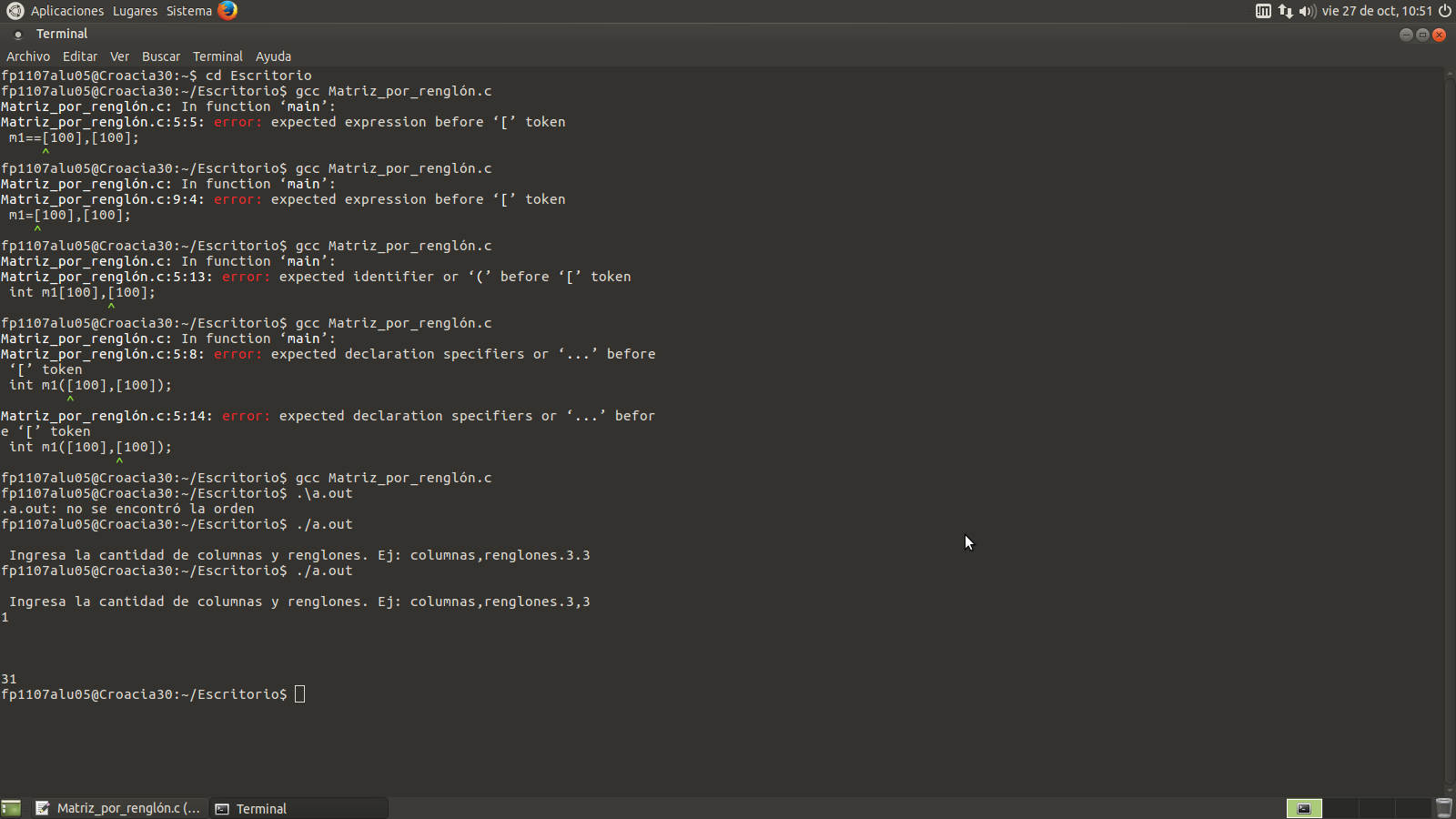
**.** ****

En éste primer intento tuvimos en total tres errores.

Inicialmente al declarar la variable “m1” colocamos un signo de igual que no va por lo cual no leía esa variable.

Después el salto de página en el “printf” lo colocamos fuera de las comillas lo cual está mal.

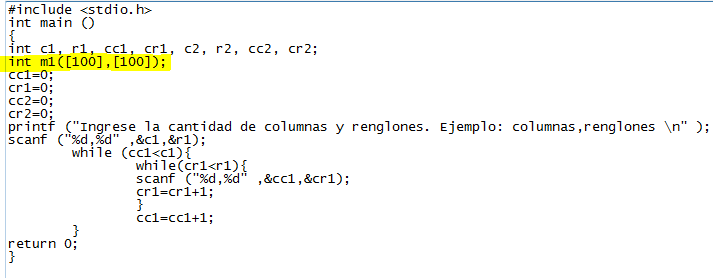
Finalmente en ese mismo renglón nos marcaba un error en la colocación del paréntesis.

****

En el segundo intento pudimos corregir el problema de salto de página incluyendo el “\n” dentro de los paréntesis.

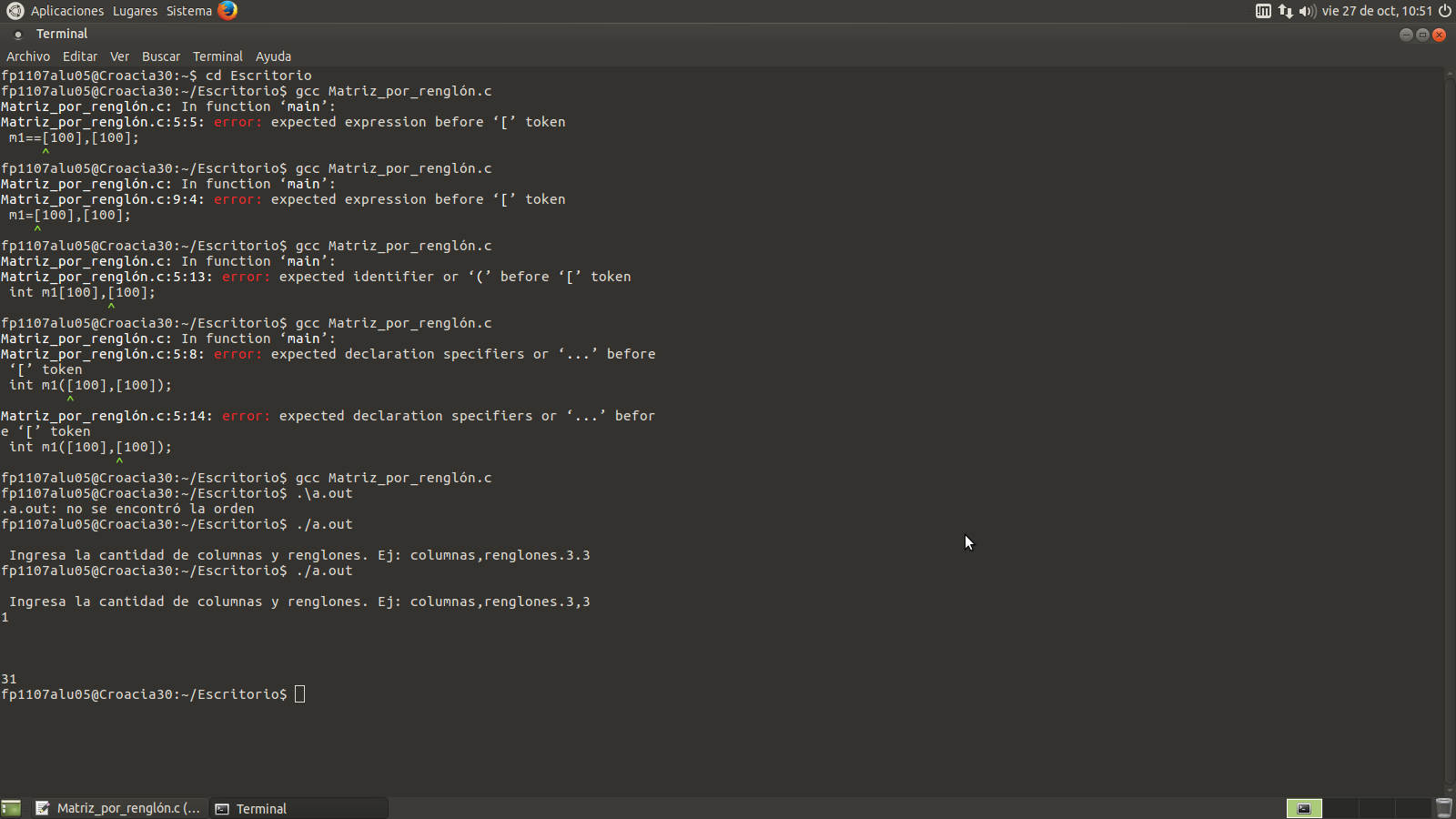
Sin embargo aún no comprendíamos nuestro problema con la declaración de la variable “m1”. Pensamos que quizá como lo habíamos planteado antes para darle un valor a una variable había que poner doble “=” pero por lo visto esto no resolvió nuestro error.

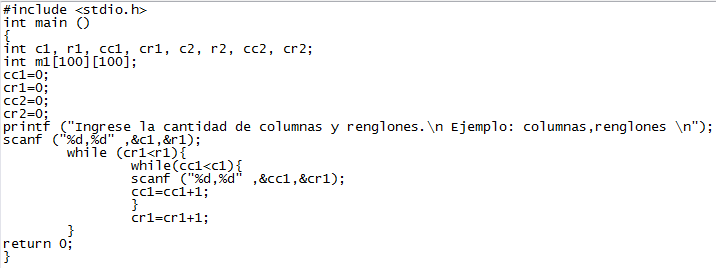
Nos marcaba un tercer error en medio de las casillas donde se supondría que irían los valores impresos.



En el tercer intento creímos que tal vez nos hacía falta agrupar la declaración de nuestra variable compuesta: “m1 [100][100];” por lo que la pusimos dentro de paréntesis con la idea de que el programa la tomara como un todo.

Pensamos que esto resolvería los dos problemas que nos quedaban.

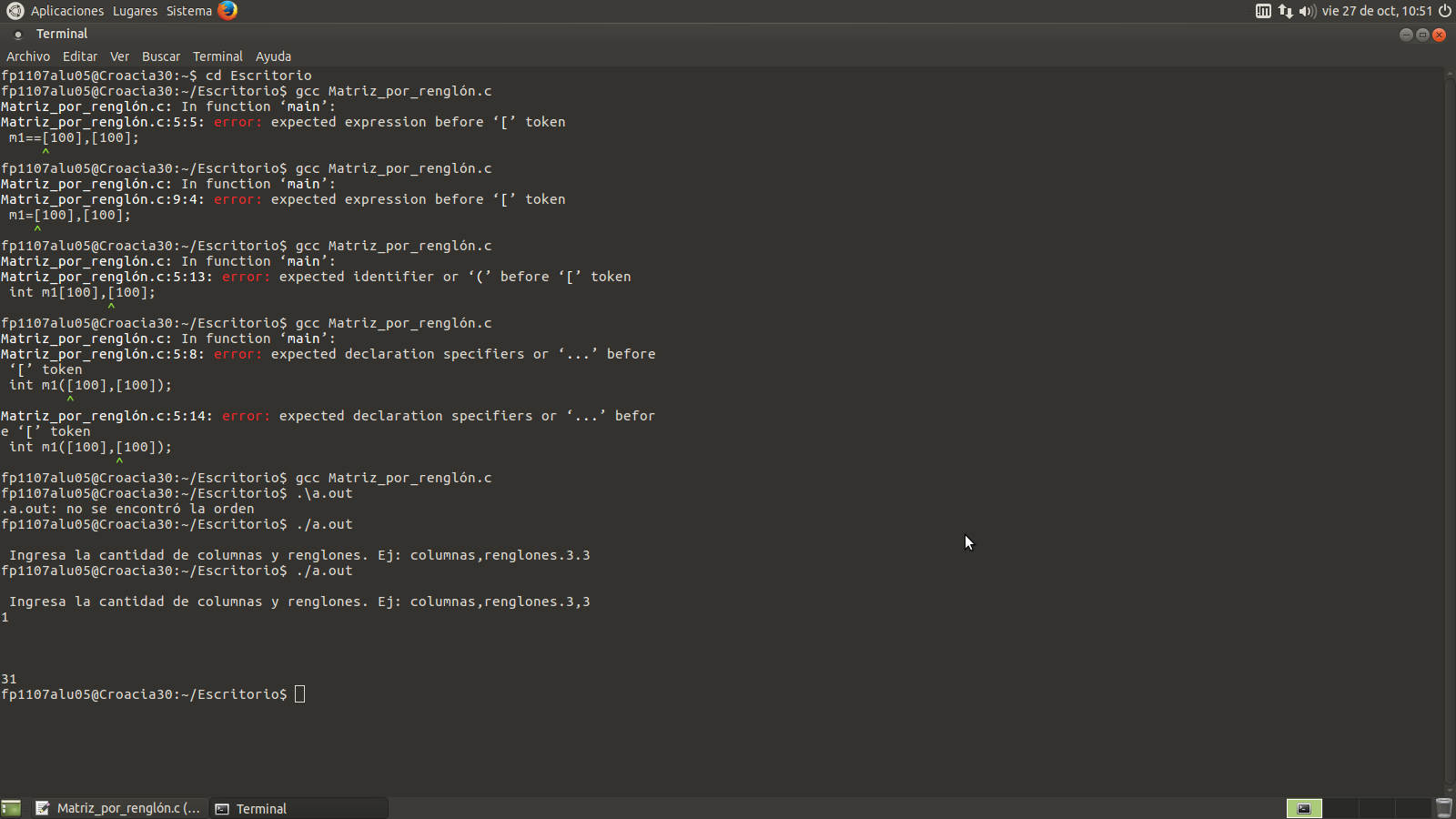
****



Finalmente en el cuarto intento, el programa era correcto.

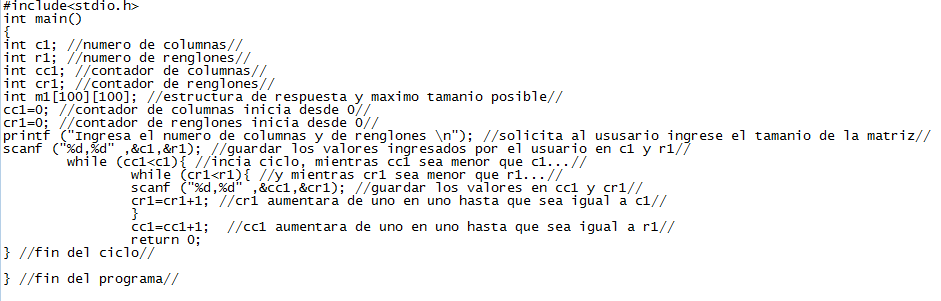
Comparando con la escritura en C que habíamos hecho anteriormente pudimos notar nuestros errores.

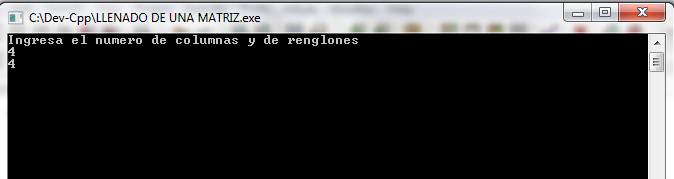
En la línea de la declaración de la variable “m1” no hacía falta poner ningún “=” además la coma que colocamos entre las casillas donde se imprimirían los resultados estaba de más.

****

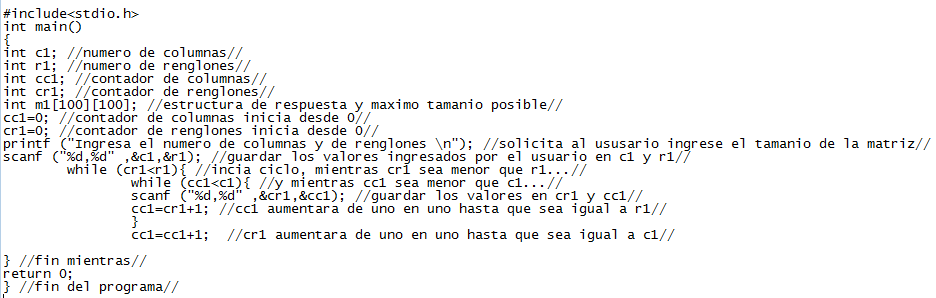
**Código**

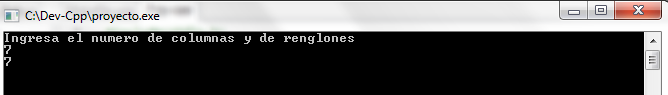
**Por columnas:**

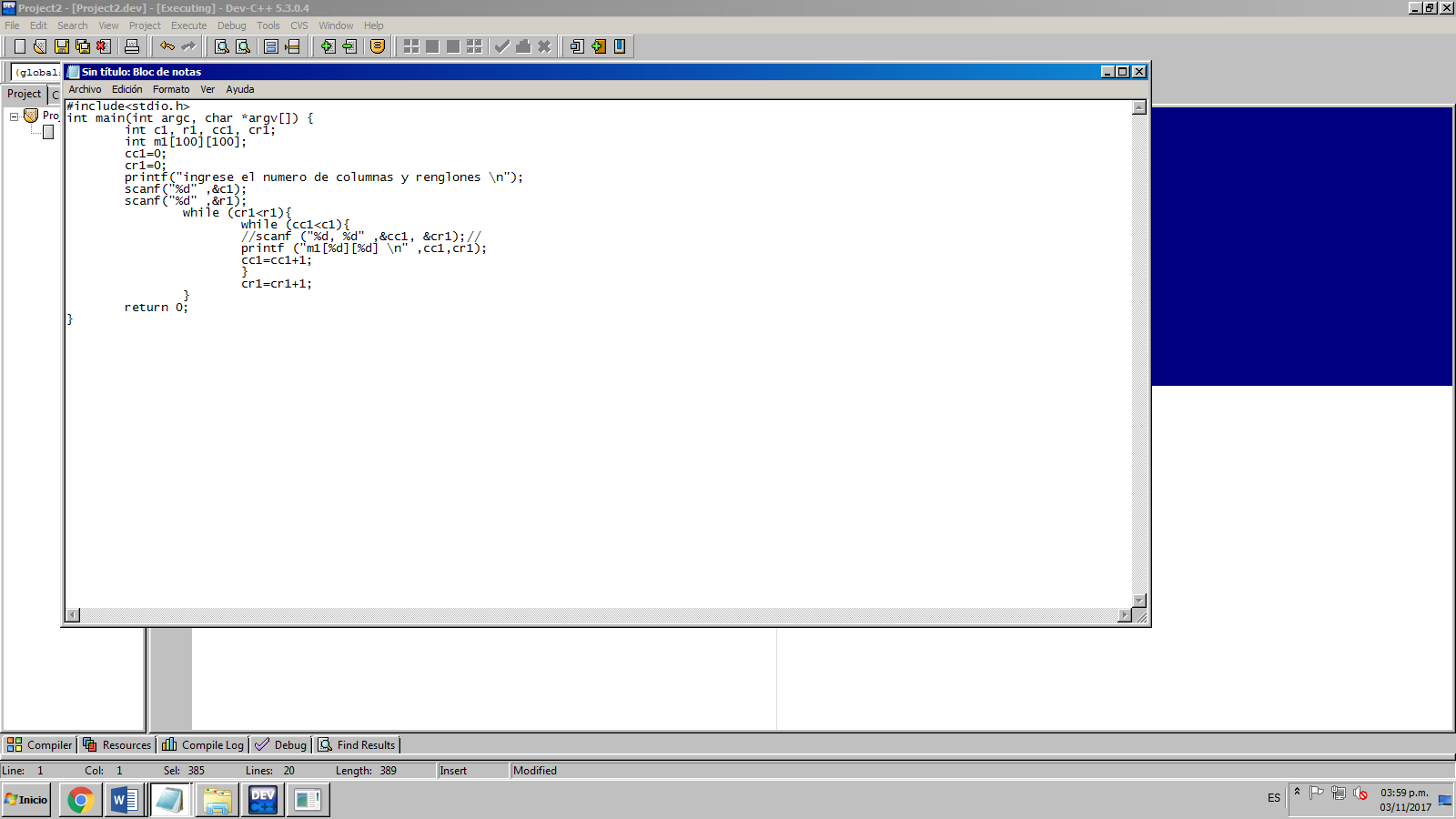


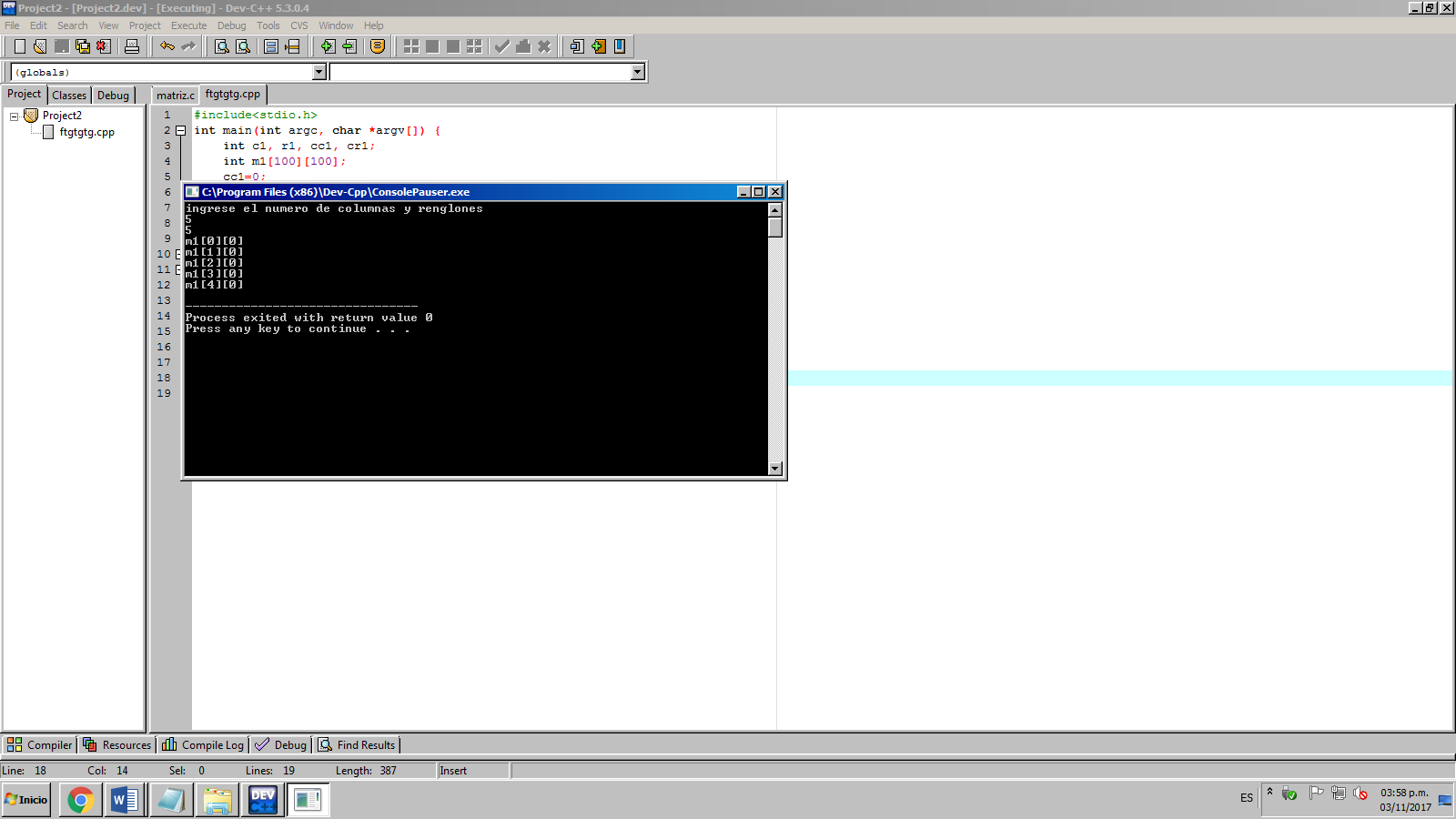


**Por renglones**



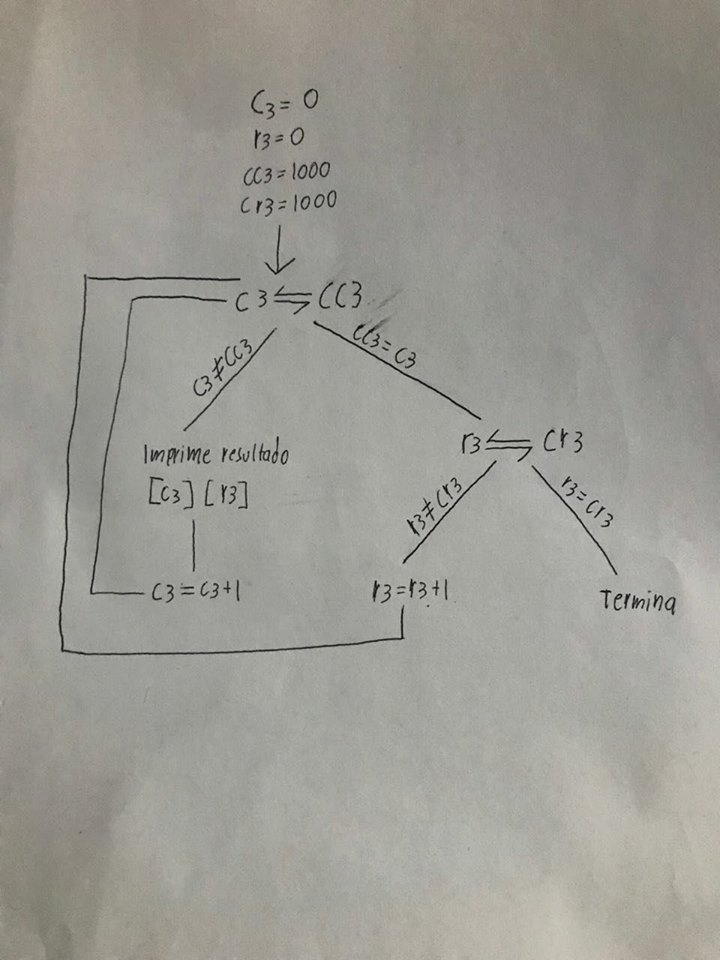






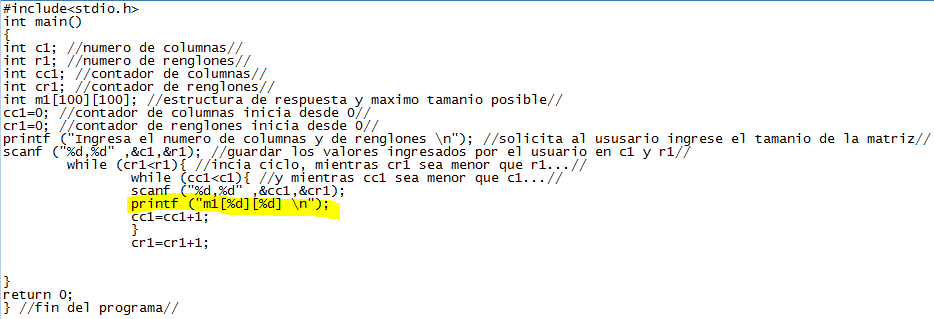
**IMPRESIÓN DE MATRICES**

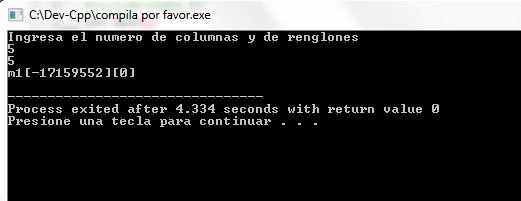
**Diagrama de flujo**



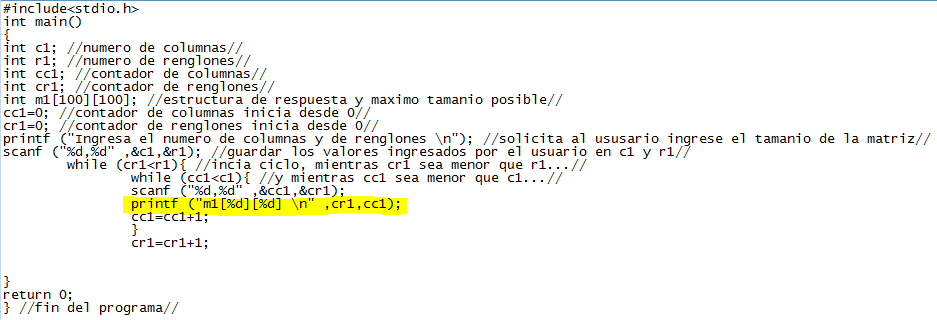
**Pseucódigo**

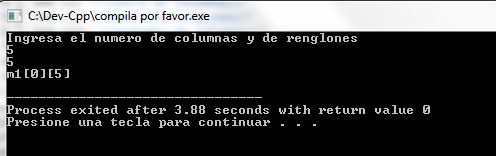
**Intentos previos al código**

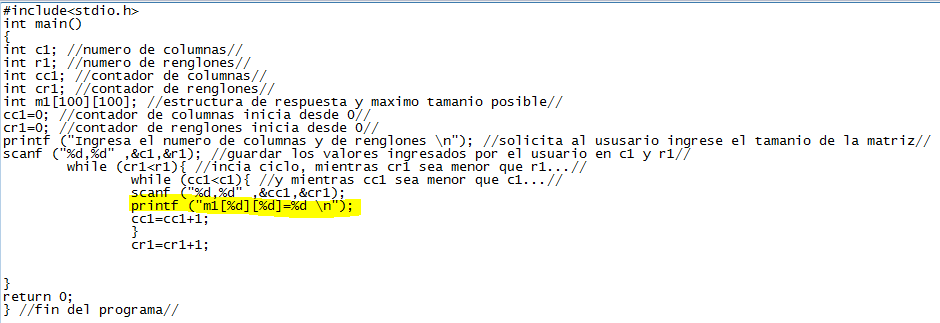


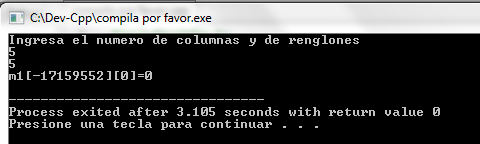


En este primer intento cometimos el error de no declarar de que variable tomaría el código los valores para imprimir al mandar el resultado en “printf (“m1 [%d][%d] \n”)

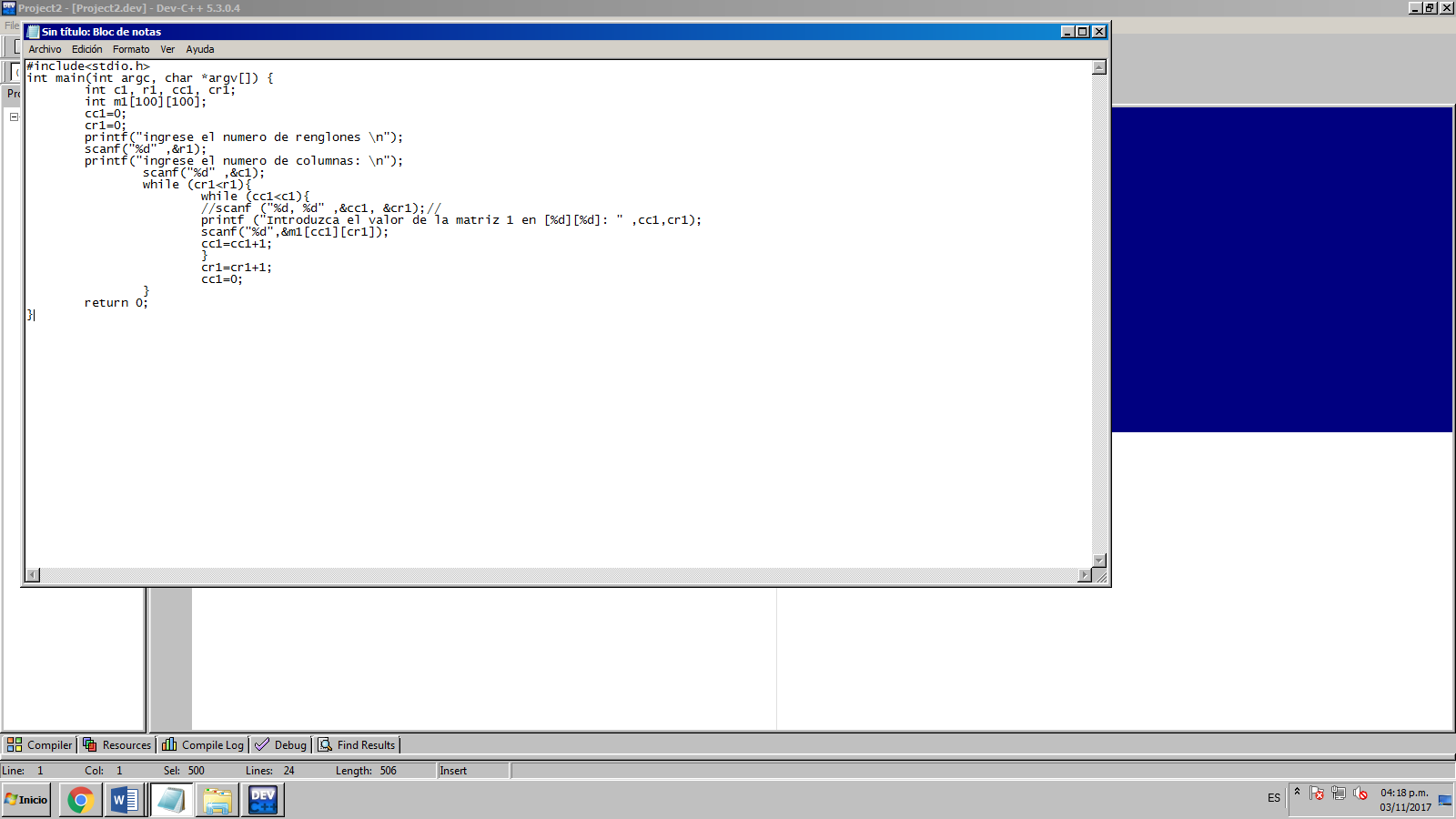


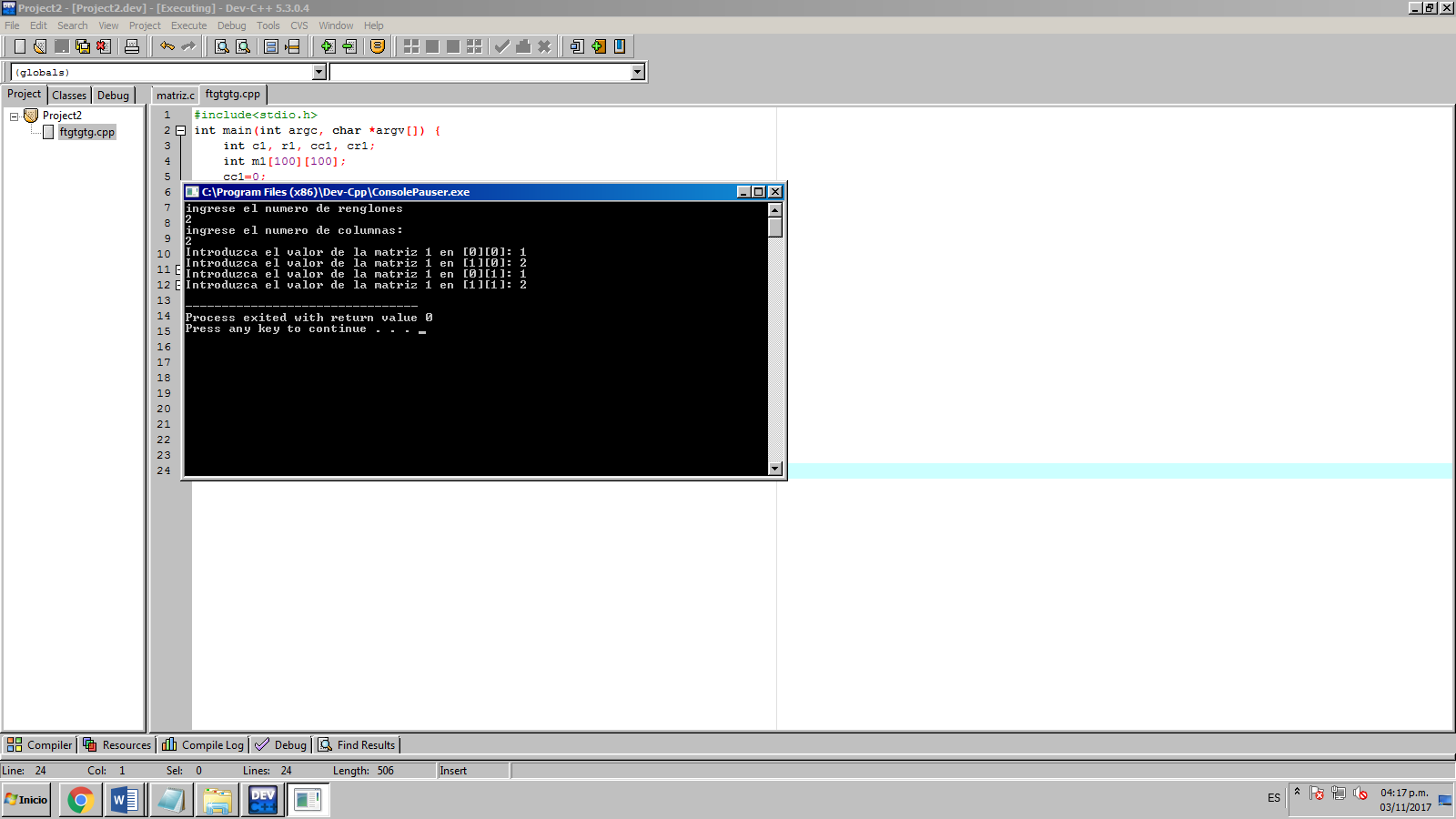




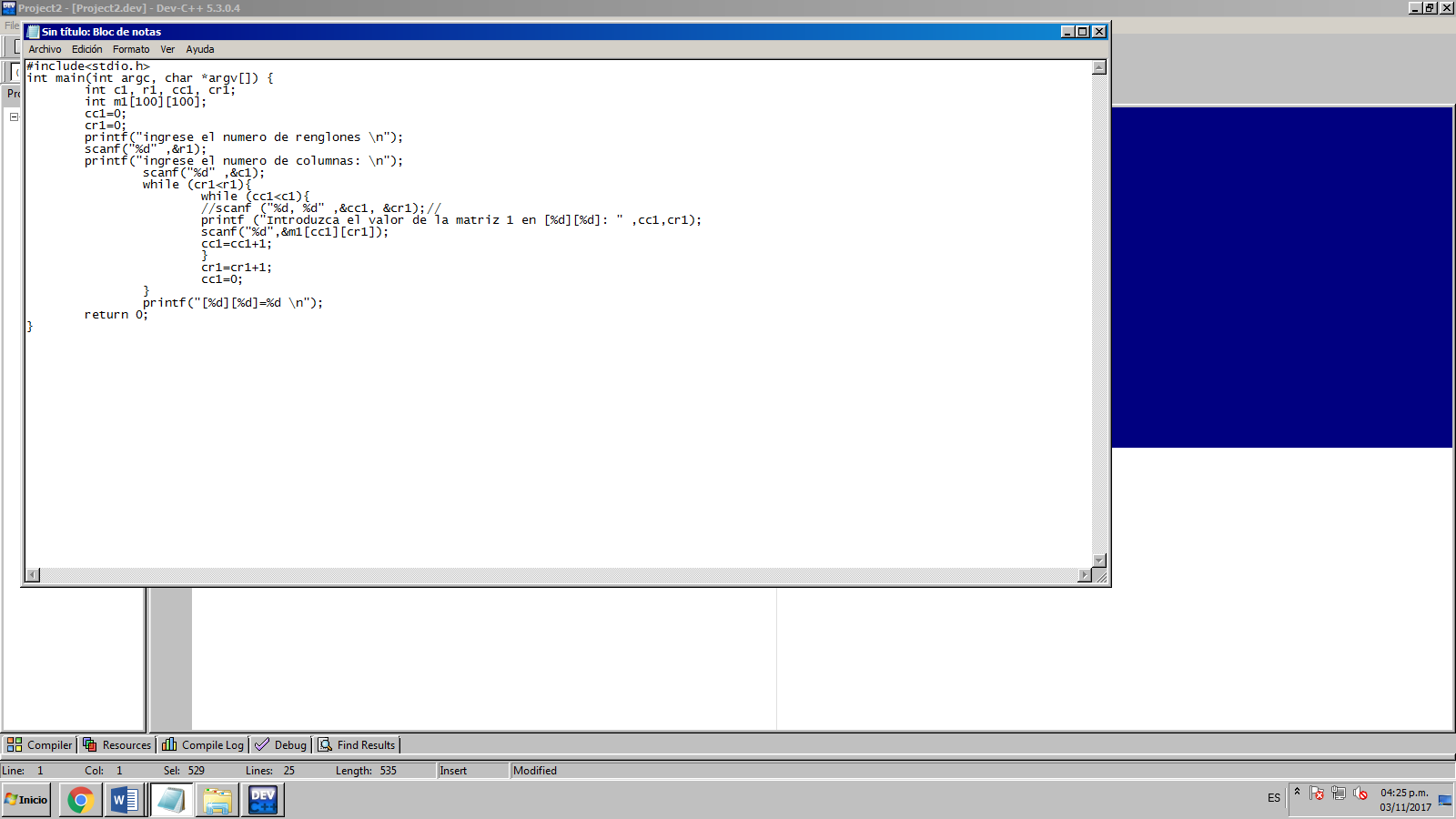


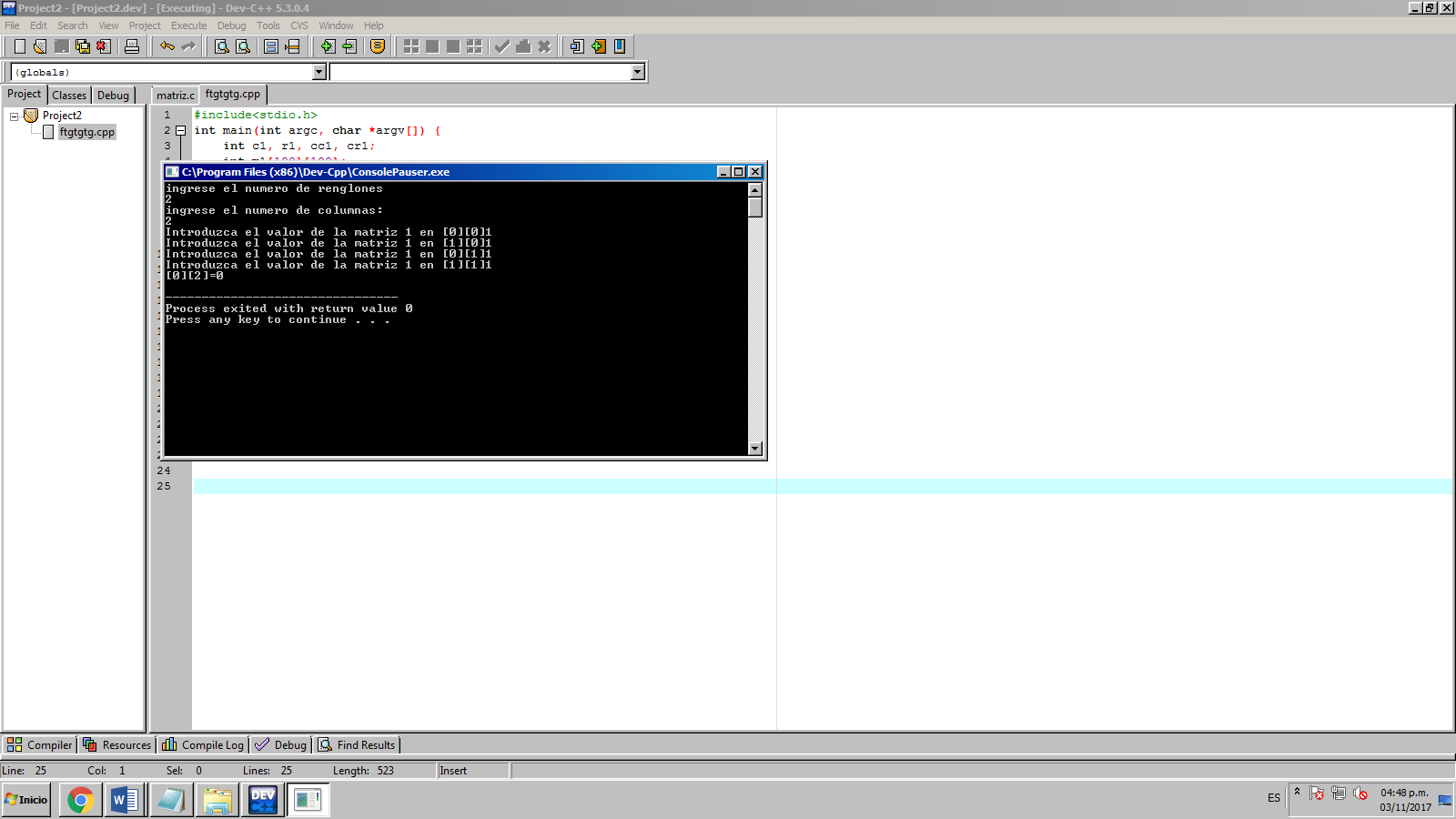
En este intento se imprime sólo una vez las coordenadas de la matriz, en columnas asigna un número del cual no conocemos su procedencia y en renglones mantiene el 0.



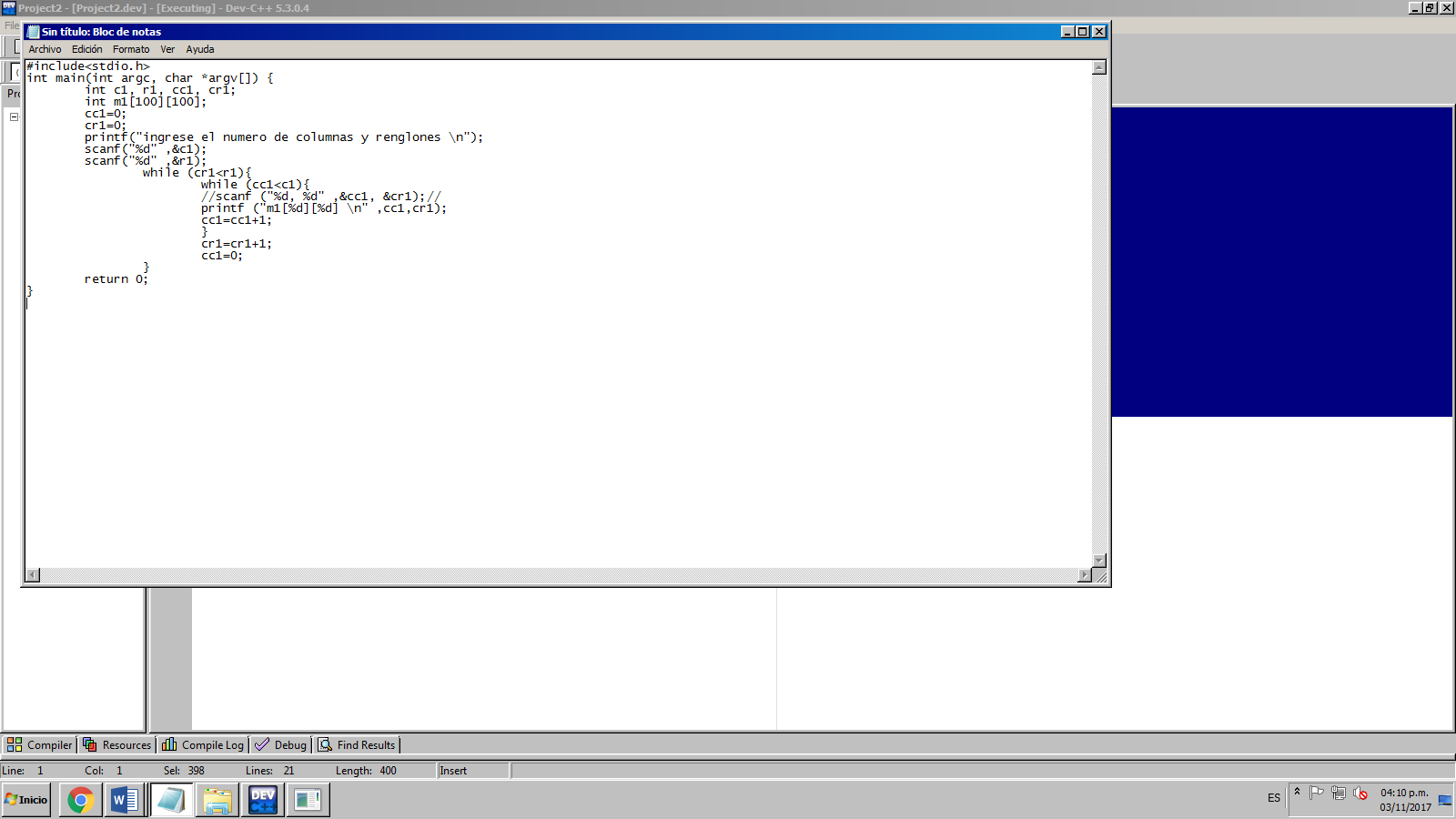


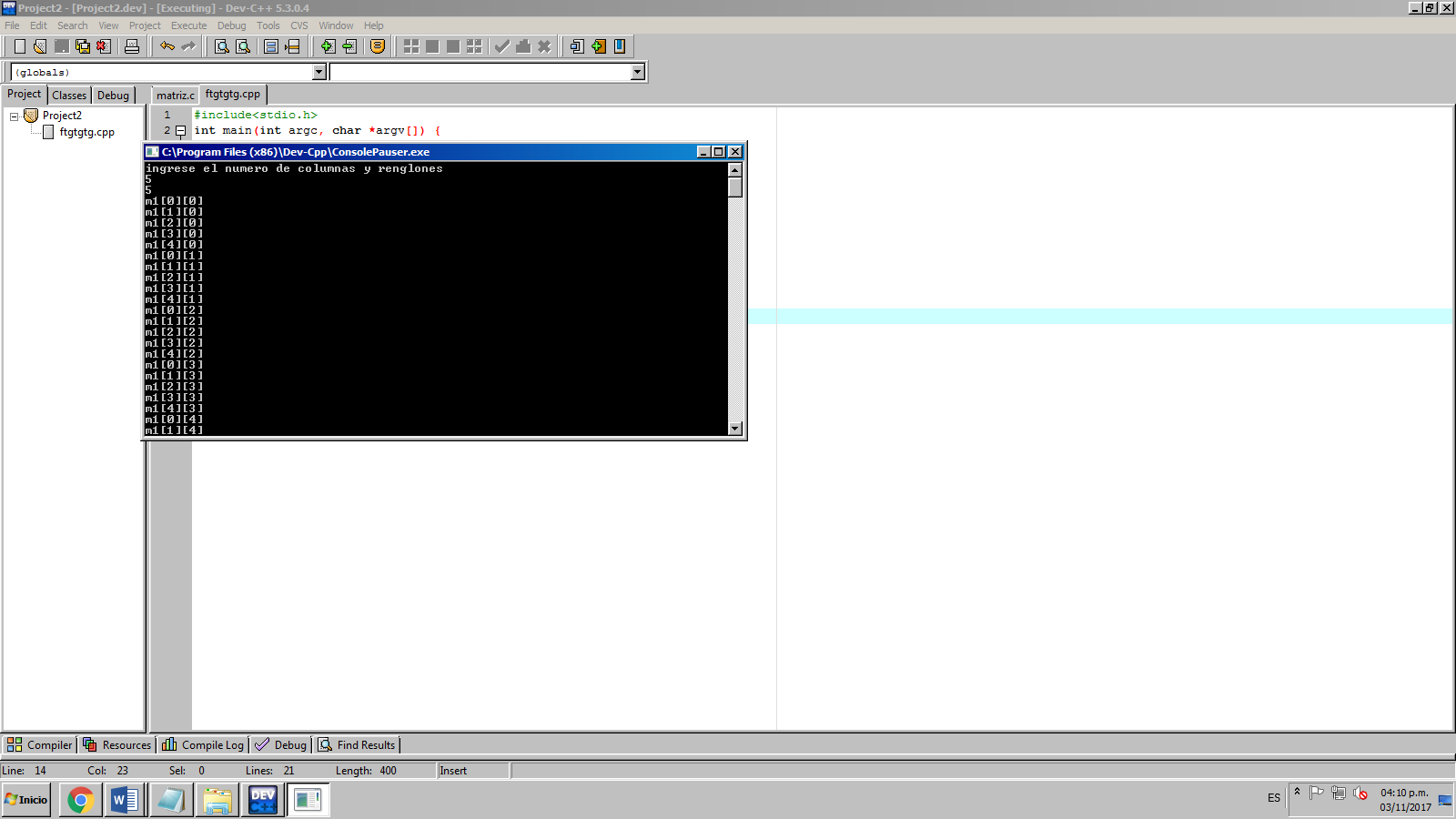
Aquí si se imprimen las coordenadas adecuadas para la matriz pero no estamos seguros de que la computadora guarde los valores que el usuario está ingresando.



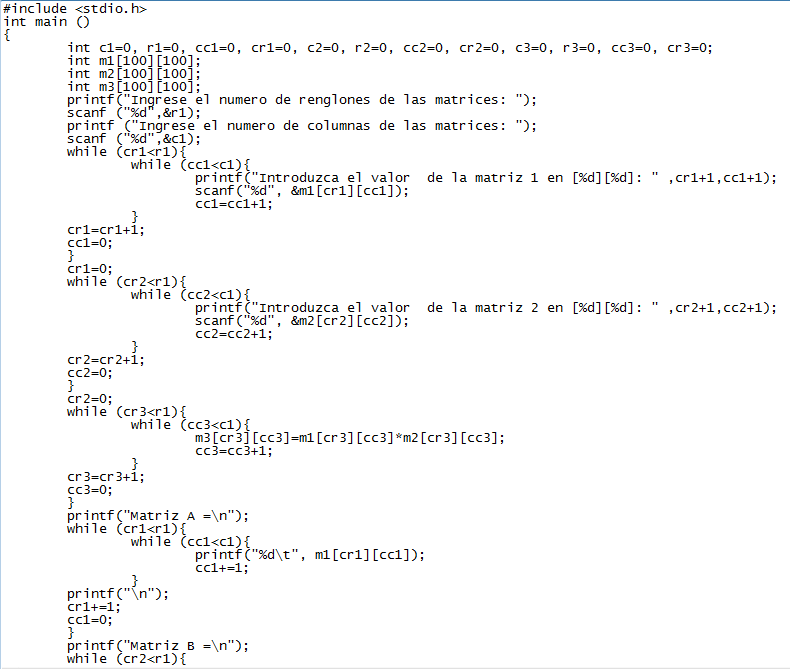


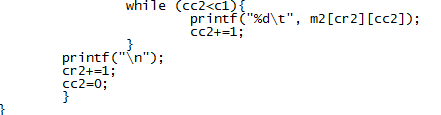
No funcionó la modificación que realizamos pues en lugar de comprobarnos que se estaban guardando los datos ingresados por el usuario, ahora asignaba la unidad para todas las coordenadas, sin importar lo que ingresara el usuario.

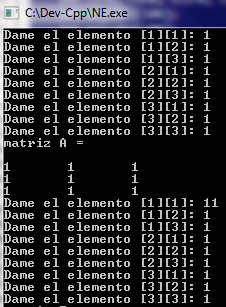


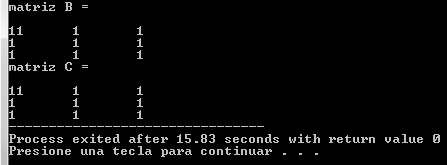


Este programa ya imprimía las m1 que se necesitaban y asignaba las coordenadas adecuadas para cada término









Finalmente logramos que se guardaran correctamente los valores que el usuario ingresó y que además los acomodara a partir de sus coordenadas.

**MULTIPLICACIÓN DE MATRICES**

**Diagrama de flujo**

**Pseucódigo**

**Intentos previos al código**

**Código**

