|  |
| --- |
| Laboratorio de Computación  Salas A y B |
| |  |  | | --- | --- | | Profesor: | Claudia Rodríguez Espino | | Asignatura: | Fundamentos de Programación | | Grupo: | 1104 | | No de Práctica(s): | 9 | | Integrante(s): | Téllez Torres Lorena Alejandra | |  |  | |  |  | |  |  | | No. de Equipo de cómputo empleado: | 51 | | Semestre: | 1ª | | Fecha de entrega: |  | | Observaciones: |  | |  |  | |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Objetivo**

Elaborar programas en C para la resolución de problemas básicos que incluyan las estructuras de repetición y la directiva define.

**Desarrollo**

Las estructuras de repetición son las llamadas estructuras cíclicas, iterativas o de bucles.

Permiten ejecutar un conjunto de instrucciones de manera repetida (o cíclica) mientras que la expresión lógica a evaluar se cumpla (sea verdadera).

En lenguaje C existen tres estructuras de repetición: while, do-while y for. Las estructuras while y do-while son estructuras repetitivas de propósito general.

Estructura de control repetitiva while

La estructura repetitiva (o iterativa) while primero valida la expresión lógica y si ésta se cumple (es verdadera) procede a ejecutar el bloque de instrucciones de la estructura, el cual está delimitado por las llaves {}. Si la condición no se cumple se continúa el flujo normal del programa sin ejecutar el bloque de la estructura, es decir, el bloque se puede ejecutar de cero a ene veces. Su sintaxis es la siguiente:

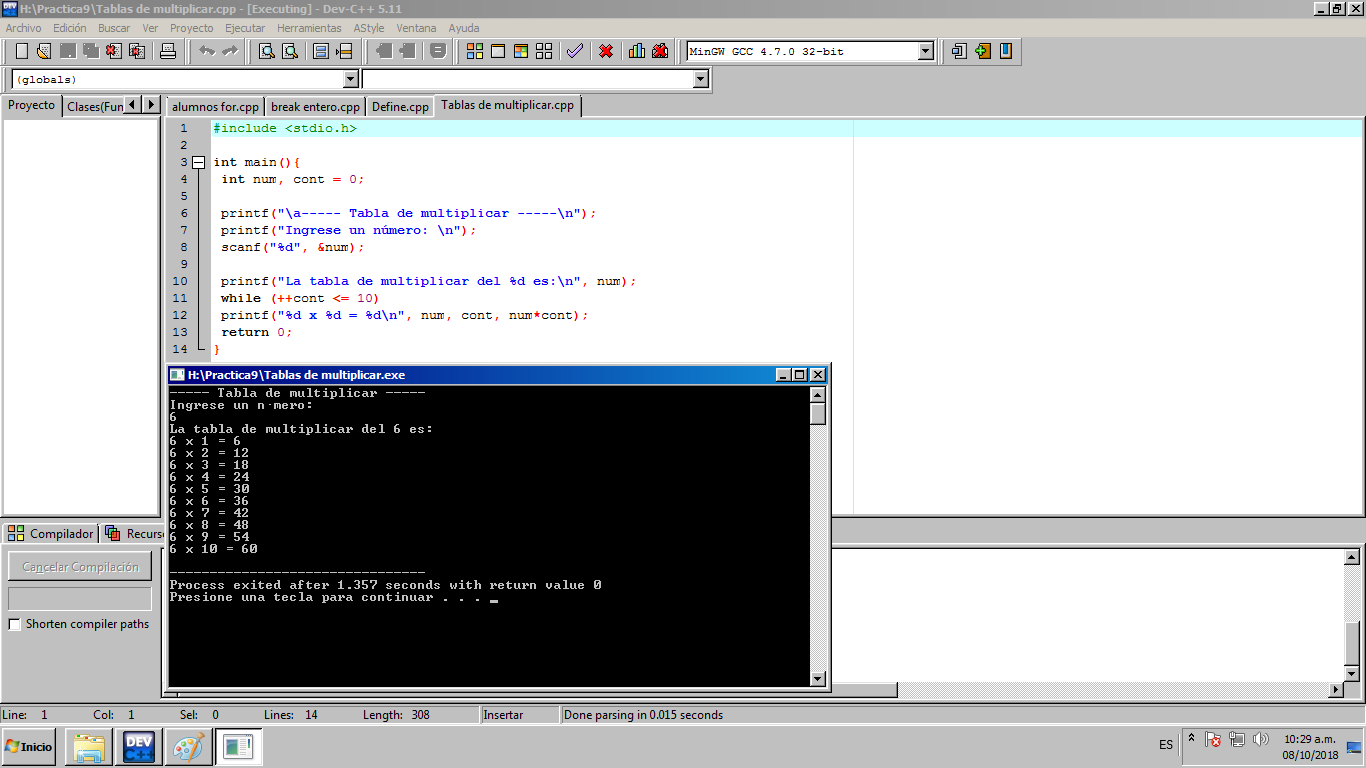
*while (expresión\_lógica) {*

*// Bloque de código a repetir*

*// mientras que la expresión*

*// lógica sea verdadera.*

*}*



**Estructura de control repetitiva do-while do-while**

Es una estructura cíclica que ejecuta el bloque de código que se encuentra dentro de las llaves y después valida la condición, es decir, el bloque de código se ejecuta de una a ene veces. Su sintaxis es la siguiente:

*do {*

*/\**

*Bloque de código que se ejecuta*

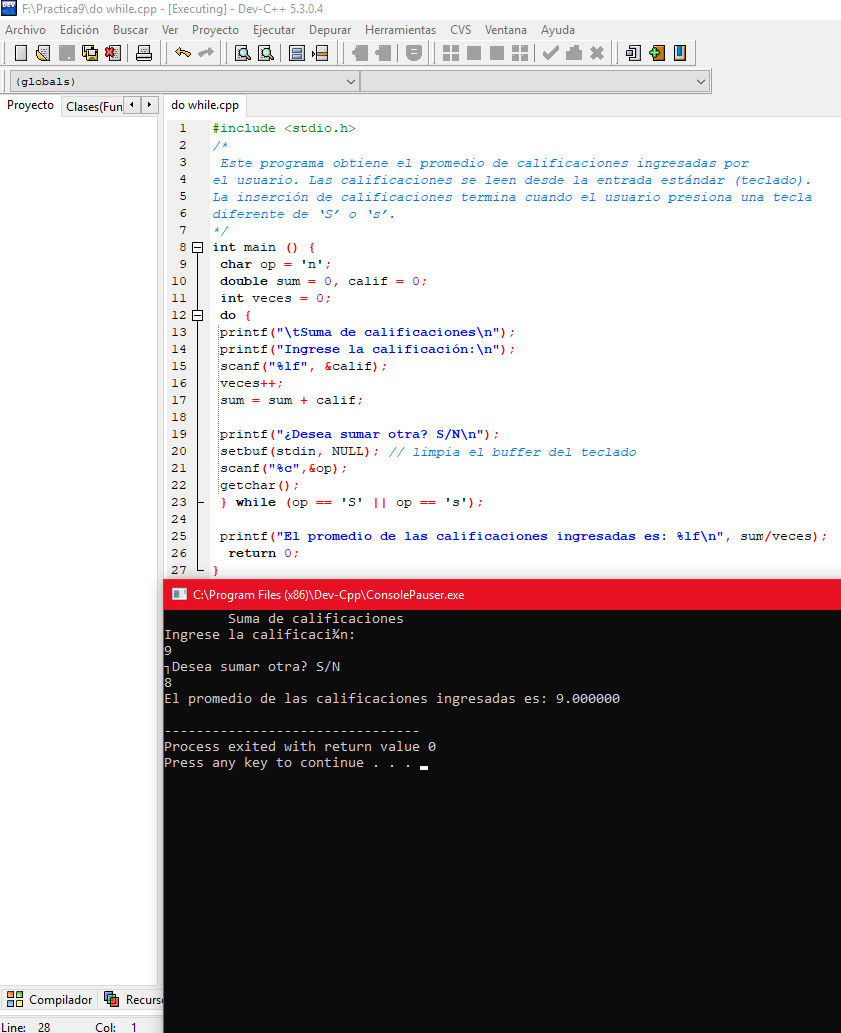
*por lo menos una vez y se repite*

*mientras la expresión lógica sea*

*verdadera.*

*\*/*

*} while (expresión\_lógica);*



**Estructura de control de repetición for**

Lenguaje C posee la estructura de repetición for la cual permite realizar repeticiones cuando se conoce el número de elementos que se quiere recorrer. La sintaxis que generalmente se usa es la siguiente:

*for (inicialización ; expresión\_lógica ; operaciones por iteración) {*

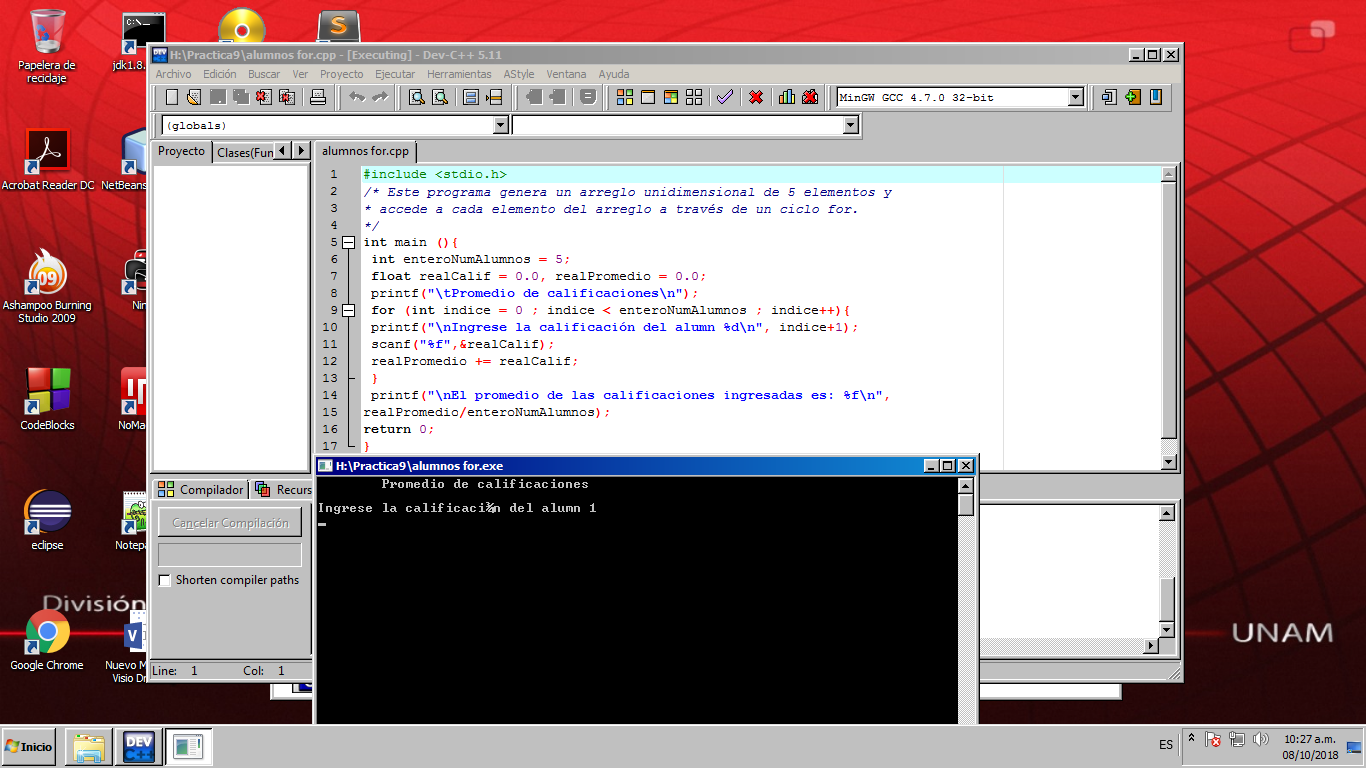
*/\**

*Bloque de código*

*a ejecutar*

*\*/*

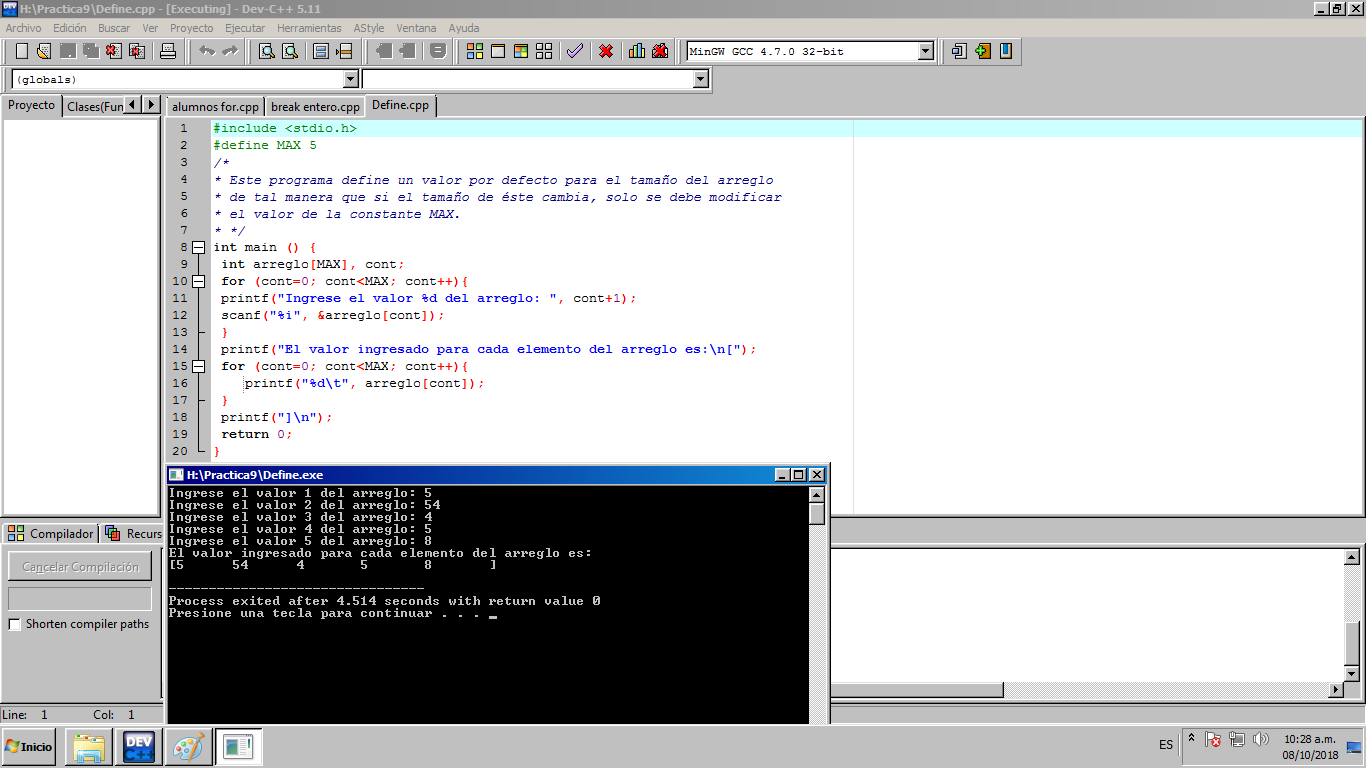
*}*

**

**Define**

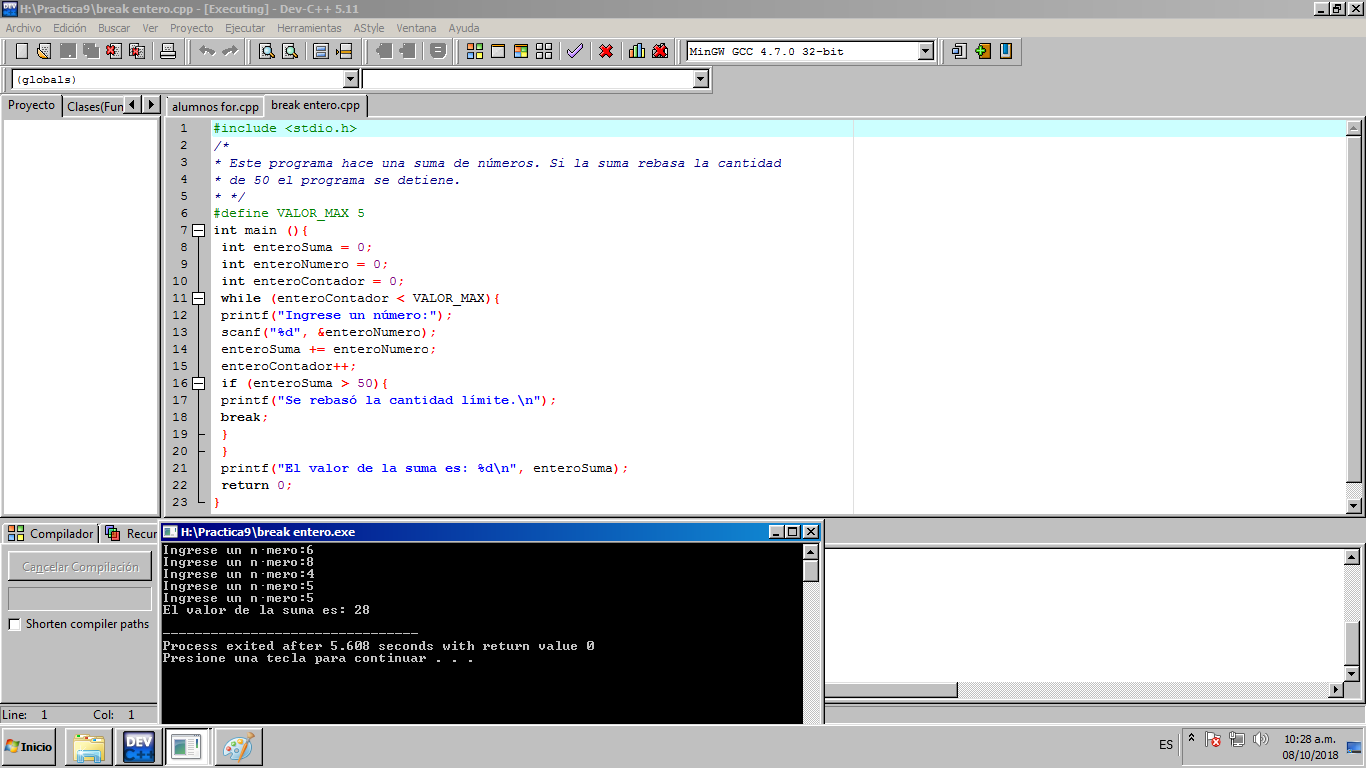
Las líneas de código que empiezan con # son directivas del preprocesador, el cual se encarga de realizar modificaciones en el texto del código fuente, como reemplazar un símbolo definido con #define por un parámetro o texto, o incluir un archivo en otro archivo con #include. Define permite definir constantes o literales; se les nombra también como constantes simbólicas. Su sintaxis es la siguiente:

*#define <nombre> <valor>*



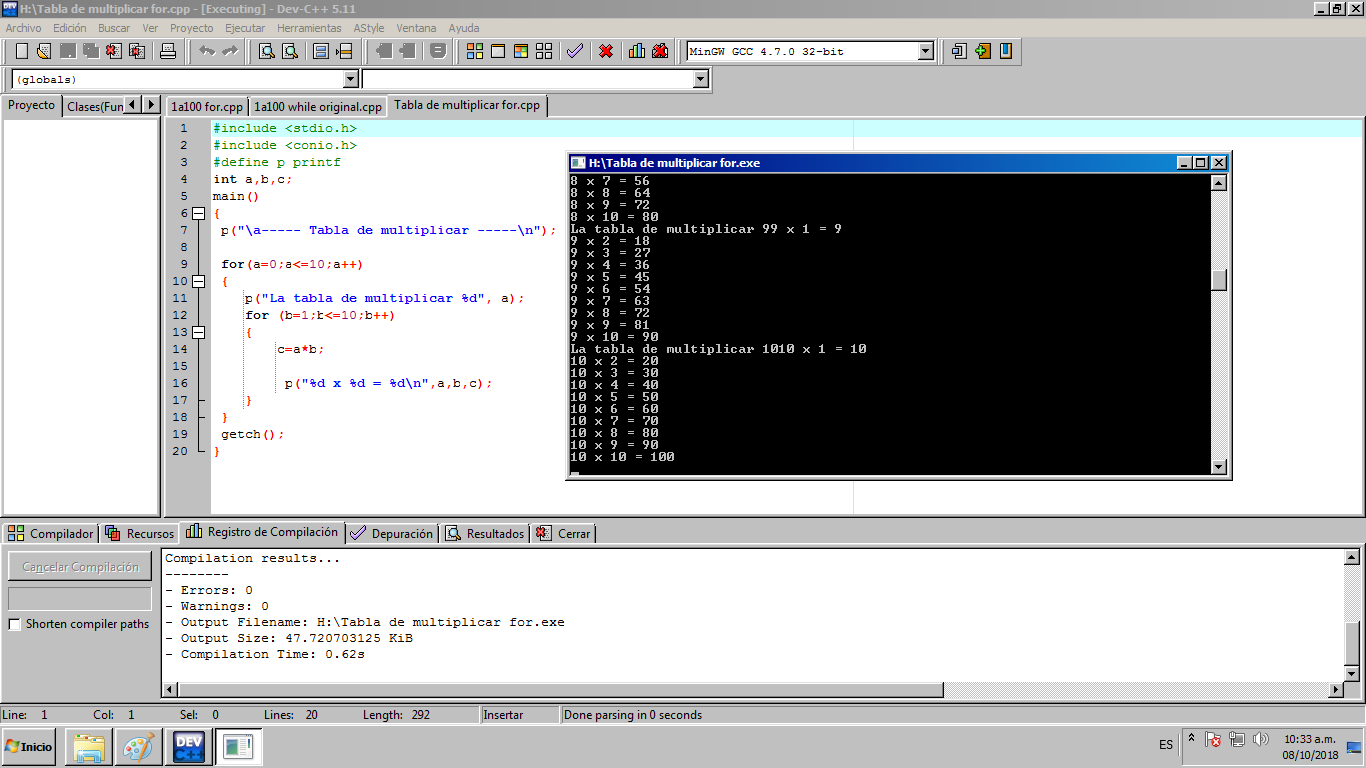
**Break**

Algunas veces es conveniente tener la posibilidad de abandonar un ciclo. La proposición break proporciona una salida anticipada dentro de una estructura de repetición, tal como lo hace en un switch. Un break provoca que el ciclo que lo encierra termine inmediatamente.

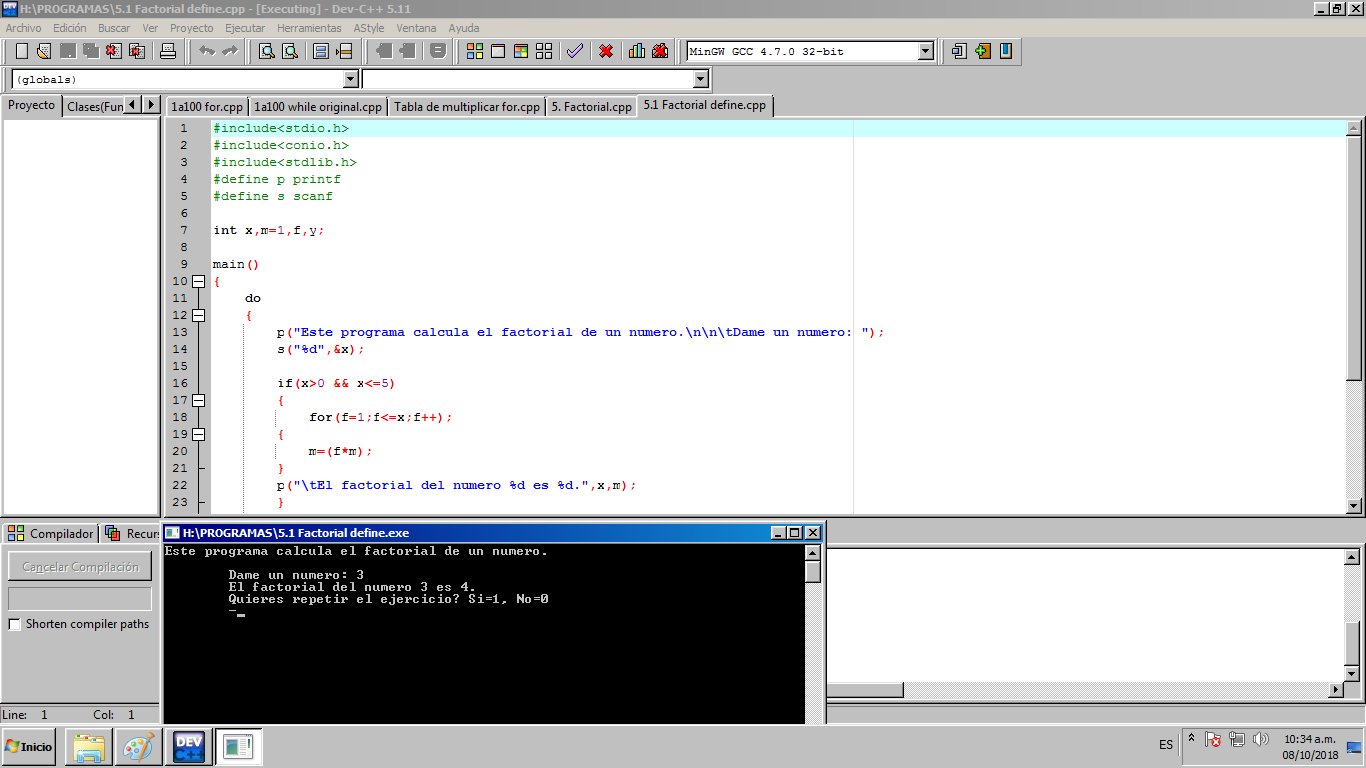


**Ejercicios**

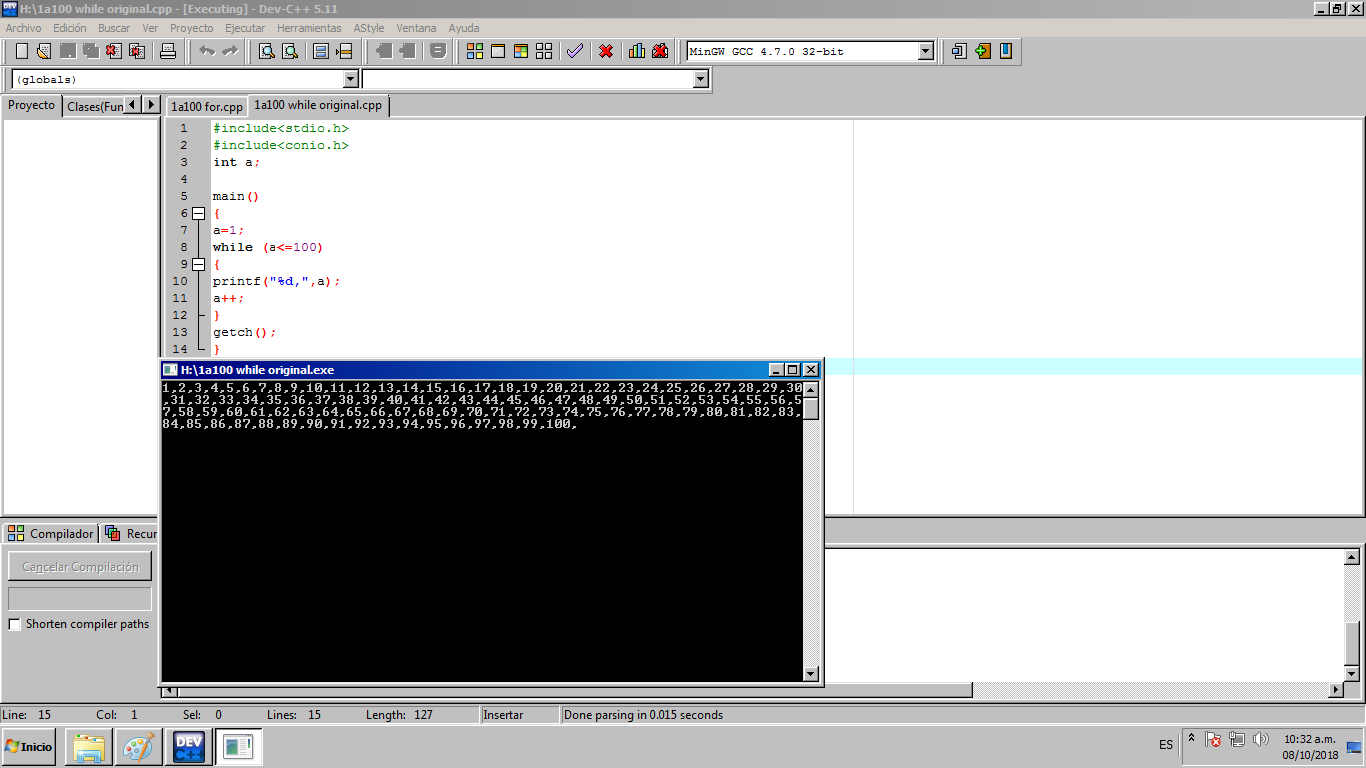
1. **Tablas de multiplicar con “for”**

****

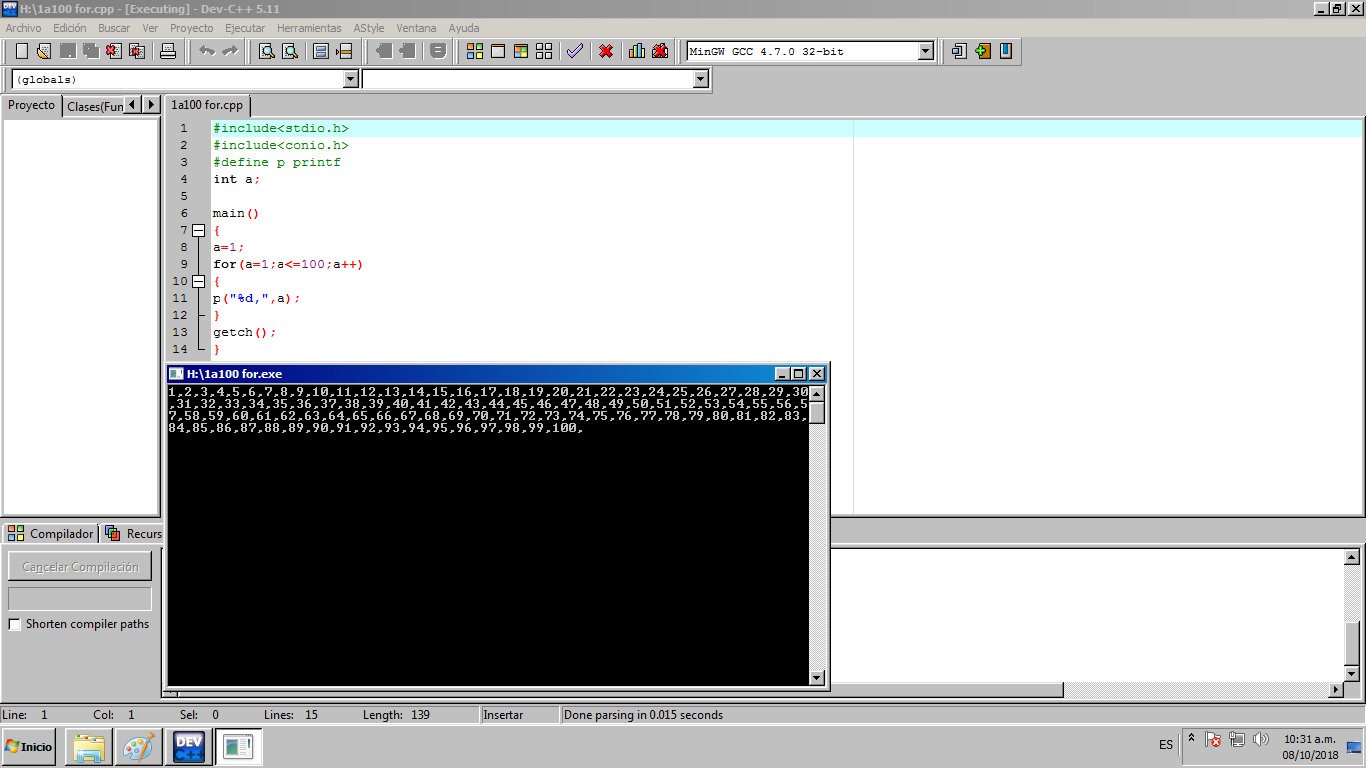
1. **Factorial “Defnine”**



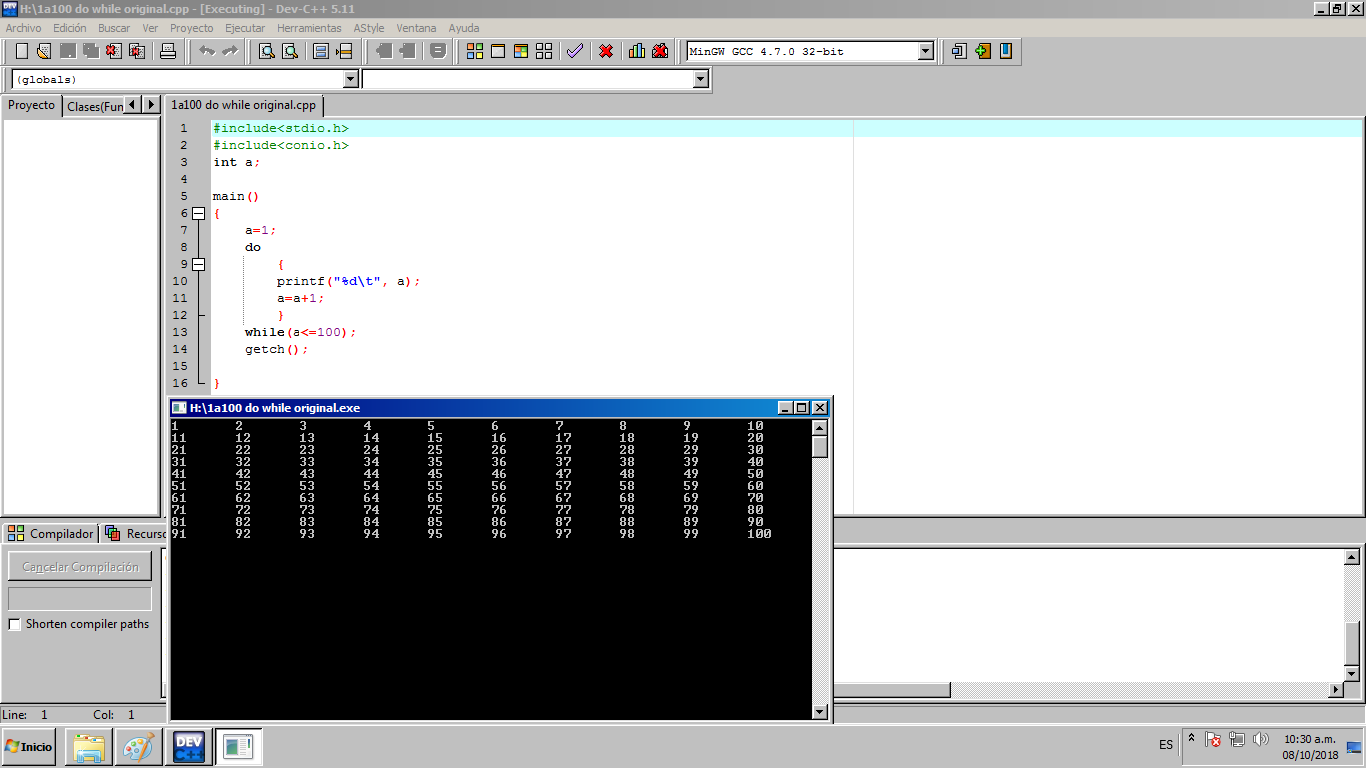
1. **1 a 100 “while”**

****

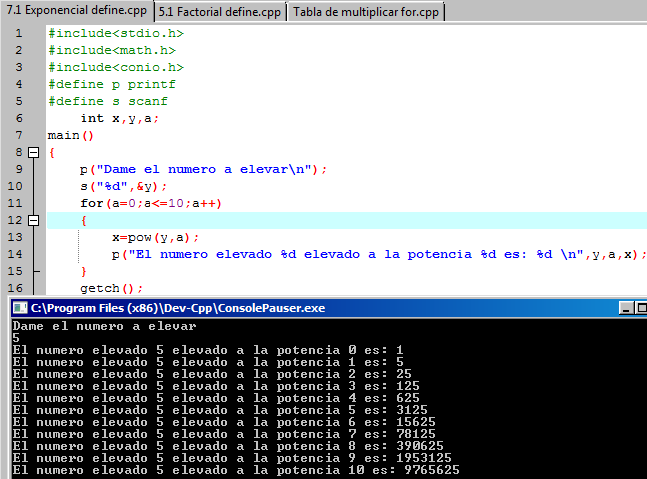
1. **1 a 100 “for”**

****

1. **1 a 100 “do-while”**

****

1. **Potencia cualquiera “for”**

****

**Conclusiones**

Me gustó esta práctica dado que siempre es bueno saber varias formas de como programar en el lenguaje de C, es importante para nuestra carrera encontrar varias formas o soluciones para resolver el problema que nos den con nuestro programa. Me gustó trabajar con estas funciones y ver cómo pueden servir para dar varias alternativas al programador.