|  |
| --- |
| Laboratorio de Computación  Salas A y B |
|  |

|  |  |
| --- | --- |
| Profesor: | GARCIA MORALES KARINA ING. |
| Asignatura: | FUNDAMENTOS DE PROGRAMACION |
| Grupo: | 1121 |
| No de Práctica(s): | 12 |
| Integrante(s): | JOSE DANIEL CALLEJAS SANDOVAL |
|  |  |
|  |  |
|  |  |
| No. de Equipo de cómputo empleado: | 21 |
| Semestre: | 1 |
| Fecha de entrega: | 13-11-2018 |
| Observaciones: |  |
|  |  |

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

Guía práctica de estudio 12: Funciones

**Objetivo:**

Elaborar programas en C donde la solución del problema se divida en funciones. Distinguir lo que es el prototipo o firma de una función y la implementación de ella, así como manipular parámetros tanto en la función principal como en otras.

**Desarrollo:**

Un programa en lenguaje C consiste en una o más funciones. C permite tener dentro de un archivo fuente varias funciones, esto con el fin de dividir las tareas y que sea más fácil la depuración, la mejora y el entendimiento del código. En lenguaje C la función principal se llama main. Cuando se ordena la ejecución del programa, se inicia con la ejecución de las instrucciones que se encuentran dentro de la función main, y ésta puede llamar a ejecutar otras funciones, que a su vez éstas pueden llamar a ejecutar a otras funciones, y así sucesivamente.

Funciones: La sintaxis básica para definir una función es la siguiente: valorRetorno nombre (parámetros){// bloque de código de la función } El nombre de la función se refiere al identificador con el cual se ejecutará la función; se debe seguir la notación de camello.

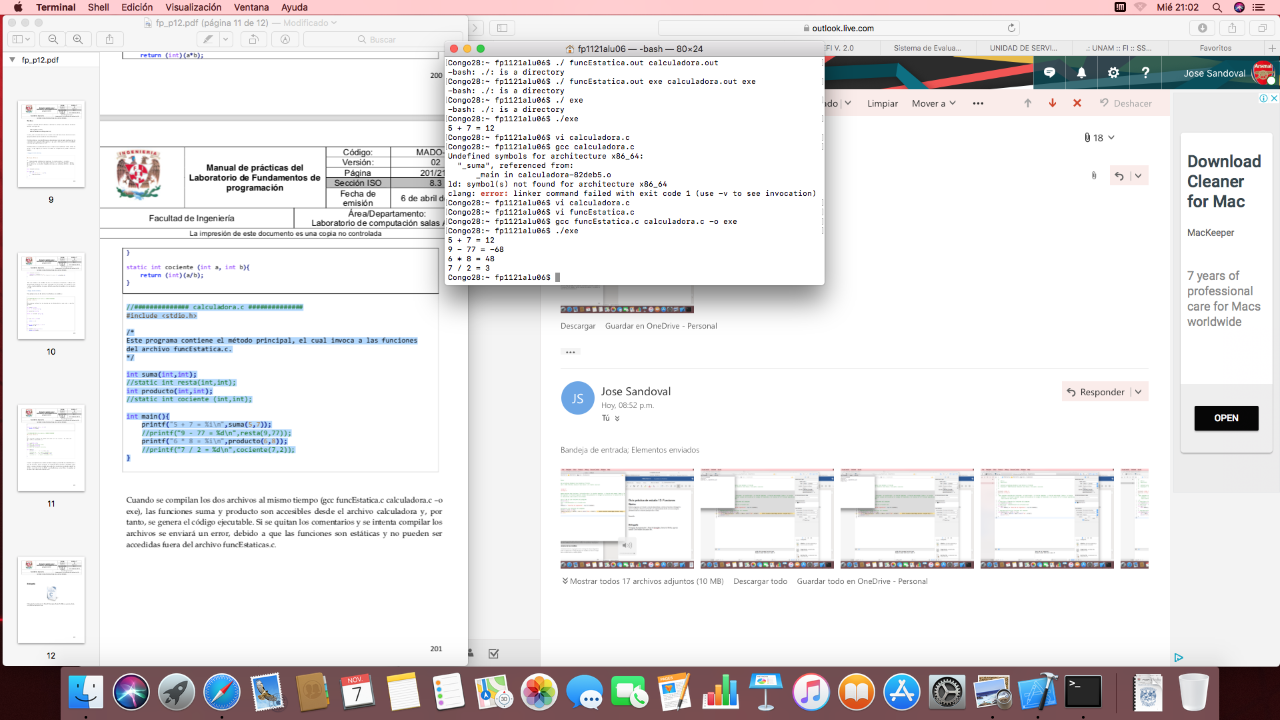
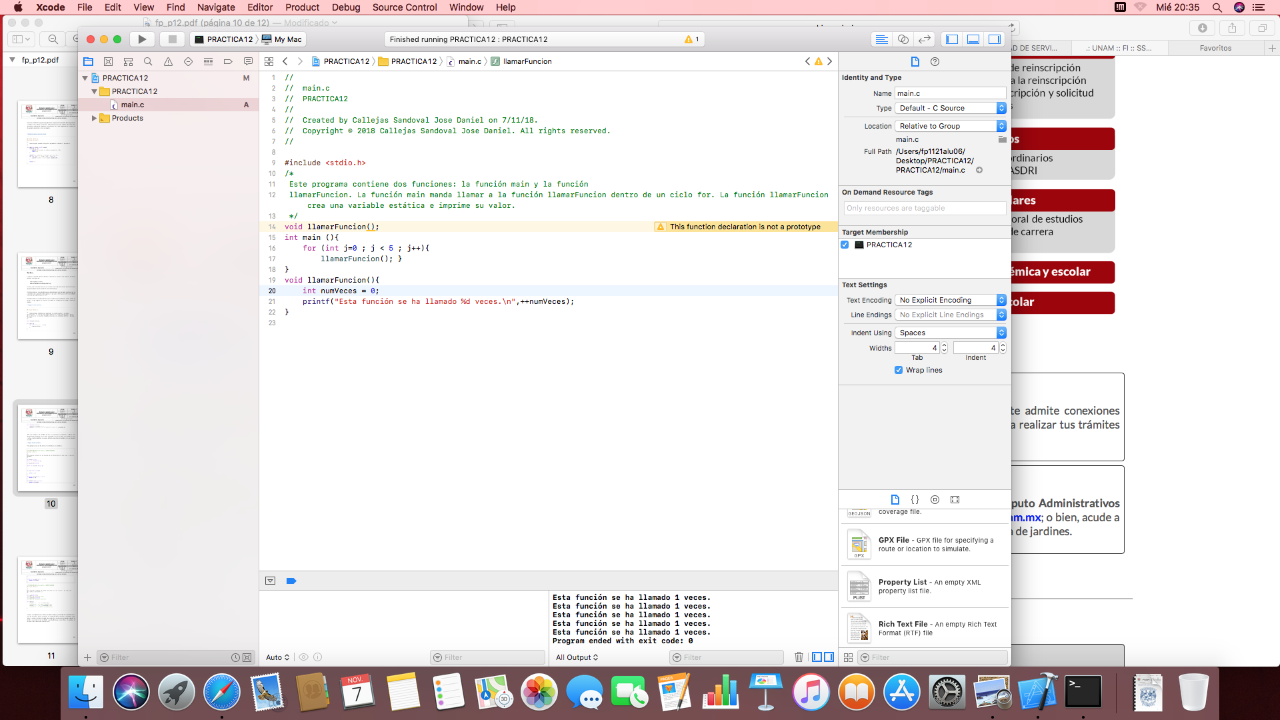
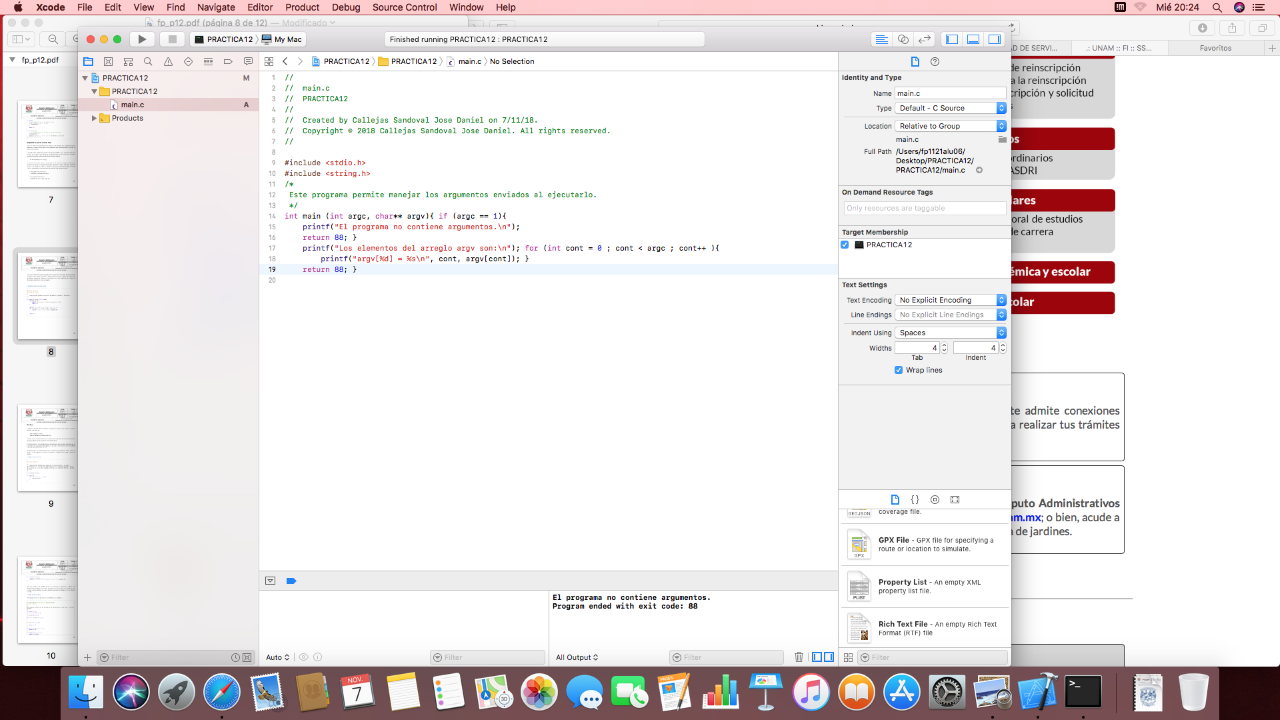
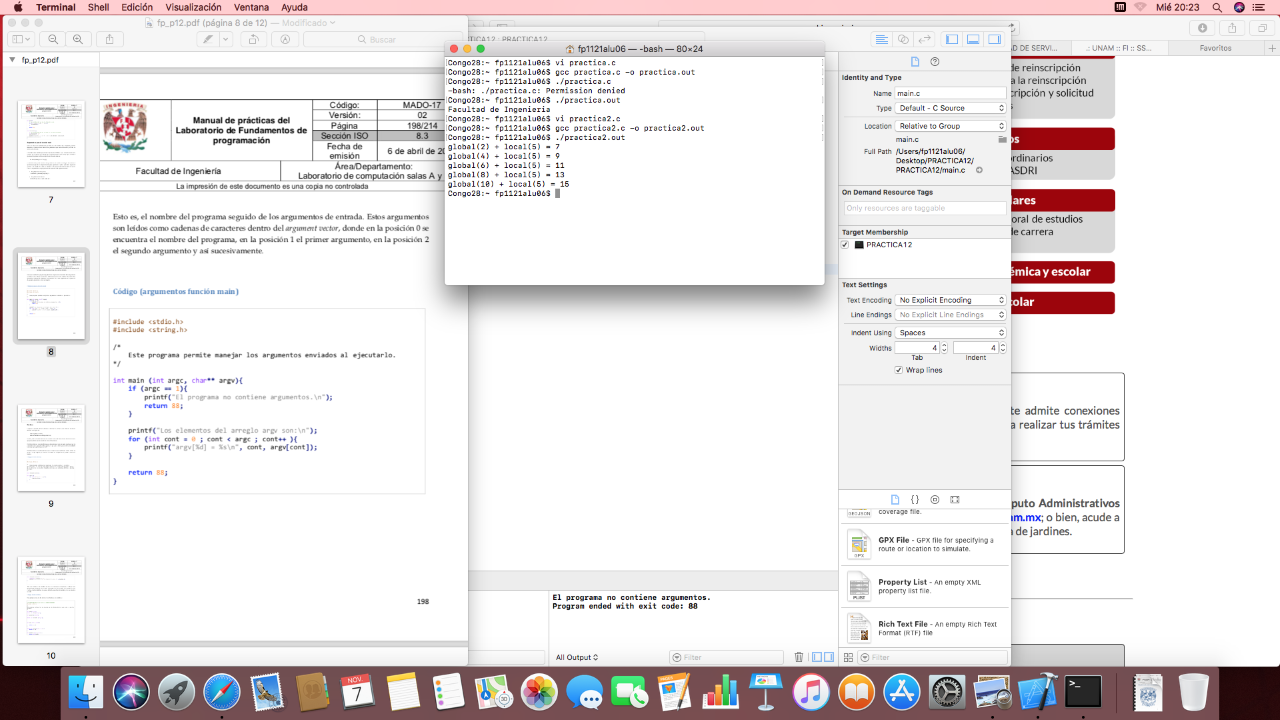
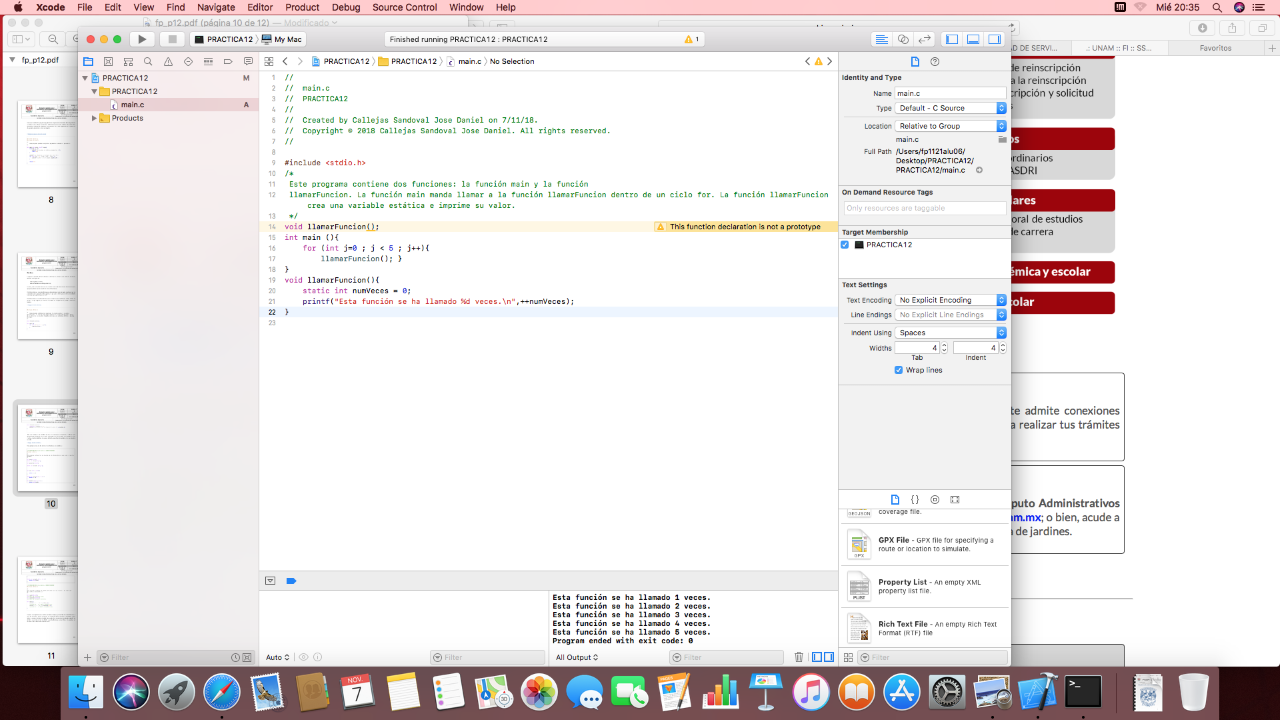
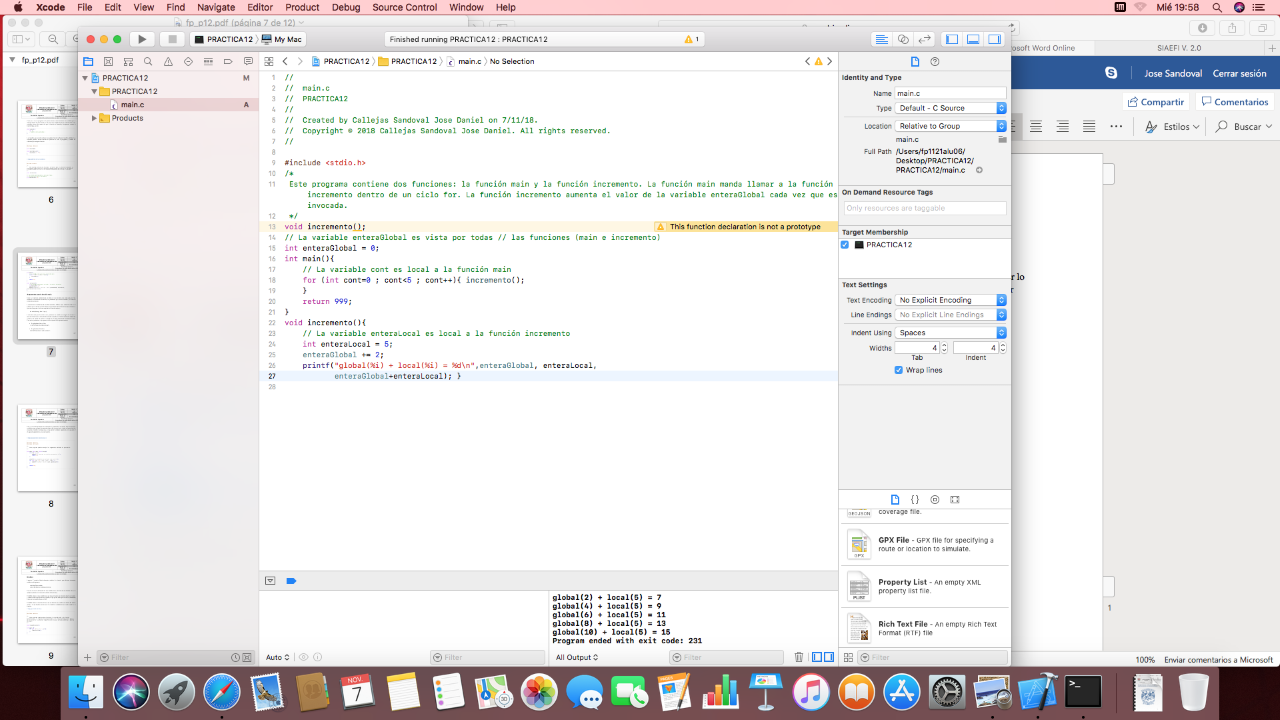
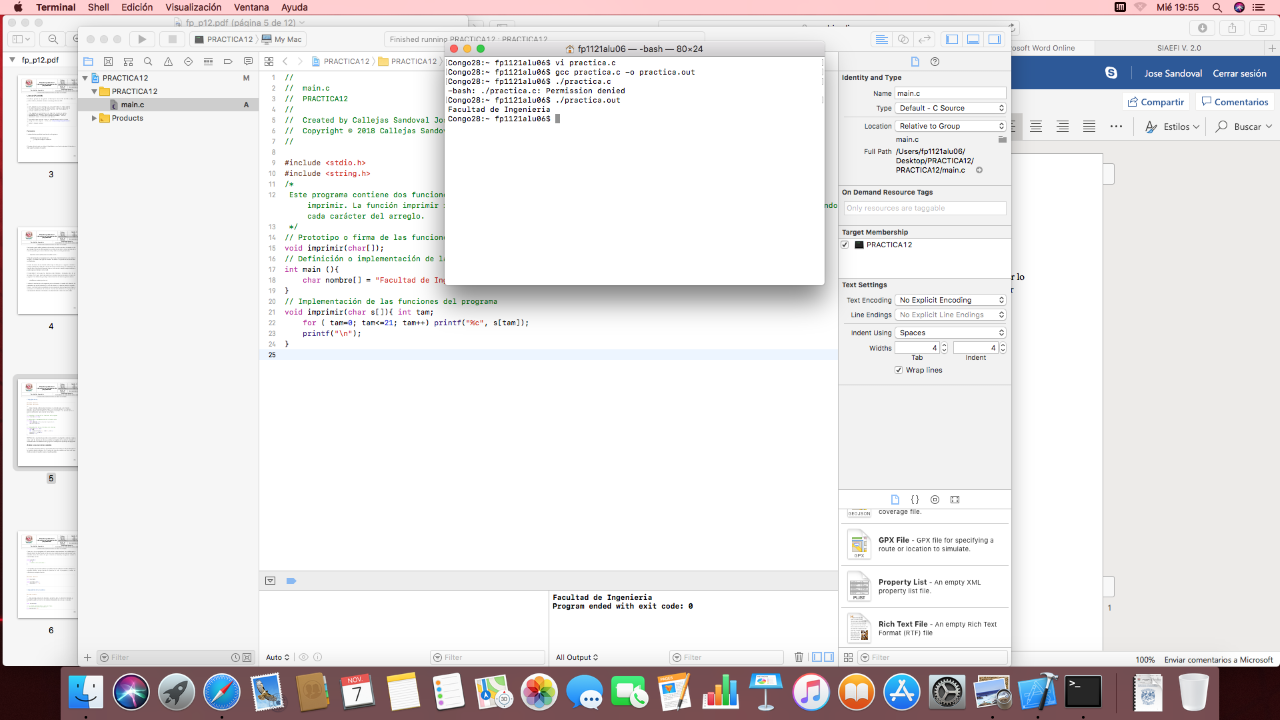
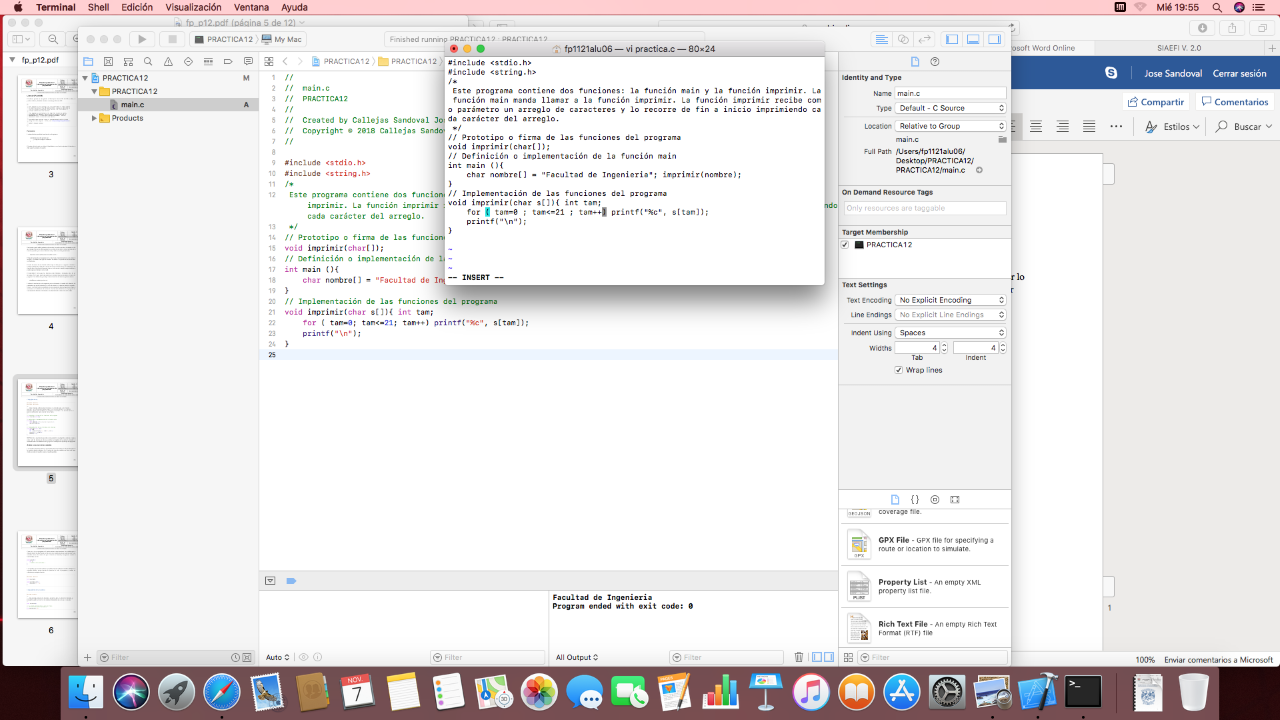
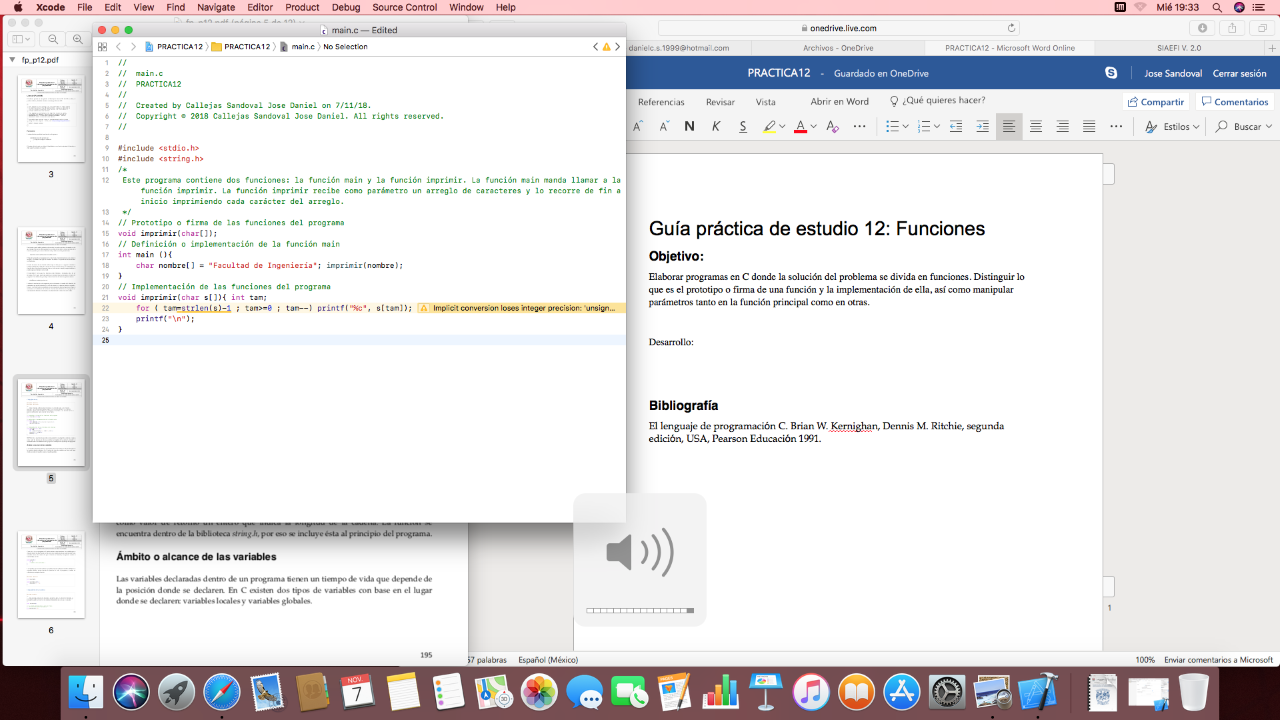
Una función puede recibir parámetros de entrada, los cuales son datos de entrada con los que trabajará la función, dichos parámetros se deben definir dentro de los paréntesis de la función, separados por comas e indicando su tipo de dato, de la siguiente forma: (tipoDato nom1, tipoDato nom2, tipoDato nom3…).

El compilador C revisa que las funciones estén definidas o declaradas antes de ser invocadas. Por lo que una buena práctica es declarar todas las funciones al inicio del programa. Una declaración, prototipo o firma de una función tiene la siguiente sintaxis: valorRetorno nombre (parámetros);

La firma de una función está compuesta por tres elementos: el nombre de la función, los parámetros que recibe la función y el valor de retorno de la función; finaliza con punto y coma (;)

Estático: Lenguaje C permite definir elementos estáticos. La sintaxis para declarar elementos estáticos es la siguiente: static tipoDato nombre; static valorRetorno nombre(parámetros); Es decir, tanto a la declaración de una variable como a la firma de una función solo se le agrega la palabra reservada static al inicio de las mismas. El atributo static en una variable hace que ésta permanezca en memoria desde su creación y durante toda la ejecución del programa, lo que quiere decir que su valor se mantendrá hasta que el programa llegue a su fin.

Una vez declarada una variable estática, esta permanece en memoria a lo largo de la ejecución del programa, por lo tanto, la segunda vez que se llama a la función ya no se vuelve a crear la variable, si no que se utiliza la que está en la memoria y por eso conserva su valor.

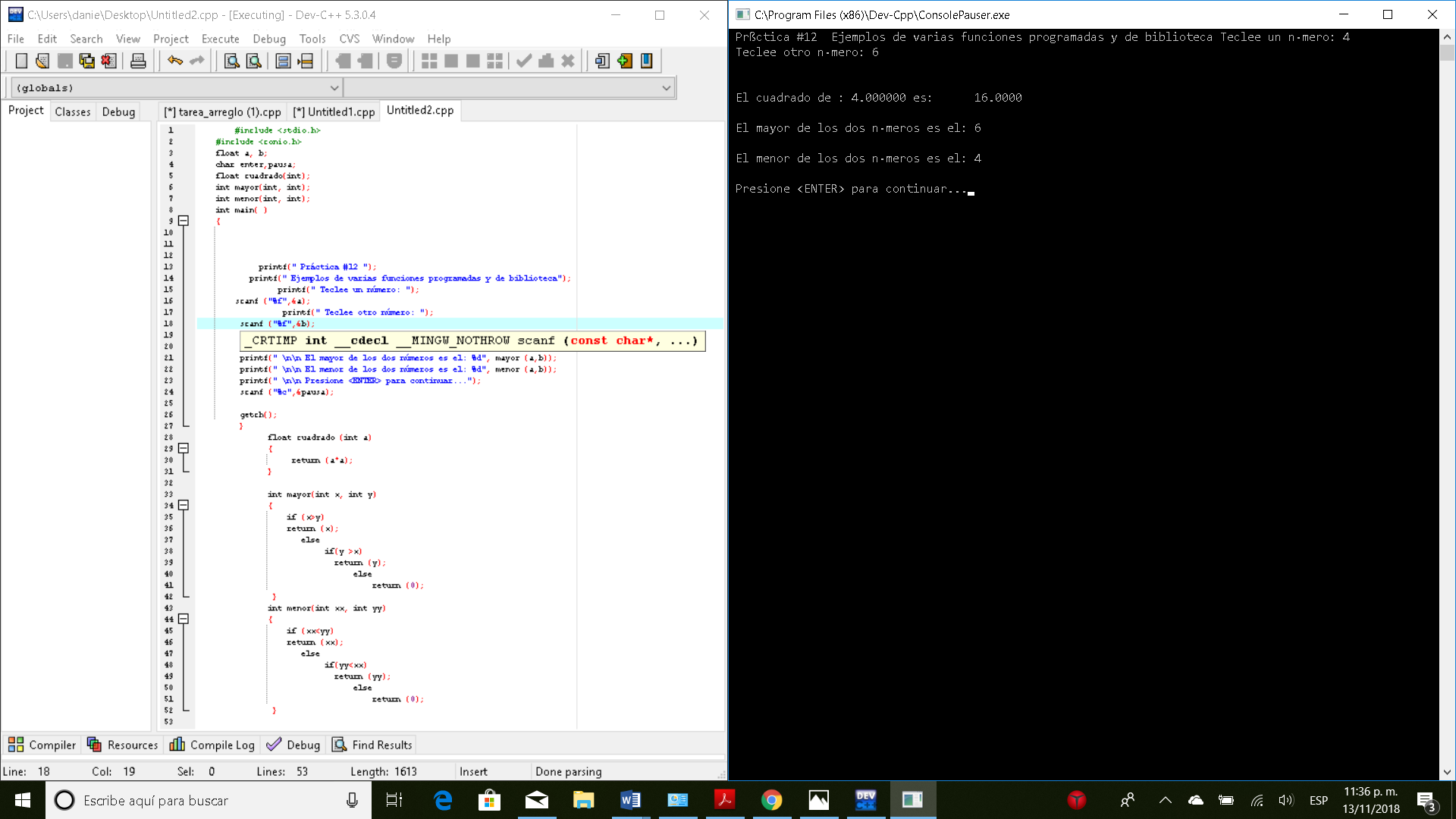


**Ejercicios de tarea**

**1.-**

a) los errores que obtuvo (en caso de haberlos): La mayoría de los errores tenían que ver con argumentos que no fueron declarados y que además el main debía de ser “int”.

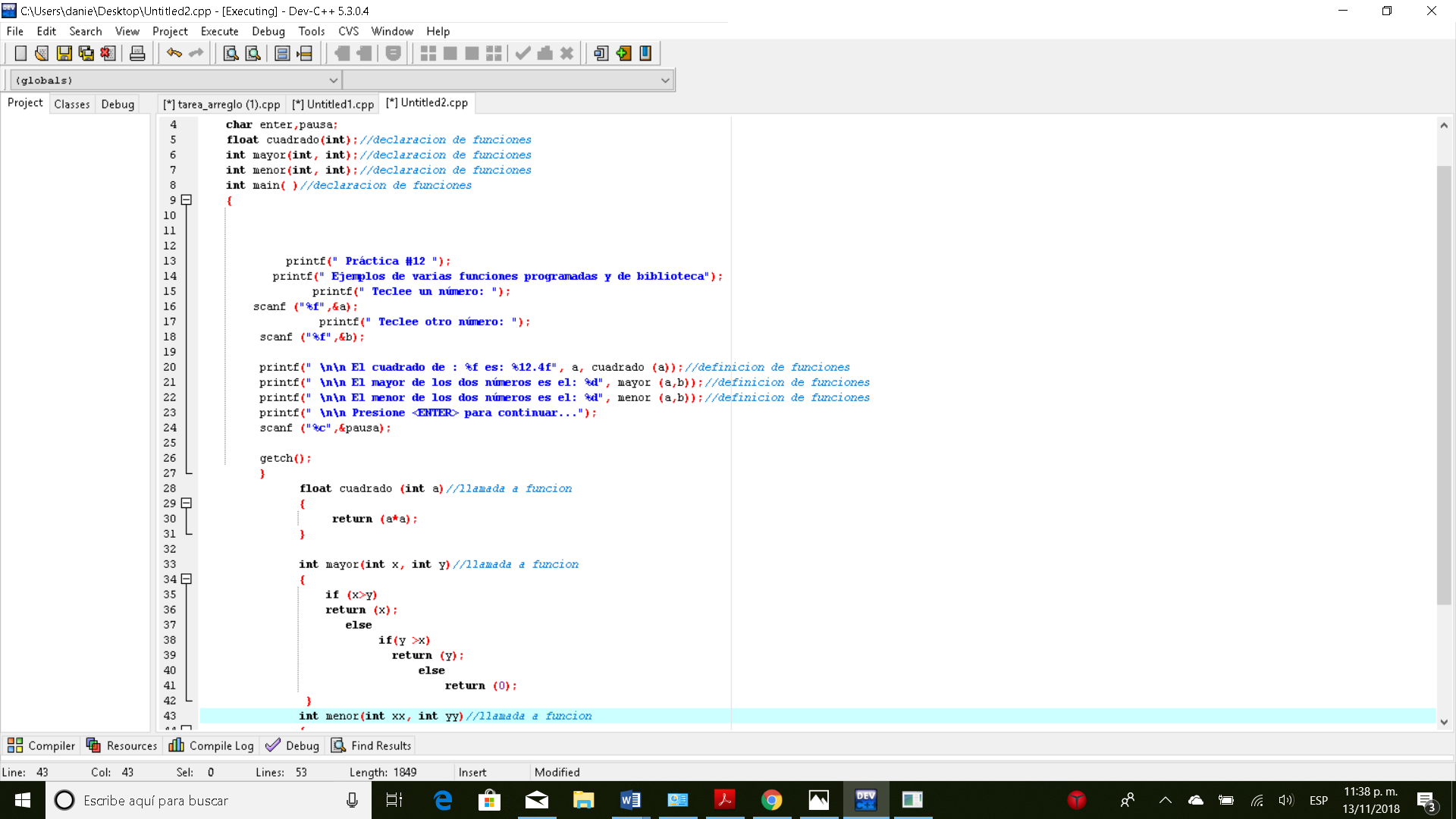
b) el desplegado obtenido:



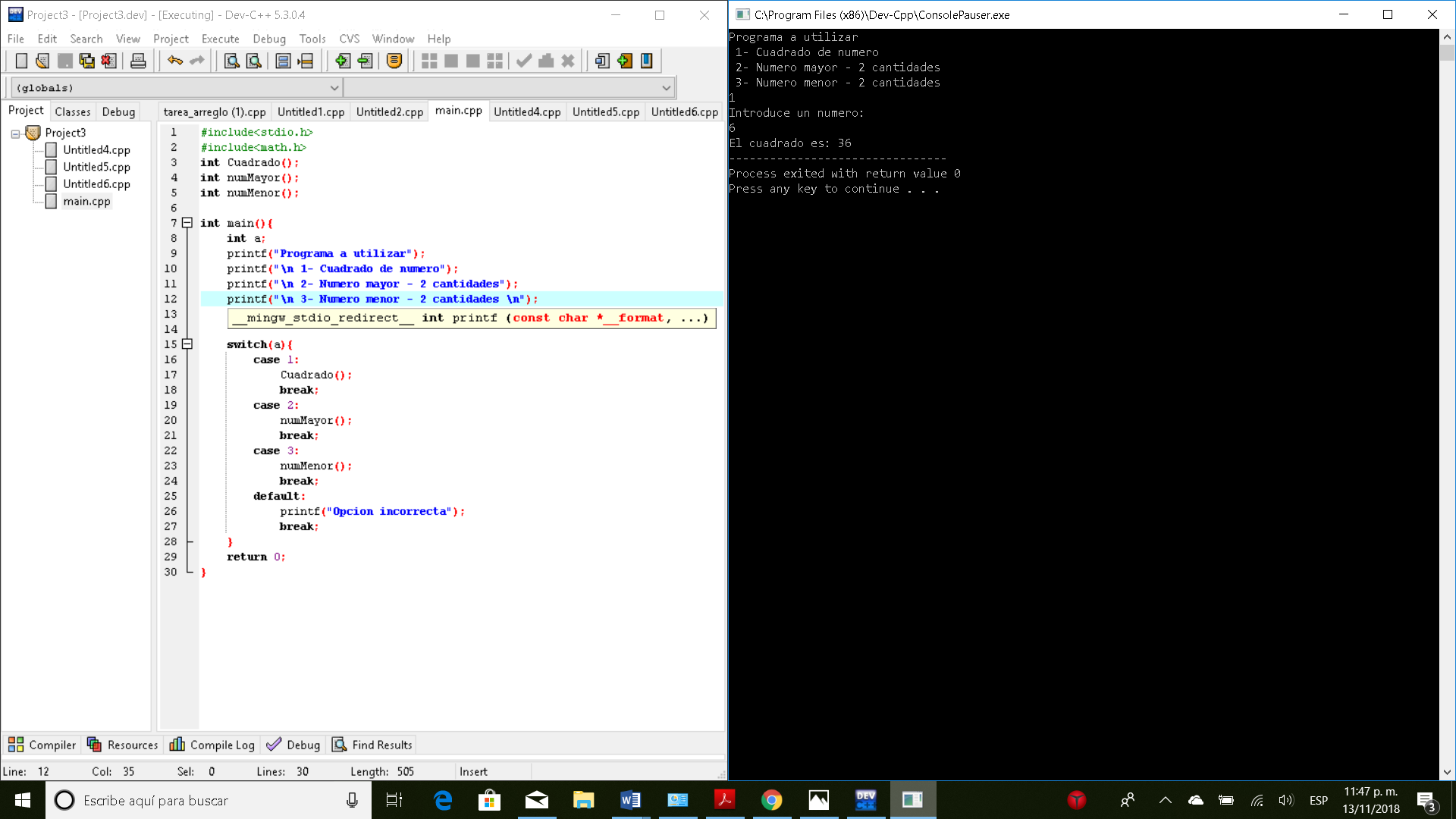
a) las llamadas a las funciones

b) la declaración de las funciones

c) la definición de las funciones:



Genere el programa como un proyecto y separe las funciones, una por archivo.



**2.-**

**Conclusiones:**

Al finalizar la práctica, pude comprender a fondos todos los elementos de las funciones y que es lo que hace cada uno.

Los ejercicios propuestos, me permitieron también unir el tema de depuración para poder optimizar el programa.

El ritmo de la práctica fue el adecuado para poder analizar los distintos programas mostrados.

**Bibliografía:**

El lenguaje de programación C. Brian W. Kernighan, Dennis M. Ritchie, segunda edición, USA, Pearson Educación 1991.