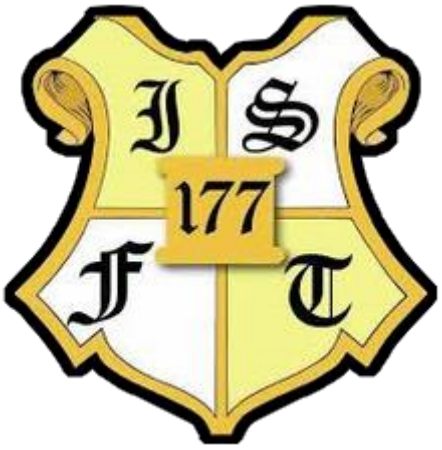
INSTITUTO SUPERIOR DE FORMACIÓN TÉCNICA N° 177



Algoritmos y Estructuras de Datos 1

Estructura de Repetición while

Prof. Lic. Walter Carnero
profewaltercarnero@gmail.com

Año 2023

Código de apunte: AyEDD05

APUNTE: AyEDD05 – ESTRUCTURA DE REPETICIÓN WHILE

LA ESTRUCTURA DE REPETICIÓN WHILE

Cuando diseñamos un algoritmo o escribimos un programa puede darse el caso en que necesitemos repetir una secuencia de código una determinada cantidad de veces o mientras una condición sea verdadera, por ejemplo se ingresan artículos mientras no se pulse la tecla escape [ESC], o se ingresan números hasta que se ingrese el número 0 (cero).

En todos los lenguajes de programación existen estructuras que permiten realizar lo antes dicho, a estas estructuras se las denomina de repetición o repetitivas, también se las suele llamar bucles (por la acción que permiten realizar).

La estructura de repetición ***while*** será la que analizaremos en el presente apunte. El diagrama de flujo que representa la estructura es el siguiente:

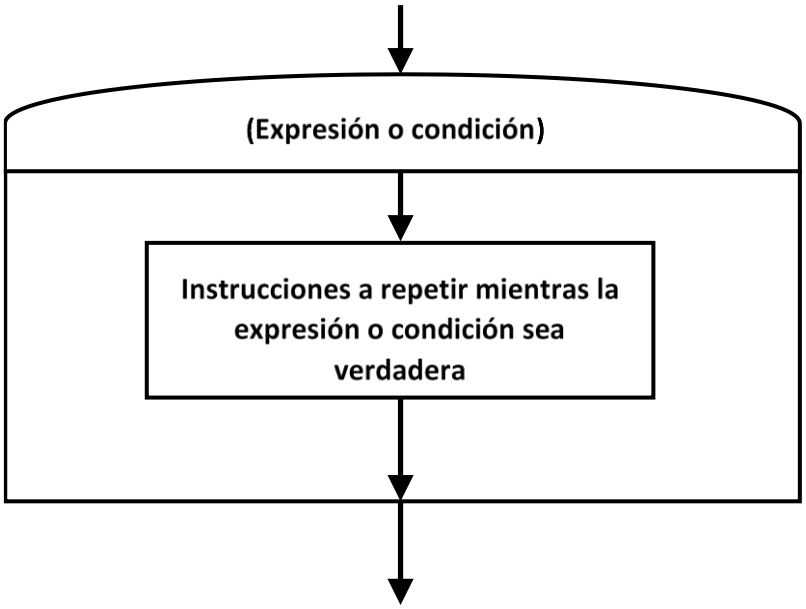


Figura 1

La estructura ***while*** permite repetir las instrucciones mientras la expresión o condición sea verdadera, observar que como la condición se evalúa al principio, si la misma es falsa de entrada, las instrucciones no se ejecutarán nunca. La sintaxis para la estructura de repetición ***while*** es la siguiente:

```
while (expresión o condición)
  {
     Instrucciones a repetir mientras la condición sea verdadera;
  }
```

El siguiente algoritmo permite mostrar por pantalla los números desde el 1 hasta el 30 inclusive.

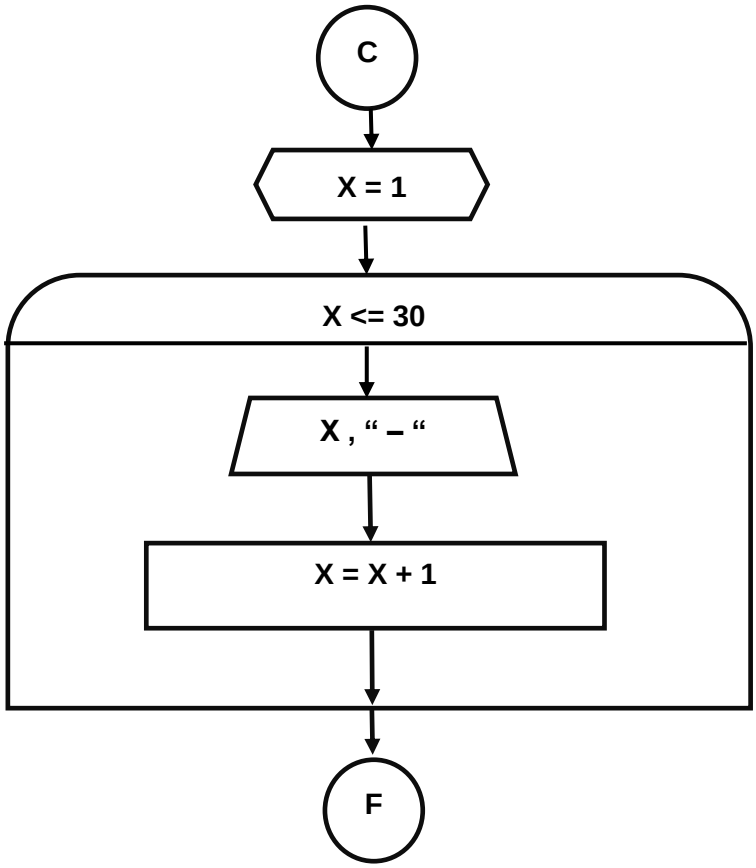


Figura 2

La salida que se observa por pantalla luego de ejecutar el algoritmo de la *figura 2*

1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-

Como se puede visualizar la estructura de repetición se ejecuta mientras la variable **X** es menor o igual a 30. A esta variable se le denomina también variable de control o variable centinela.

Sobre los diagramas de Jackson

Otra posibilidad es analizar la estructura de repetición en base a los diagramas de Jackson. Para el ejercicio propuesto con anterioridad, el resultado obtenido sería el siguiente:

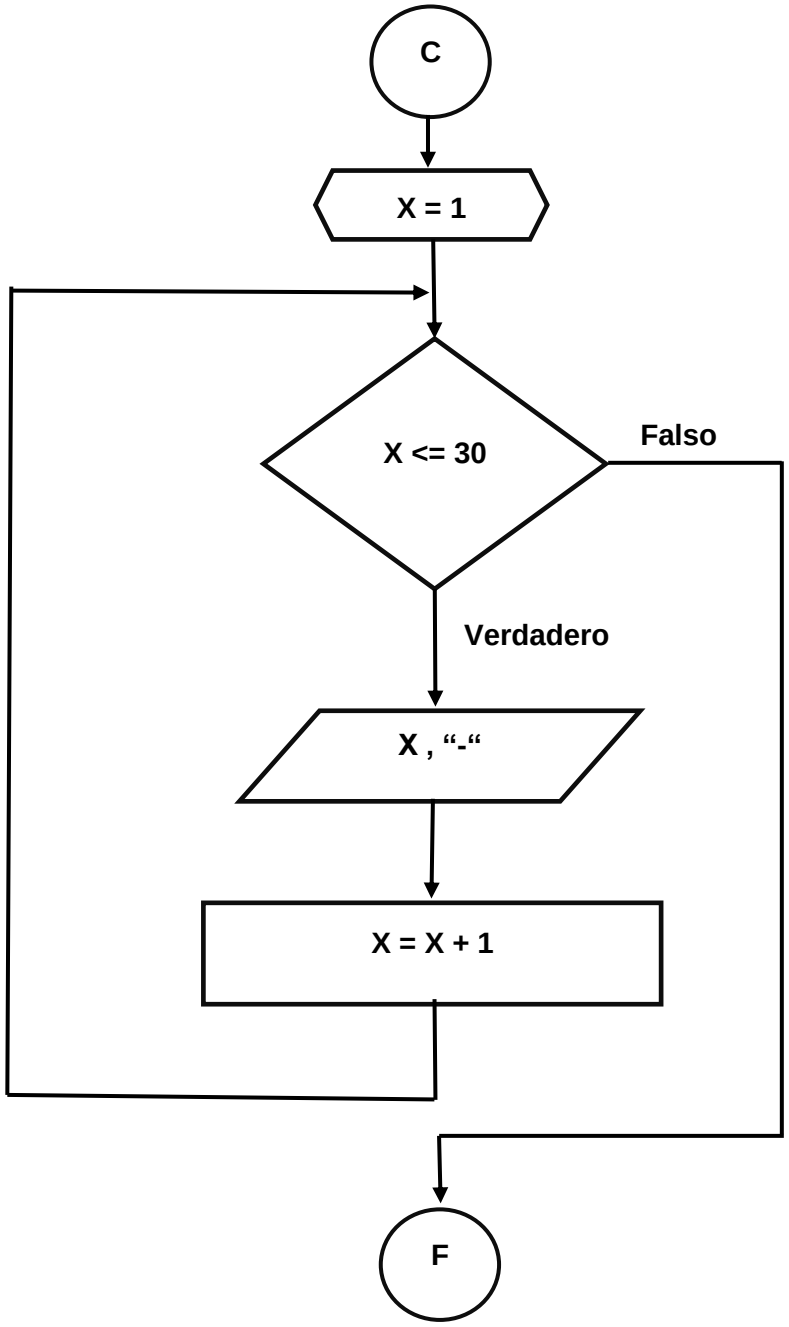


Figura 3

Sobre el código

Se detalla a continuación la codificación aplicando la sintaxis del lenguaje C# para el ejercicio propuesto, el programa que presenta la solución es el siguiente:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
namespace ConsoleApplication3
{
    class Program
    {
        static void Main(string[] args)
        {
            int x = 1;
            while(x <= 30)
            {
                Console.Write(x + "-");
                x = x + 1;
            }
            Console.ReadKey();
        }
    }
}
```

Ejemplo de aplicación

A continuación se desarrolla un algoritmo que permite mostrar por pantalla los números impares en forma regresiva desde el número 23 al número 1. El diagrama de flujo que resuelve la consigna se observa en la *figura 4.*

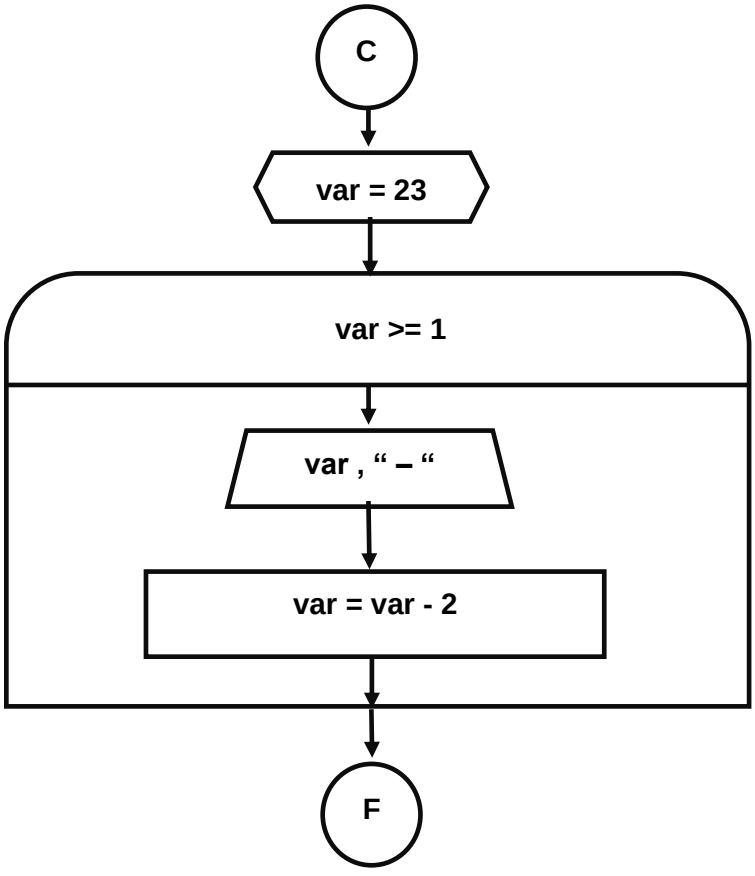


Figura 4

La salida por pantalla sería la siguiente:

23-21-19-17-15-13-11-9-7-5-3-1-

Si analizamos el funcionamiento del algoritmo de la *figura 4* observamos que la variable ***var*** se inicializa en 23 y la estructura de repetición se ejecuta mientras ***var*** sea mayor o igual a 1, siendo el decremento de la variable de control (la misma ***var***) de menos 2, es decir, primero vale 23 luego 21, 19… y así hasta 1.

El código fuente para la solución sería:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApplication3
{
    class Program
    {
        static void Main(string[] args)
        {
            int var = 23;//Se declara var como entero
            while(var >= 1) //Se ejecuta mietras var sea >= a 1
            {
                Console.Write(var + "-");
                var = var - 2; //Decremento de var
            }
            Console.ReadKey(); //Se espera la pulsación de una tecla
        }
    }
}
```

Observar que todo el texto que se encuentra luego de los símbolos **“//”** son comentarios, dichos comentarios no forman parte del programa (el compilador no los ve), sólo son de utilidad para el programador. Una buena práctica de programación es utilizar siempre comentarios, es importante que los programas que desarrollemos se encuentren bien documentados, esto nos permitirá que a futuro las correcciones o mejoras que realicemos en nuestros códigos sean más fáciles de implementar.