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

Laboratorios de computación

salas A y B

|  |  |
| --- | --- |
| Profesor: | García Morales Karina |
| Asignatura: | Fundamentos de Programación |
| Grupo: | 1121 |
| No de Práctica(s): | 9 |
| Integrante(s): | Tinoco Ramos Isaac Ricardo |
|  |  |
|  |  |
|  |  |
| No. de Equipo de cómputo empleado: |  |
| Semestre: | Primero |
| Fecha de entrega: | 23/10/2018 |
| Observaciones: |  |
|  |  |

Calificación\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |
| --- | --- | --- |
| escudofi_color_m2008_jpg | **Carátula para entrega de prácticas** | |
| Facultad de Ingeniería | | Laboratorios de docencia |

**Objetivo:**

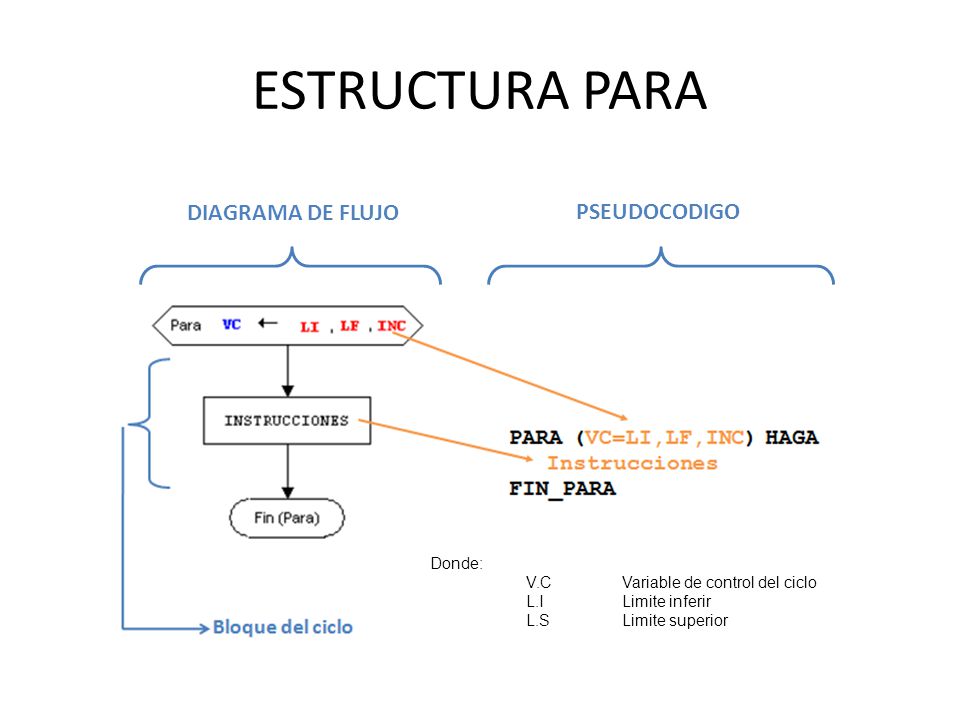
Elaborar programas en C para la resolución de problemas básicos que incluyan las estructuras de repetición y la directiva define.

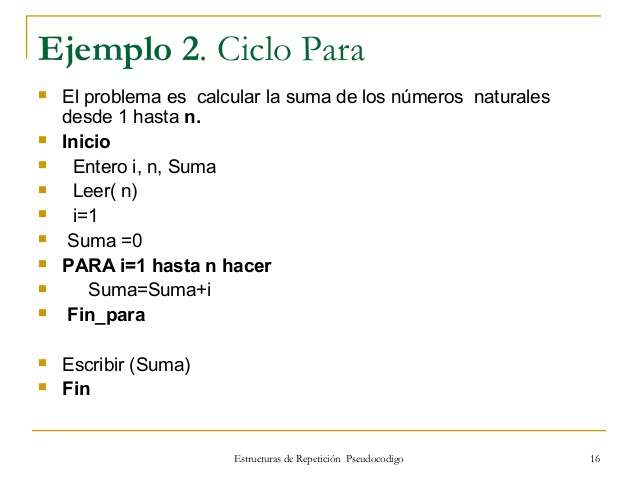
**Introducción:**

La estructura de repetición o bucle hace posible la ejecución repetida de una o más instrucciones, nos permiten ejecutar varias veces una misma línea de  
código, describen procesos que se repiten varias veces en la solución del problema. El conjunto de acciones que se repiten conforman el cuerpo del bucle y cada ejecución del cuerpo del bucle se denomina iteración

**Ciclo For:**

La declaración “for” se usa para repetir un bloque de sentencias encerradas entre llaves, un número determinado de veces. Cada vez que se ejecutan las instrucciones del bucle se vuelve a testear la condición. La declaración “for” tiene tres partes separadas por un (**;**). La inicialización de la variable local se produce una sola vez y la condición se testea cada vez que se termina la ejecución de las instrucciones dentro del bucle. Si la condición sigue cumpliéndose, las instrucciones del bucle se vuelven a ejecutar. Cuando la condición no se cumple, el bucle termina.



Cualquiera de los tres elementos de cabecera puede omitirse, aunque el punto y coma es obligatorio. También las declaraciones de inicialización, condición y expresión puede ser cualquier estamento válido en lenguaje C sin relación con las variables declaradas.

**La sintaxis es la siguiente:**

[view source](http://codigoprogramacion.com/cursos/tutoriales-c/ciclo-for-en-c-y-ejemplos.html#viewSource)

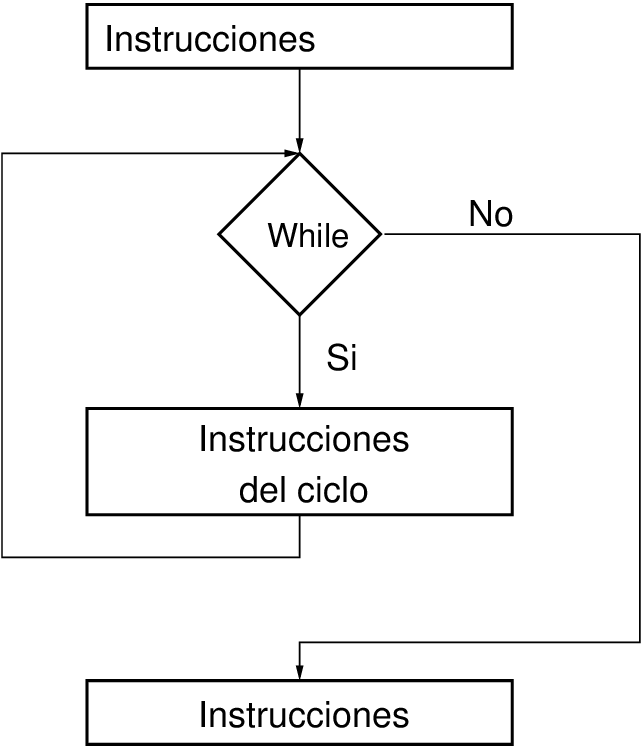
|  |  |  |
| --- | --- | --- |
| 1 | for(inicio;mientras;incremento) | |
| 2 | { |

|  |  |  |
| --- | --- | --- |
| 3 | //CODIGO A EJECUTAR | |
| 4 | } |

**Ciclo While:**

Un bucle del tipo while es un bucle de ejecución continua mientras se cumpla la expresión colocada entre paréntesis en la cabecera del bucle. La variable de prueba tendrá que cambiar para salir del bucle. La situación podrá cambiar a expensas de una expresión dentro el código del bucle o también por el cambio de un valor en una entrada de un sensor.

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.



**La sintaxis es la siguiente:**

while (expresión\_lógica) {

// Bloque de código a repetir

// mientras que la expresión

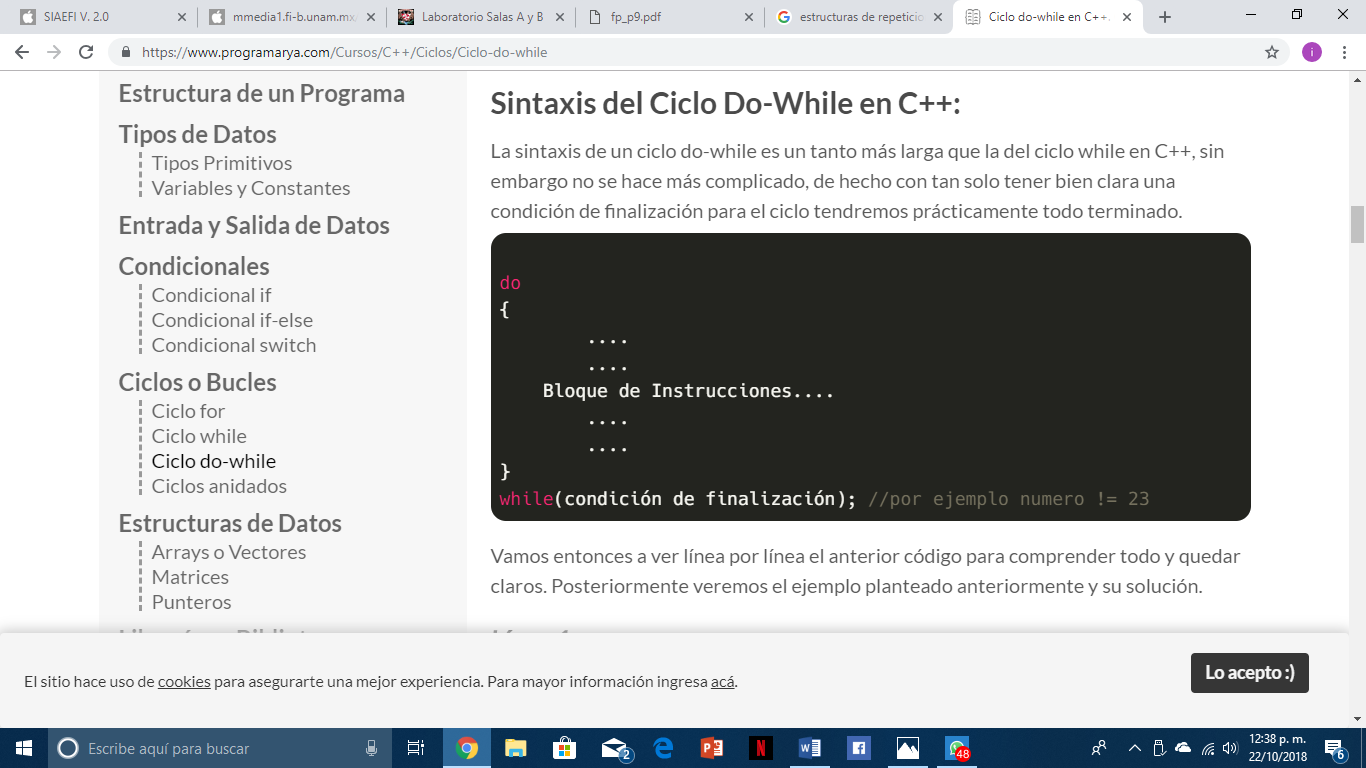
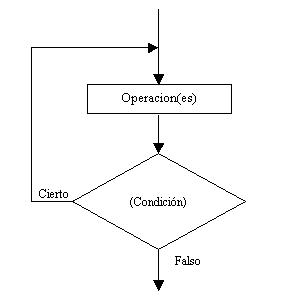
// lógica sea verdadera.

}

**Ciclo Do- While:**

Los ciclos do-while son una estructura de control cíclica, los cuales nos permiten ejecutar una o varias líneas de código de forma repetitiva sin necesidad de tener un valor inicial e incluso a veces sin siquiera conocer cuándo se va a dar el valor final, hasta aquí son similares a los ciclos while, sin embargo el ciclo do-while nos permite añadir cierta ventaja adicional y esta consiste que nos da la posibilidad de ejecutar primero el bloque de instrucciones antes de evaluar la condición necesaria, de este modo los ciclos do-while, son más efectivos para algunas situaciones específicas.

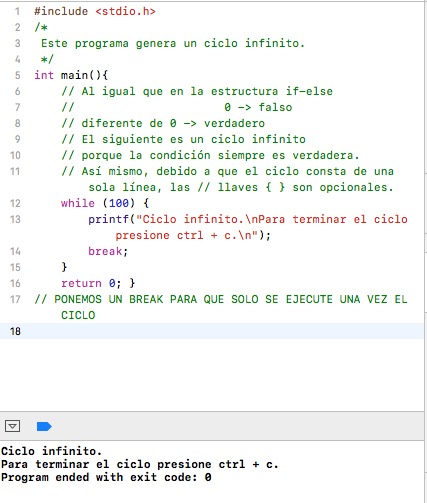
En resumen, un ciclo do-while, es una estructura de control cíclica que permite ejecutar de manera repetitiva un bloque de instrucciones sin evaluar de forma inmediata una condición específica, sino evaluándola justo después de ejecutar por primera vez el bloque de instrucciones



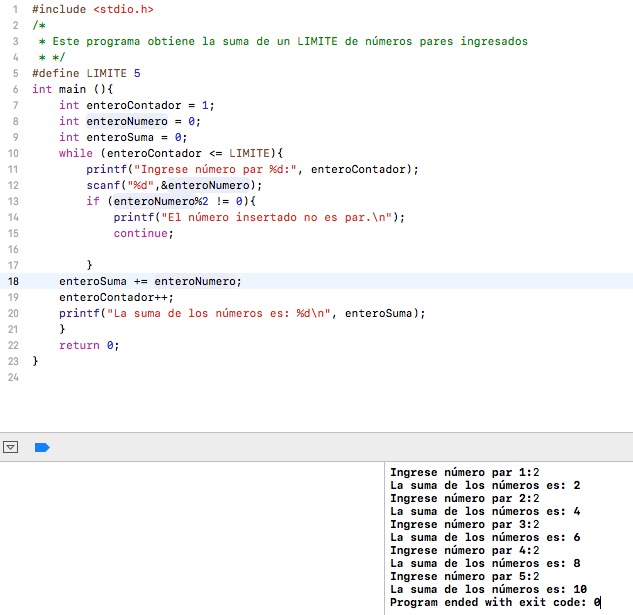
**break:**se usa en las instrucciones do, for, while para salir del bucle de una forma diferente a la indicada en el bucle, algunas veces es conveniente tener la posibilidad de abandonar un ciclo. La proposición break proporciona una salida anticipada dentro de una estructura de repetición, tal como lo hace en un switch. Un break provoca que el ciclo que lo encierra termine inmediatamente.

**Desarrollo:**

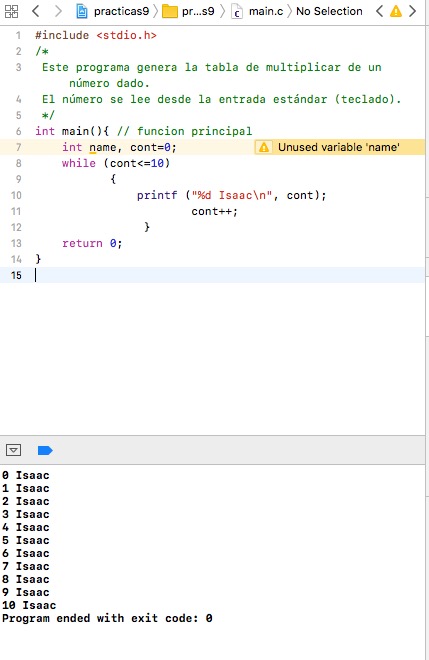
**Código (estructura while)**



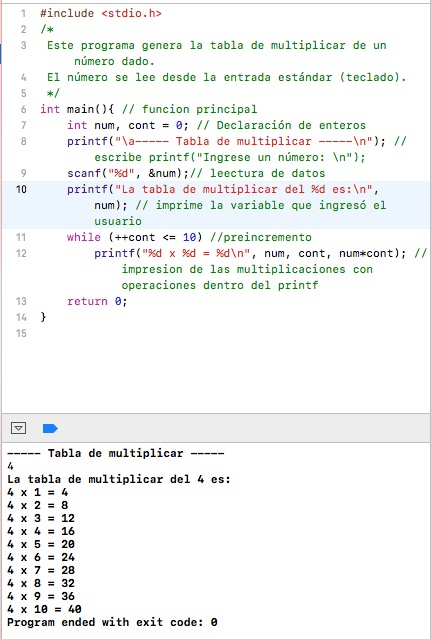
En este programa podemos ver que el ciclo se vuelve infinito gracias a que la condicion siempre va a ser verdadera.



En este programa podemos observar que lo único que hace es hacer la suma del numero ingresado “par” gracias a la estructura while que tiene como límite 5.

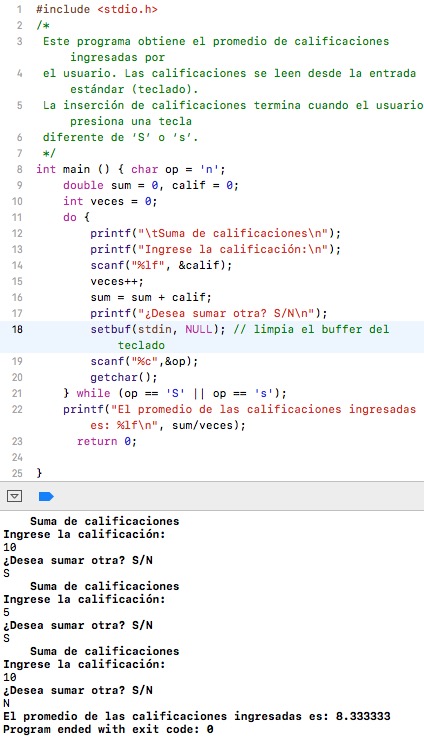


En este programa imprimimos nuestro nombre una cierta cantidad de veces utilizando un ciclo while y con un cantador con incremento.

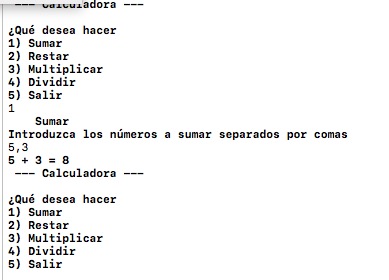
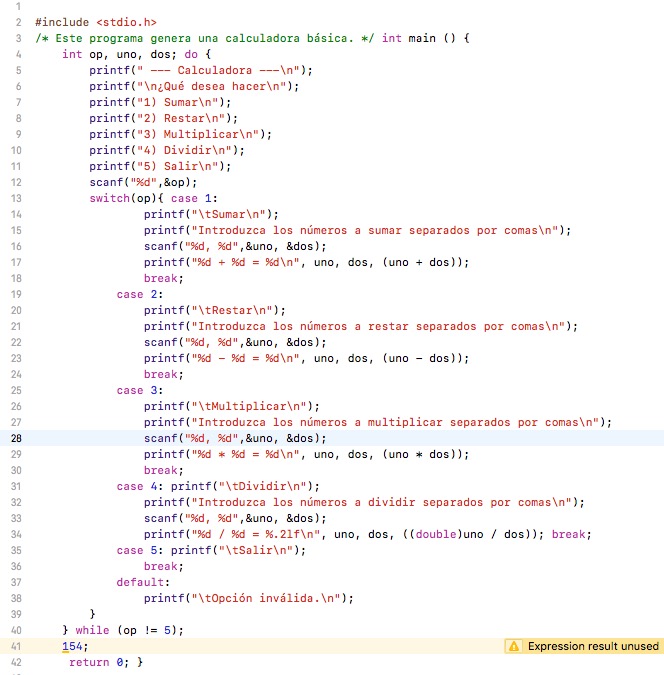


En este programa nos va a generar la tabla de cualquier numero ingresado haciendo uso del mismo ciclo while y con ayuda de un contador y en el mismo printf hacer la operación.

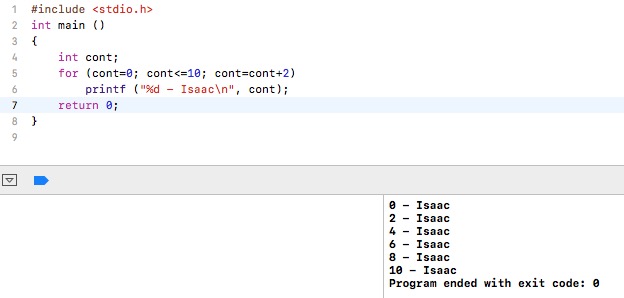
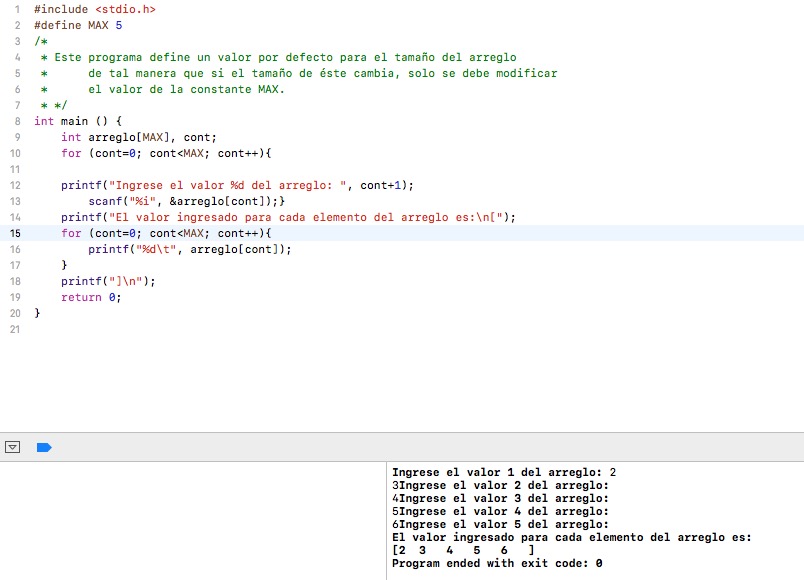
**Código (estructura do-while)**

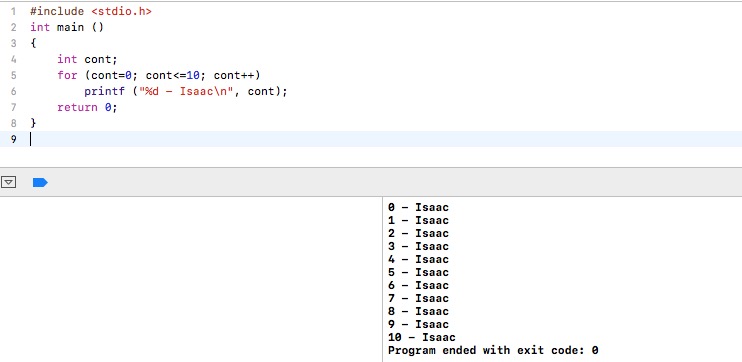


Este programa nos permite obtener el promedio de infinitas calificaiones ingresadas por el usuario gracias a una estructura do-while, pues mientras el usuario vaya ingresando valores, este los va a ir almacenando para al final sumarlos y dividirlos y asi mostrar el promedio.



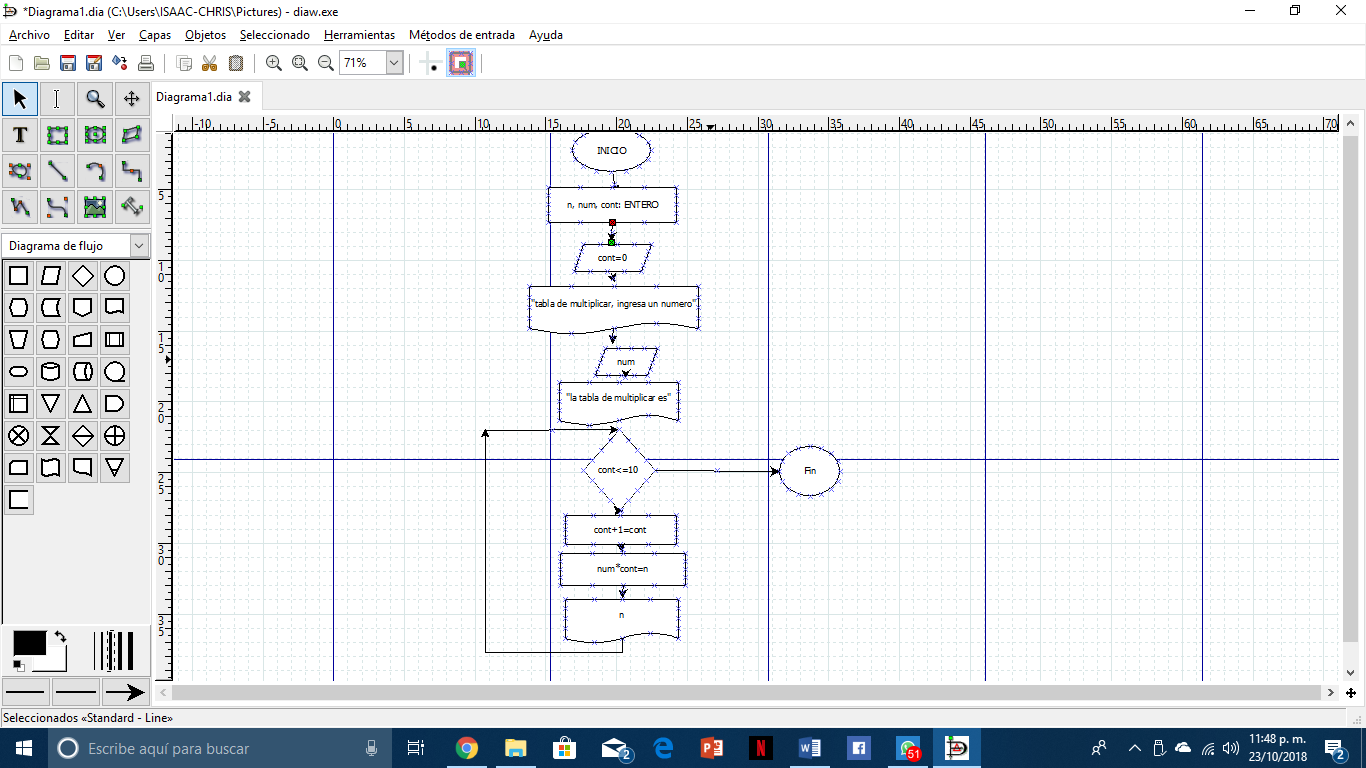
En este programa nos muestra un menú de operaciones aritméticas haciendo que se repita cuantas veces sea necesario, gracias al do-while, que internamente contiene un switch en donde hace las operaciones.

Haciendo uso del for y con un contador de incremento dos, nos mostraría solo los números pares.

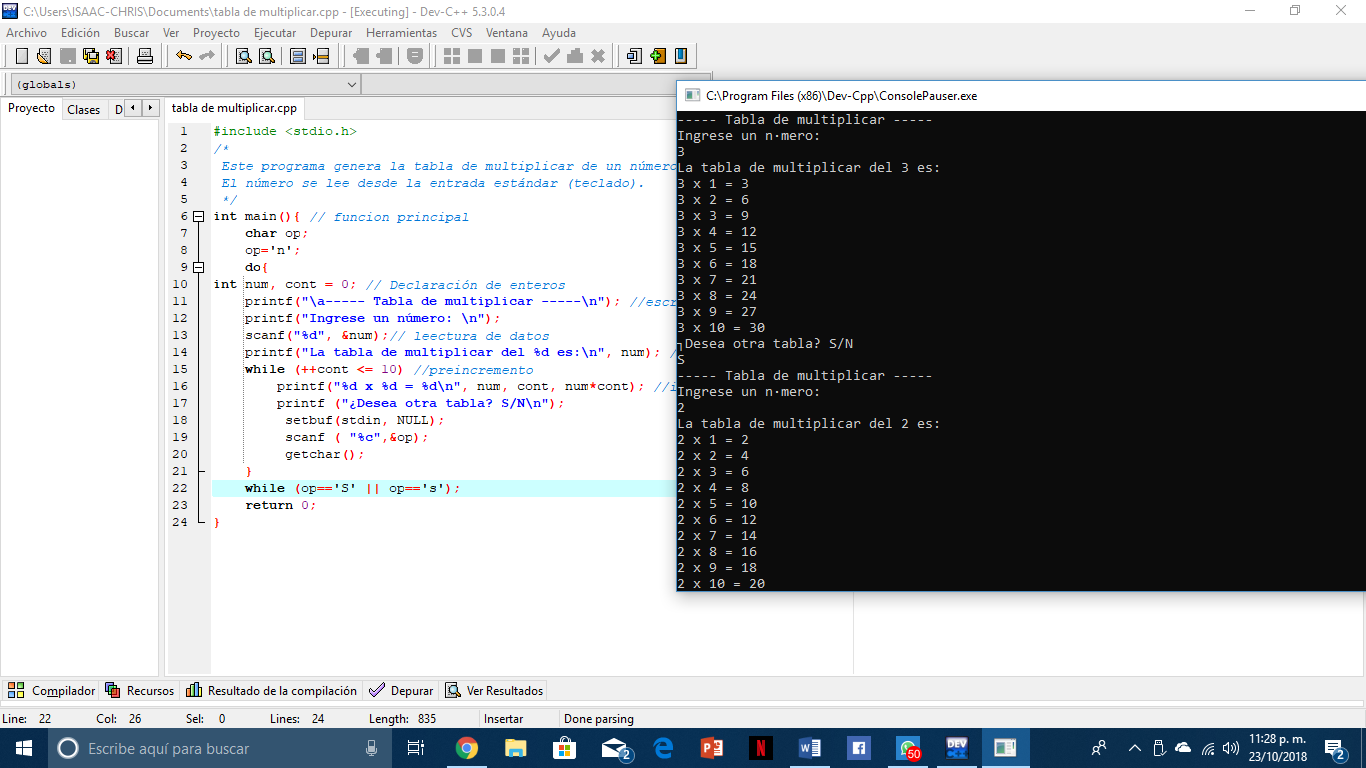


Ejercicios propuestos

1.- Dibujar el diagrama de flujo del programa para obtener la tabla de multiplicar con ciclo While.



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



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

Cuando colocamos un valor positivo o negativo en un ciclo while se ejecuta infinitas veces, debido a que en código binario los cero representan módulos apagados por lo tanto siempre van a mostrar el resultado falso.

4.- ¿Qué sucede con el ejercicio que ejecuta el break, si se omite y que sucede si se coloca?

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.

5.- ¿Qué sucede con el ejercicio que ejecuta continúe, si se omite y que sucede si se coloca?

Hace que inicie la siguiente iteración del ciclo y en caso de omitir no se itera el siguiente.

Conclusión:

Las estructuras de repetición hacen posible la ejecución repetida de una o más instrucciones, nos permiten ejecutar varias veces una misma línea de  
código, describen procesos que se repiten varias veces en la solución del problema. El conjunto de acciones que se repiten conforman el cuerpo del bucle y cada ejecución del cuerpo del bucle se denomina iteración