

Contenido

[**Objetivo.** 3](#_Toc528098210)

[**Actividades.** 3](#_Toc528098211)

[**Introducción.** 3](#_Toc528098212)

[**Estructura de control repetitiva “while”.** 3](#_Toc528098213)

[**Estructura de control repetitiva do-while.** 4](#_Toc528098214)

[**Estructura de control de repetición “for”.** 4](#_Toc528098215)

[2.- Calibrar el ejercicio de la calculadora, pagina 5 y agregar la pregunta de la opción si desea calcular o no (do{}while (op == 'S' || op == 's');). 13](#_Toc528098216)

[2.- Explica que sucede con el ciclo while al colocarle un valor positivo, negativo y cero y porque razón se obtiene ese resultado. 14](#_Toc528098217)

[3.- Que sucede con el ejercicio que ejecuta el break, si se omite y que sucede si se coloca? 14](#_Toc528098218)

[4.- Que sucede con el ejercicio que ejecuta continue, si se omite y que sucede si se coloca? 14](#_Toc528098219)

[**Conclusiones.** 14](#_Toc528098220)

[**Bibliografías.** 14](#_Toc528098221)

**Objetivo.**

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

**Actividades.**

* Elaborar un programa que utilice la estructura while en la solución de un problema.
* Elaborar un programa que requiera el uso de la estructura do-while para resolver un problema. Hacer la comparación con el programa anterior para distinguir las diferencias de operación entre while y do-while.
* Resolver un problema dado por el profesor que utilice la estructura for en lugar de la estructura while.
* Usar la directiva define para elaboración de código versátil.

**Introducción.**

Las estructuras de repetición son 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 “while”, como ya se dijo es una estructura de repetición que valida si la expresión lógica y si solo si es verdadero se procede a ejecutar el bloque de instrucciones de la estructura; se define por las llaves { }.

En caso de que la expresión lógica no se cumpla el flujo sigue de manera normal, ósea se puede realizar el siclo de 0 a n veces.

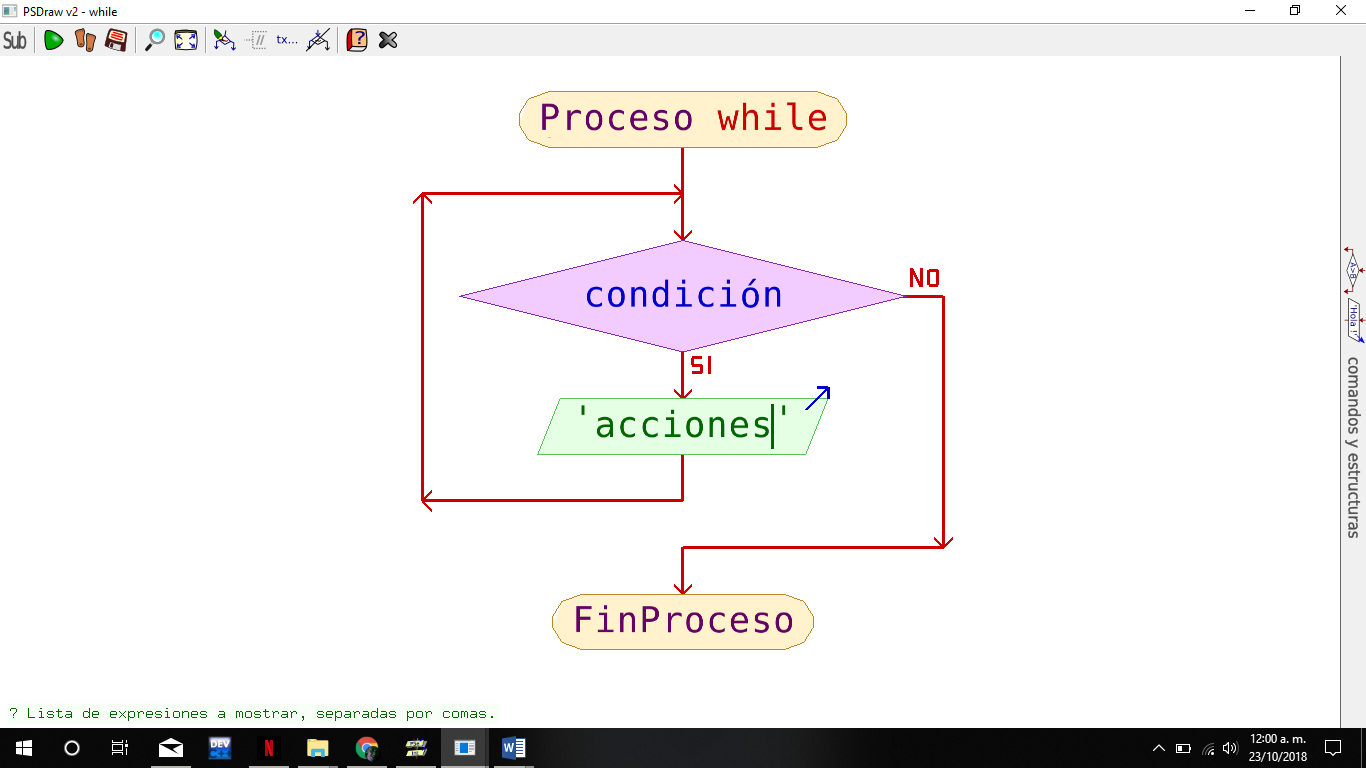
La forma general de la instrucción while es:

While(expresión)

{

Instrucción o instrucciones

}



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

En la instrucción while que acabamos de ver la condición se coloca al principio del ciclo. En el ciclo do-while la expresión se pone al final. De esta forma se garantiza que las instrucciones del ciclo se ejecutan por lo menos una vez antes de pasar por la expresión.

La forma general de la instrucción do-while es:

do

{

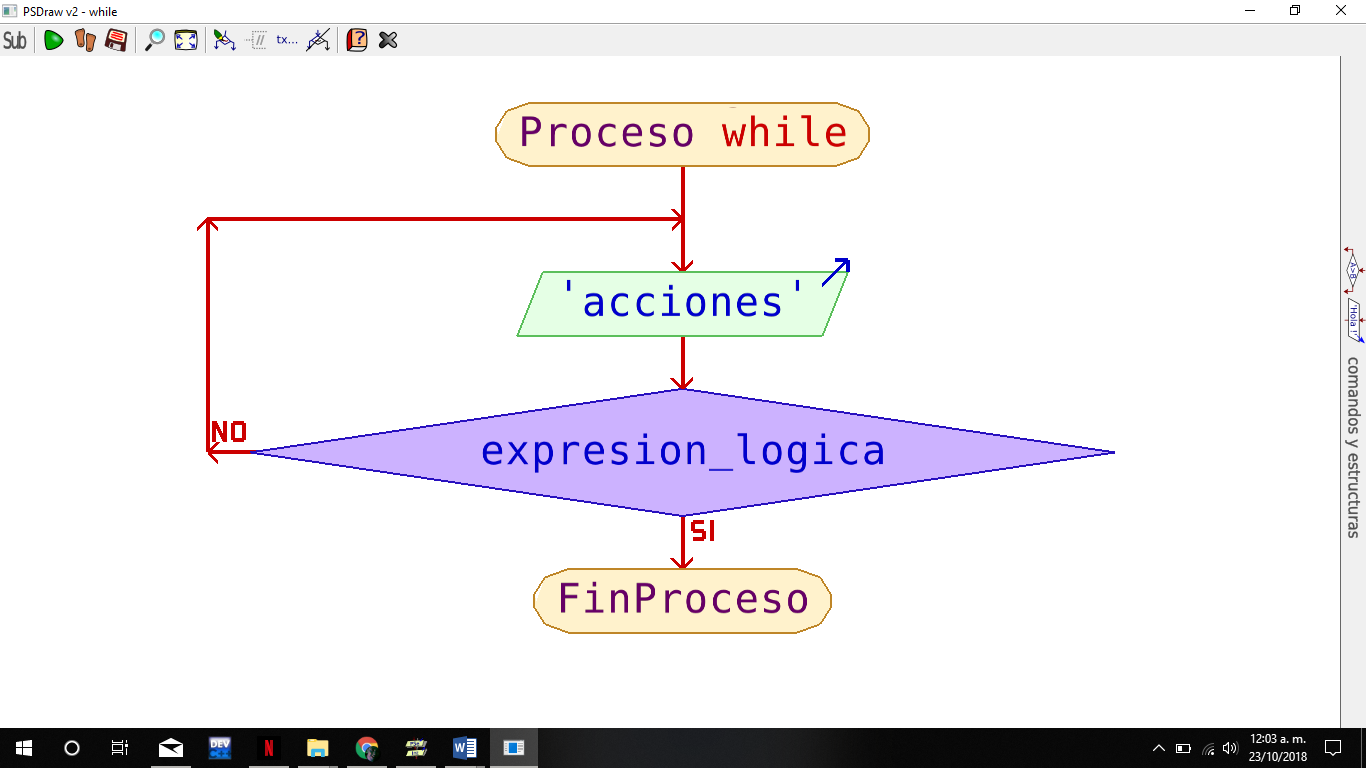
Instrucciones;

}

while (expresión);

En el ciclo do-while las instrucciones se ejecutan una vez y luego se evalúa la condición para determinar si se continua o no con el ciclo.

Es importante hacer notar que la instrucción do-while termina con un punto y coma.



Como se puede observar las instrucciones se ejecutan primero puesto que la instrucción while está al final; el printf imprime en la pantalla el valor y después se incrementa la variable. Se verifica la condición y como después del incremento el valor de la variable es 66 se sigue cumpliendo la condición y el programa vuelve a iniciarse desde el do, y así sucesivamente hasta que la condición sea mayor a 80.

**Nota:** El incremento que vemos como x++; es igual que x = x+1;

**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 de la variable; condición; incremento)

{

INSTRUCCIONES

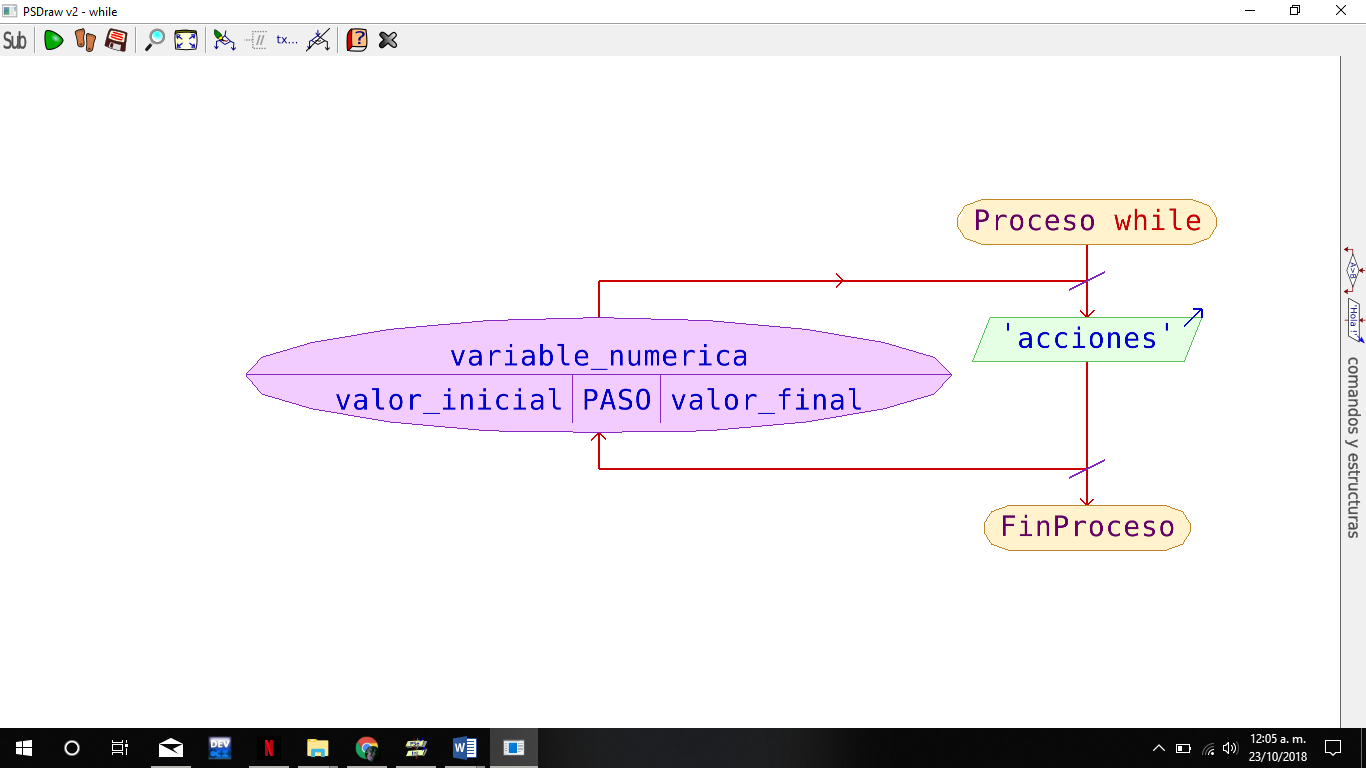
}

Ejemplo:

for (i=0; i<=10; i++)

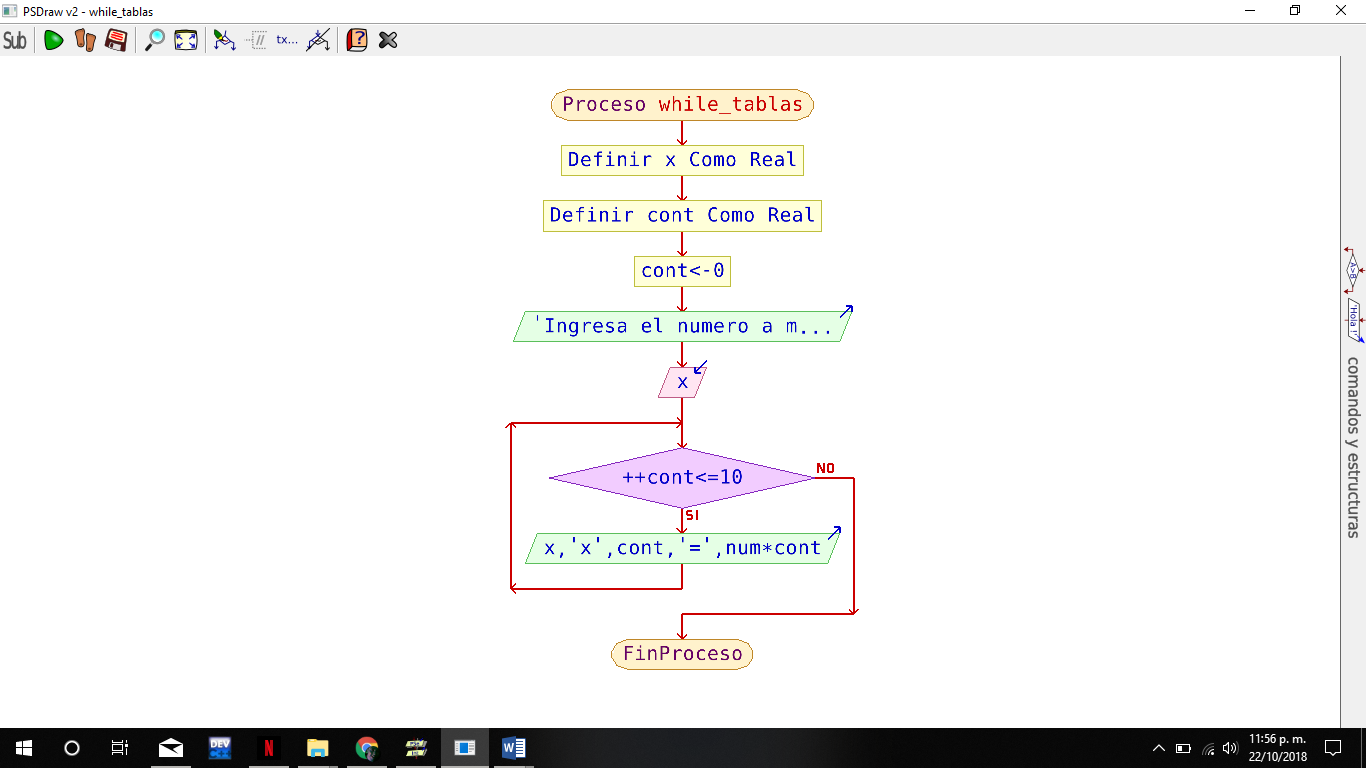
NOTA: se finaliza todo lo que realizara el ciclo cerrando con otra llave (}), después que finalice ese ciclo el programa continuara con lo que esta después de } que marca el fin del ciclo.

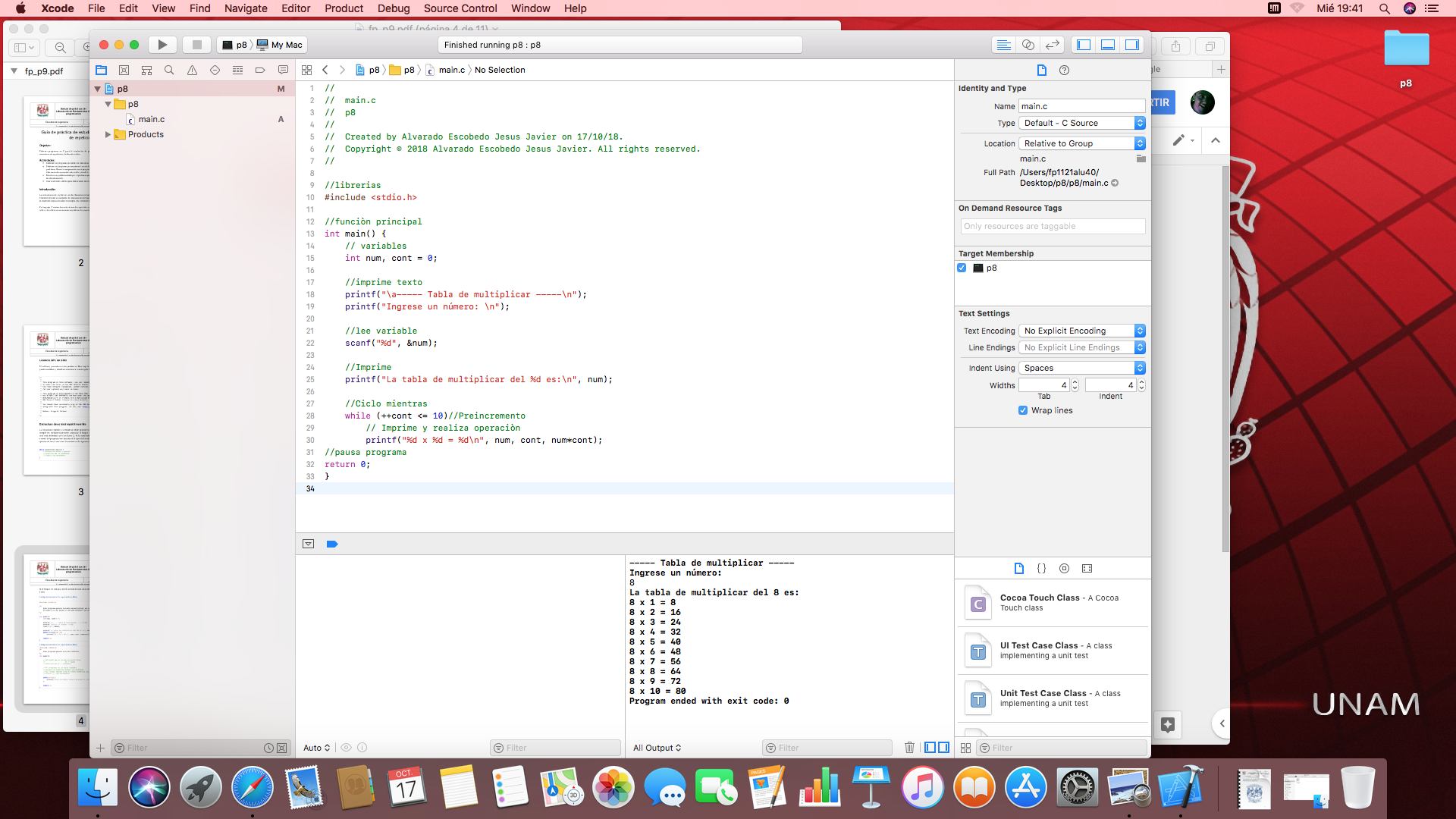
La estructura for ejecuta 3 acciones básicas antes o después de ejecutar el bloque de código.



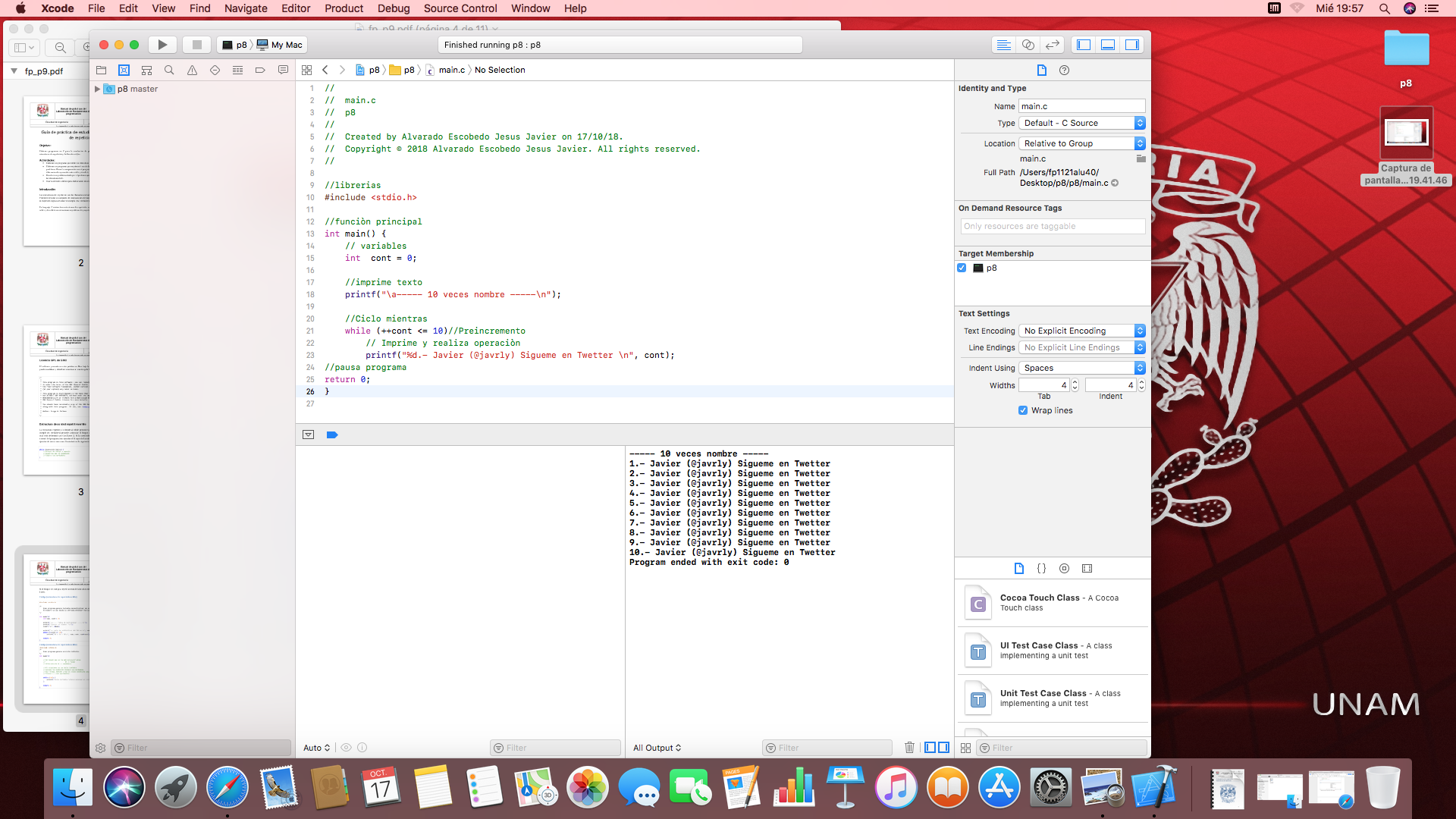
La primera acción es la inicialización, en la cual se pueden definir variables e inicializar sus valores; esta parte solo se ejecuta una vez cuando se ingresa al ciclo y es opcional. La segunda acción consta de una expresión lógica, la cual se evalúa y, si ésta es verdadera, ejecuta el bloque de código, si no se cumple se continúa la ejecución del programa; esta parte es opcional. La tercera parte consta de un conjunto de operaciones que se realizan cada vez que termina de ejecutarse el bloque de código y antes de volver a validar la expresión lógica; esta parte también es opcional.

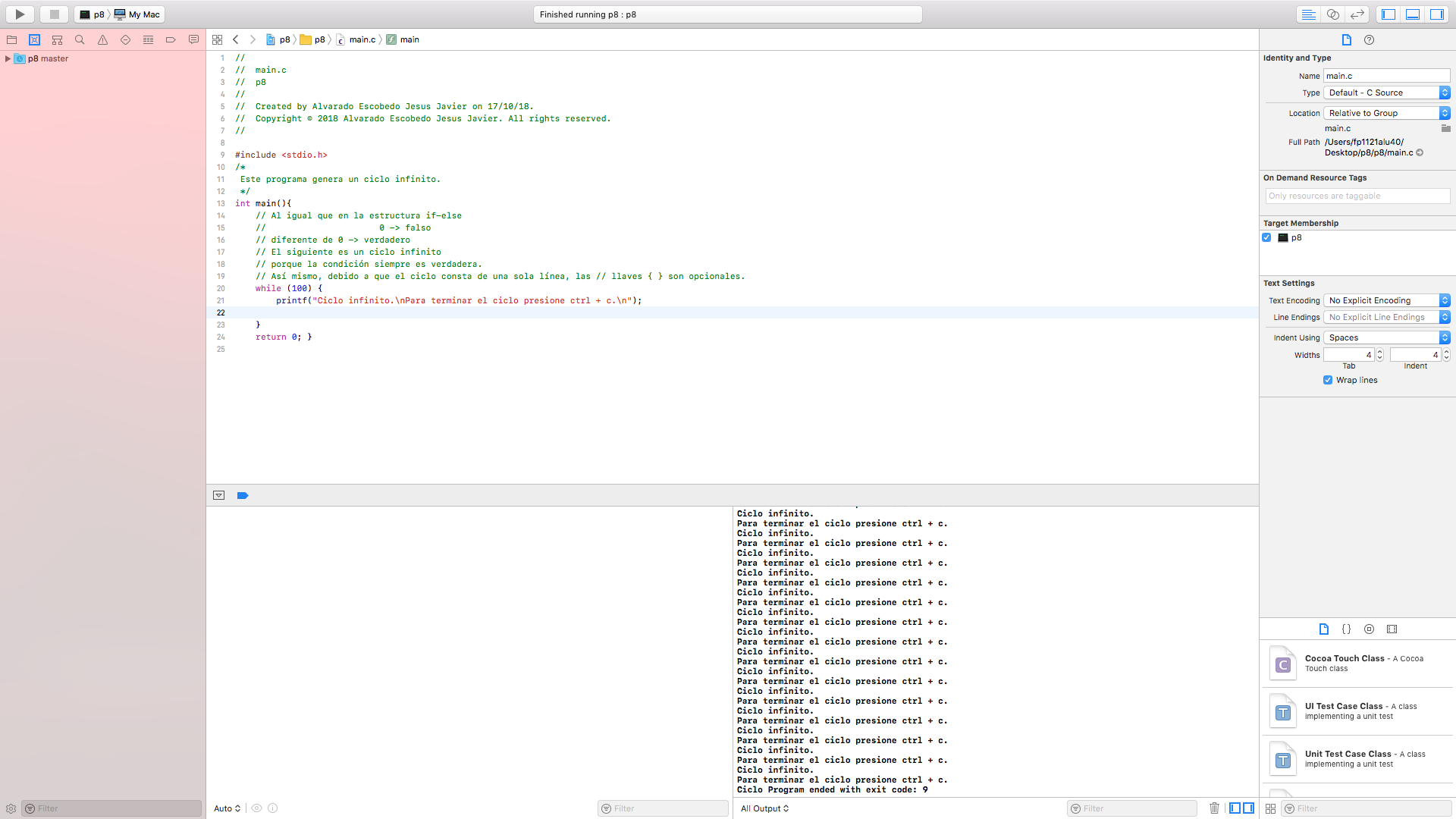
Programa con while

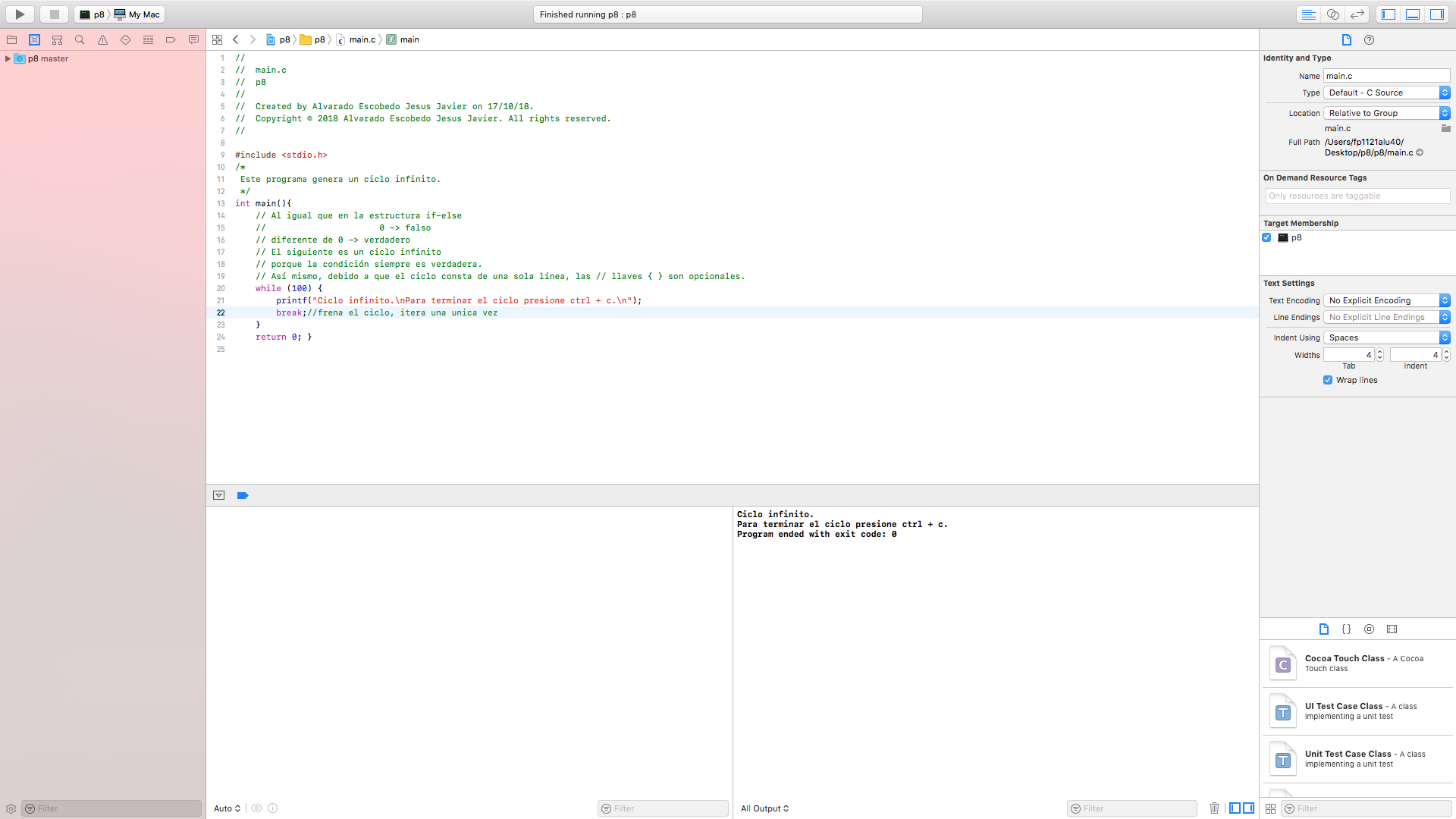




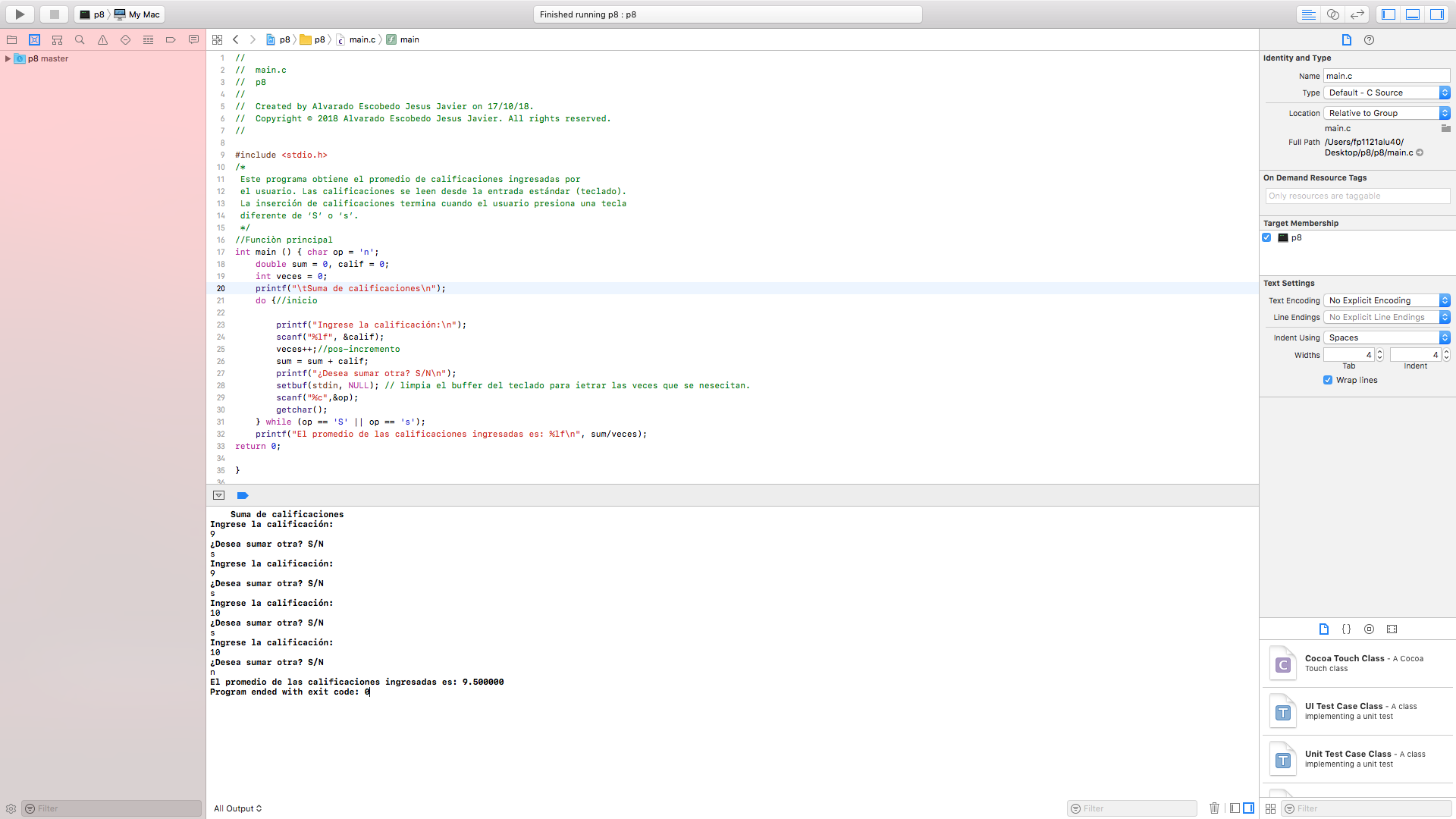
Programa que imprima 10 veces nombre y mostrar el numero.

Ciclo infinito



Uso de break;

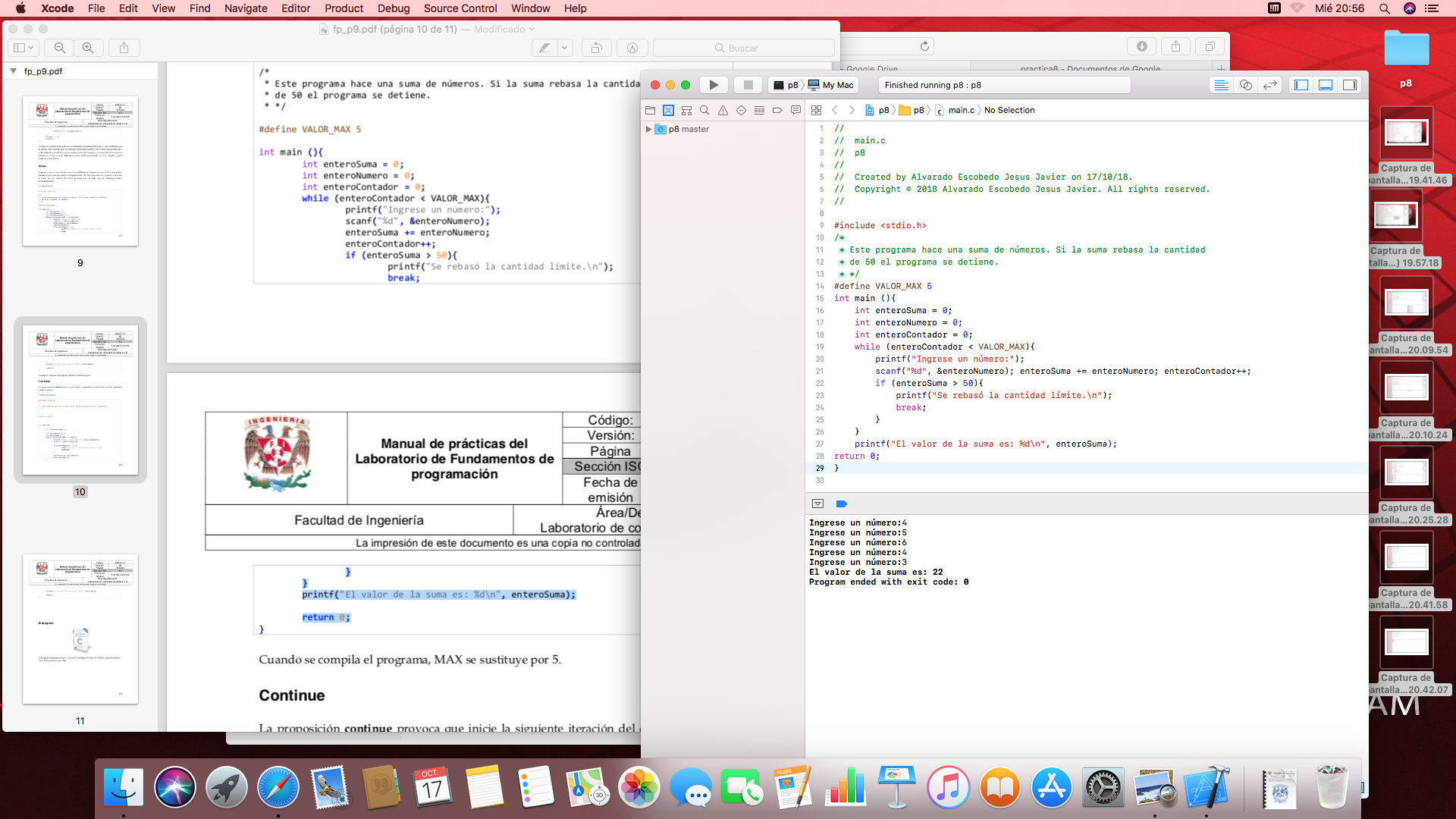
do-while



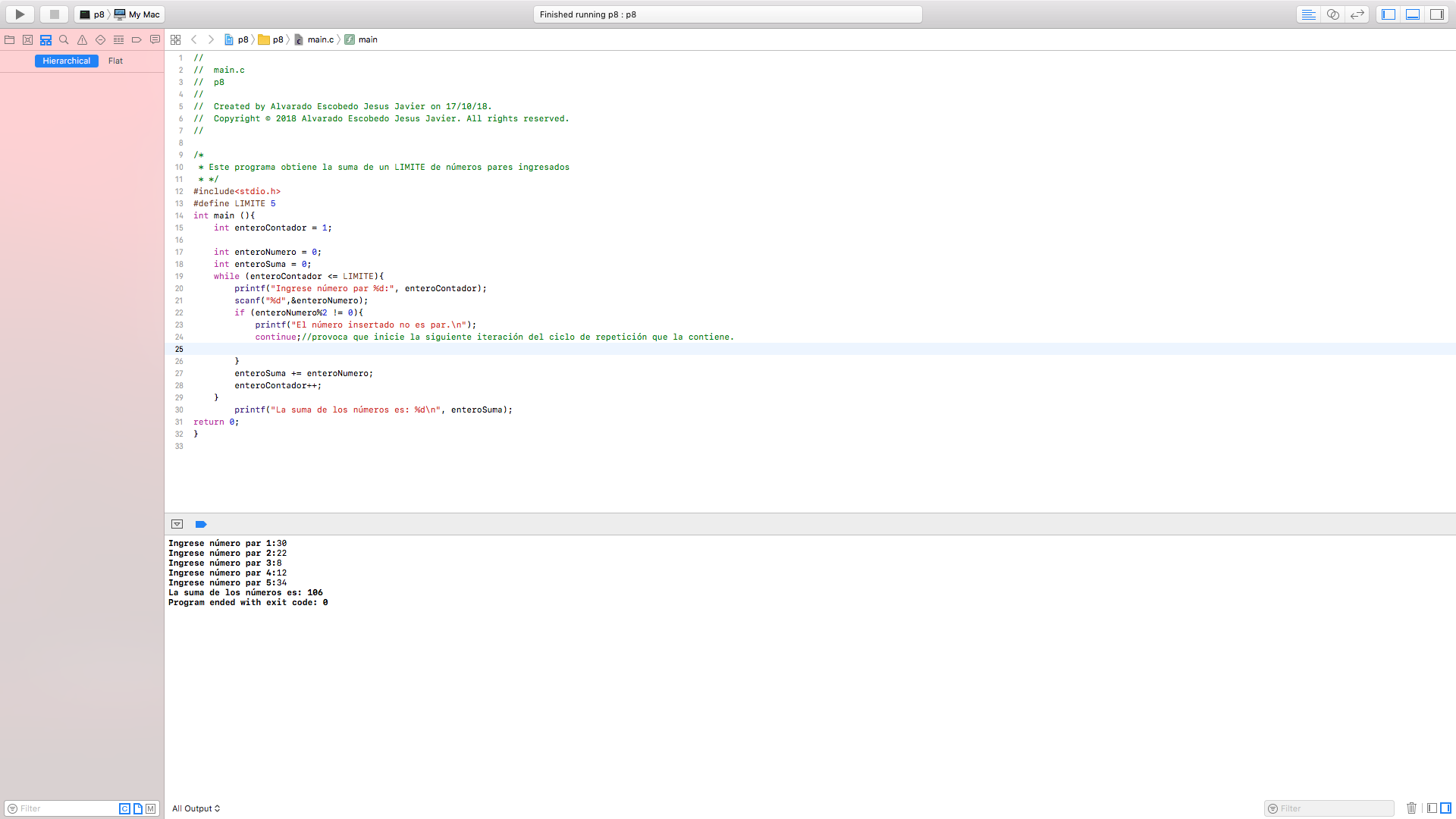
ciclo for



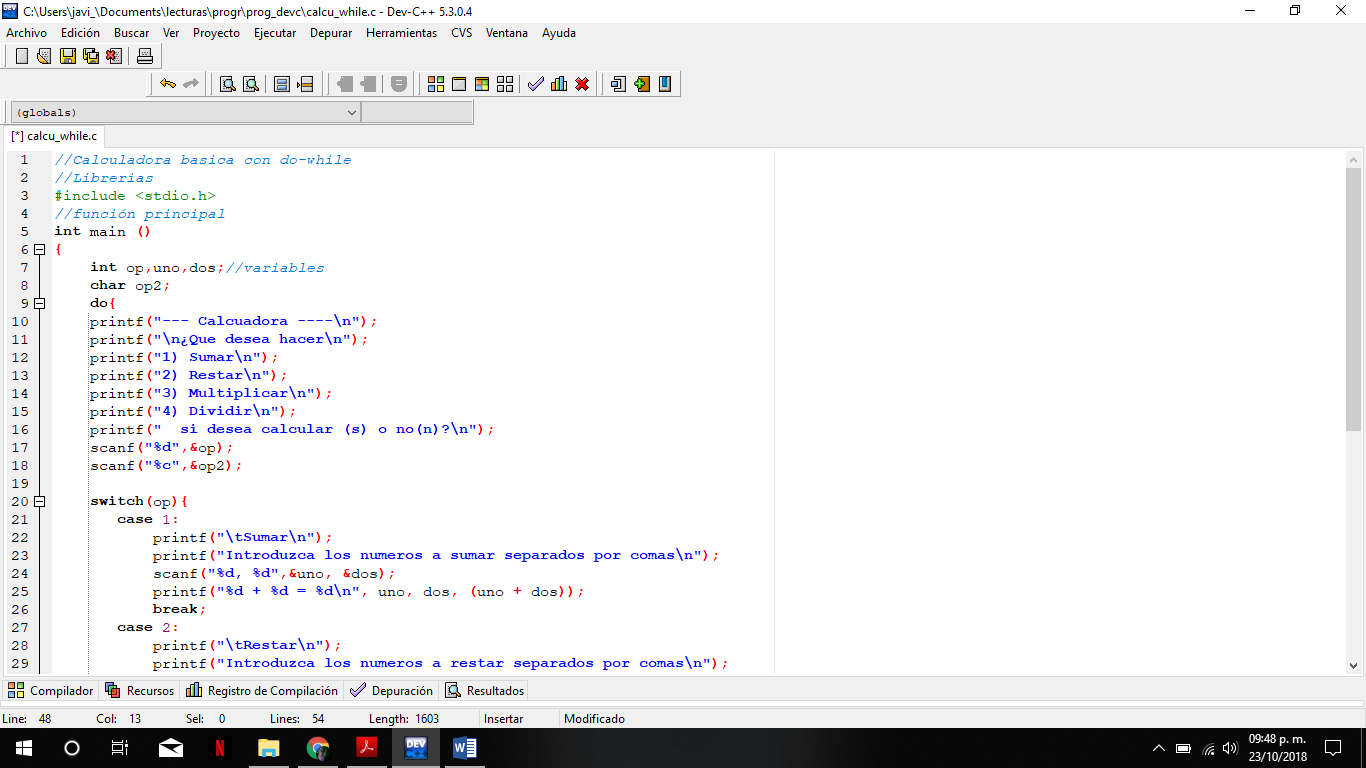
Codigo define

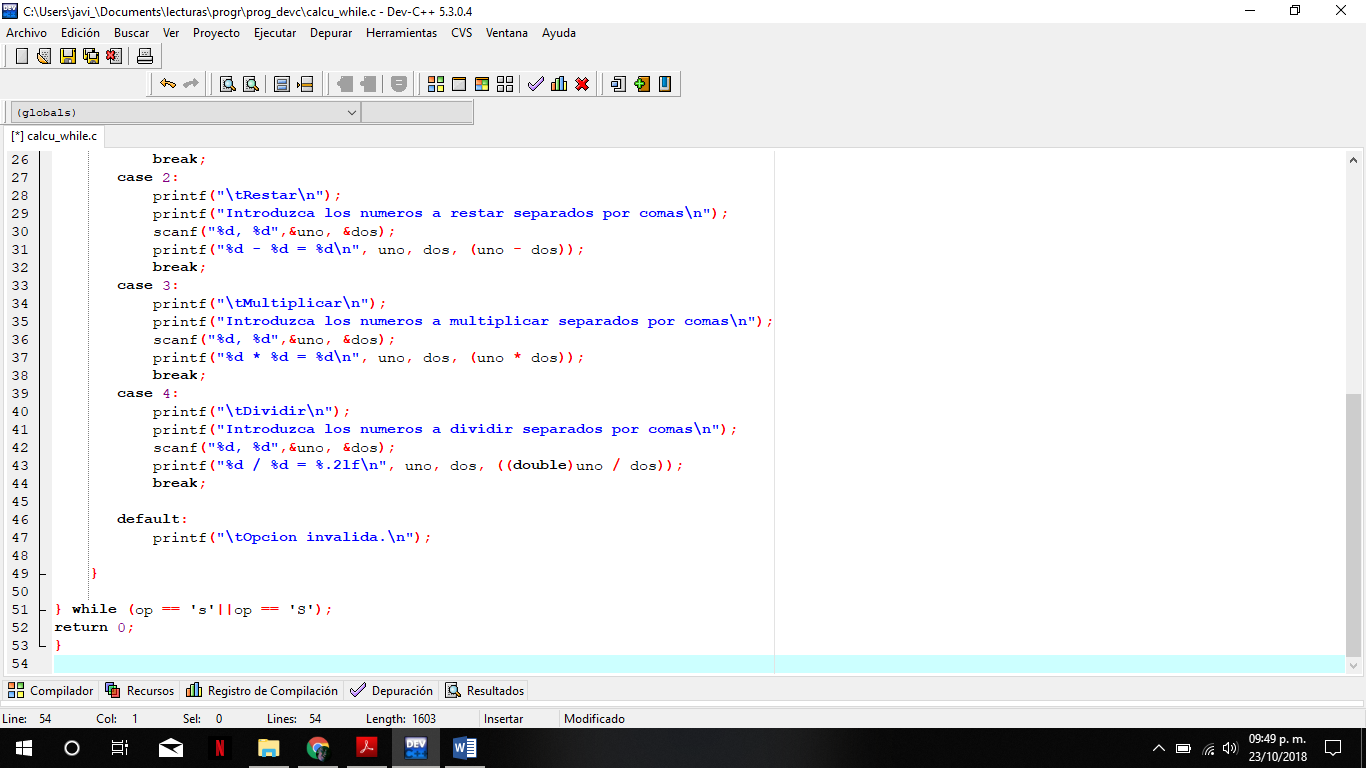


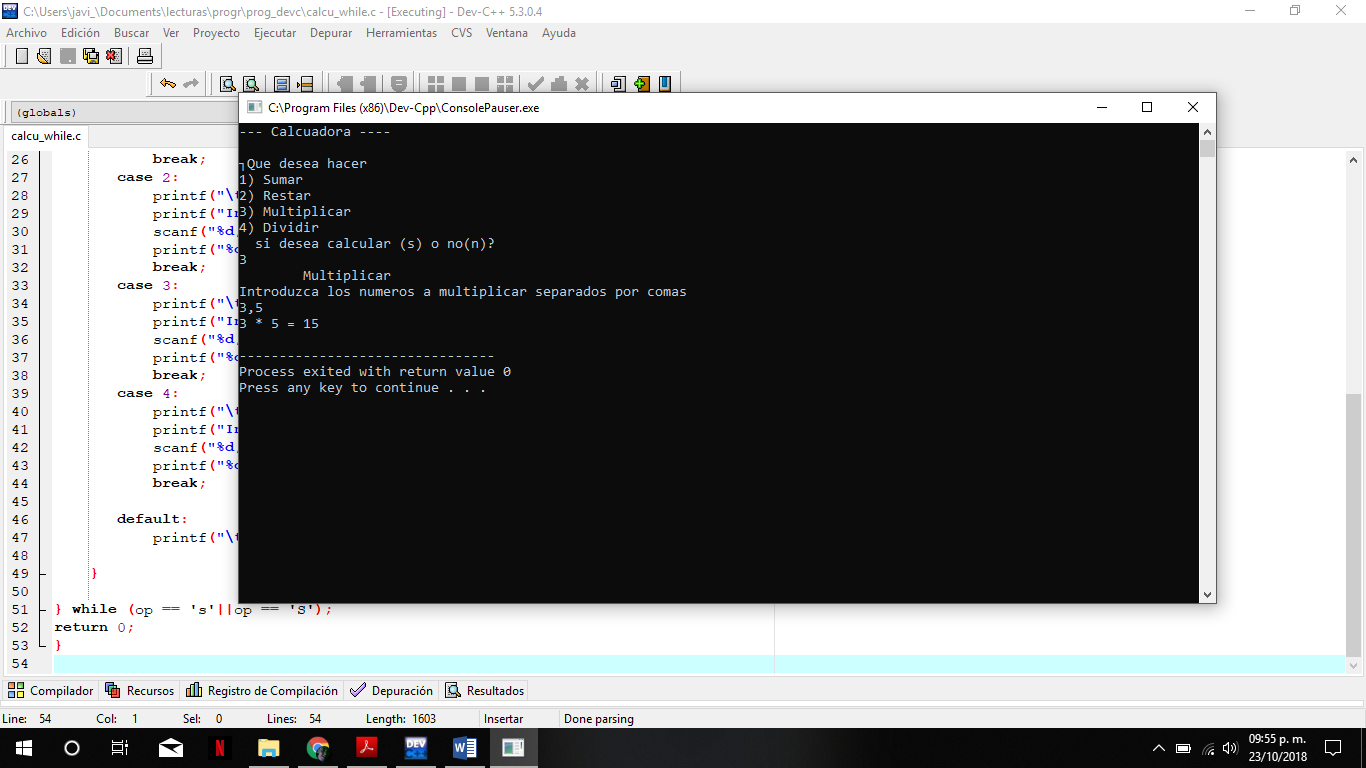
Codigo continue



# 2.- Calibrar el ejercicio de la calculadora, pagina 5 y agregar la pregunta de la opción si desea calcular o no (do{}while (op == 'S' || op == 's');).







2.- Explica que sucede con el ciclo while al colocarle un valor positivo, negativo y cero y porque razón se obtiene ese resultado.

R= por los resultados binarios 0 es positivo y cualquier otro numero negativo

3.- Que sucede con el ejercicio que ejecuta el break, si se omite y que sucede si se coloca?

R=causa la salida inmediata de dicha estructura ejecución programa continua con el primer enunciado después de la estructura

4.- Que sucede con el ejercicio que ejecuta continue, si se omite y que sucede si se coloca?

R=salta los enunciados restantes del cuerpo de la estructura; la ejecución continua con la siguiente iteración del ciclo

**Conclusiones.**

El uso de operadores ternarios es mas eficiente ya que nos ayuda a tener un código mas corto; El uso de las estructuras ya conocidos de repetición en c a mi parecer es fundamental, prácticamente con ellas puedes encontrar solución a cualquier problema.

**Bibliografías.**

* <http://profesores.fi-b.unam.mx/cintia/c_basico.pdf>
* <https://www.ibm.com/developerworks/ssa/linux/library/l-lpic1-v3-103-8/index.html>
* https://www.google.com.mx/search?q=sintaxis+while&source=lnms&tbm=isch&sa=X&ved=0ahUKEwjx0IzFj8jaAhVp9IMKHejmB7AQ\_AUICigB&biw=1366&bih=623#imgrc=Kimns5HuReFYlM: