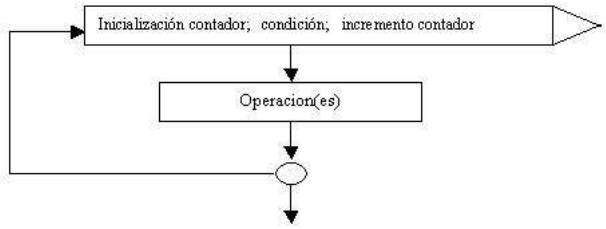
10 - Estructura repetitiva for

Cualquier problema que requiera una estructura repetitiva se puede resolver empleando la estructura while. Pero hay otra estructura repetitiva cuyo planteo es más sencillo en ciertas situaciones.

En general, la estructura for se usa en aquellas situaciones en las cuales CONOCEMOS la cantidad de veces que queremos que se ejecute el bloque de instrucciones. Ejemplo: cargar 10 números, ingresar 5 notas de alumnos, etc. Conocemos de antemano la cantidad de veces que queremos que el bloque se repita. Veremos, sin embargo, que en el lenguaje C# la estructura for puede usarse en cualquier situación repetitiva, porque en última instancia no es otra cosa que una estructura while generalizada.

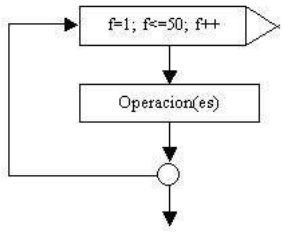
Representación gráfica:



En su forma más típica y básica, esta estructura requiere una variable entera que cumple la función de un CONTADOR de vueltas. En la sección indicada como "inicialización contador", se suele colocar el nombre de la variable que hará de contador, asignándole a dicha variable un valor inicial. En la sección de "condición" se coloca la condición que deberá ser verdadera para que el ciclo continúe (en caso de un falso, el ciclo se detendrá). Y finalmente, en la sección de "incremento contador" se coloca una instrucción que permite modificar el valor de la variable que hace de contador (para permitir que alguna vez la condición sea falsa)

Cuando el ciclo comienza, antes de dar la primera vuelta, la variable del for toma el valor indicado en la sección de de "inicialización contador". Inmediatamente se verifica, en forma automática, si la condición es verdadera. En caso de serlo se ejecuta el bloque de operaciones del ciclo, y al finalizar el mismo se ejecuta la instrucción que se haya colocado en la tercer sección.
Seguidamente, se vuelve a controlar el valor de la condición, y así prosigue hasta que dicha condición entregue un falso.

Si conocemos la cantidad de veces que se repite el bloque es muy sencillo emplear un for, por ejemplo si queremo que se repita 50 veces el bloque de instrucciones puede hacerse así:



La variable del for puede tener cualquier nombre. En este ejemplo se la ha definido con el nombre f.
Analicemos el ejemplo:

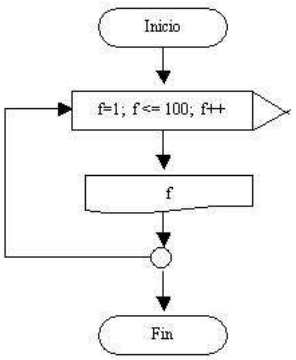
```
- La variable f  toma inicialmente el valor 1.
- Se controla automáticamente el valor de la condición: como f vale 1 y esto es menor
que 50, la condición da verdadero.
- Como la condición fue verdadera, se ejecutan la/s operación/es.
- Al finalizar de ejecutarlas, se retorna a la instrucción f++, por lo que la
variable f se incrementa en uno.
- Se vuelve a controlar (automáticamente) si f es menor o igual a 50.
Como ahora su valor es 2, se ejecuta nuevamente el bloque de instrucciones e
incrementa nuevamente la variable del for al terminar el mismo.
- El proceso se repetirá hasta que la variable f sea incrementada al valor 51.
En este momento la condición será falsa, y el ciclo se detendrá.
```

La variable f PUEDE ser modificada dentro del bloque de operaciones del for, aunque esto podría causar problemas de lógica si el programador es inexperto.
La variable f puede ser inicializada en cualquier valor y finalizar en cualquier valor. Además, no es obligatorio que la instrucción de modificación sea un incremento del tipo contador (f++).
Cualquier instrucción que modifique el valor de la variable es válida. Si por ejemplo se escribe f=f+2 en lugar de f++, el valor de f será incrementado de a 2 en cada vuelta, y no de a 1. En este caso, esto significará que el ciclo no efectuará las 50 vueltas sino sólo 25.

Problema 1:

Realizar un programa que imprima en pantalla los números del 1 al 100.

Diagrama de flujo:



Podemos observar y comparar con el problema realizado con el while. Con la estructura while el CONTADOR x sirve para contar las vueltas. Con el for el CONTADOR f cumple dicha función.
Inicialmente f vale 1 y como no es superior a 100 se ejecuta el bloque, imprimimos el contenido de f, al finalizar el bloque repetitivo se incrementa la variable f en 1, como 2 no es superior a 100 se repite el bloque de instrucciones.
Cuando la variable del for llega a 101 sale de la estructura repetitiva y continúa la ejecución del algoritmo que se indica después del círculo.
La variable f (o como sea que se decida llamarla) debe estar definida como una variable más.

Programa:

Ver video

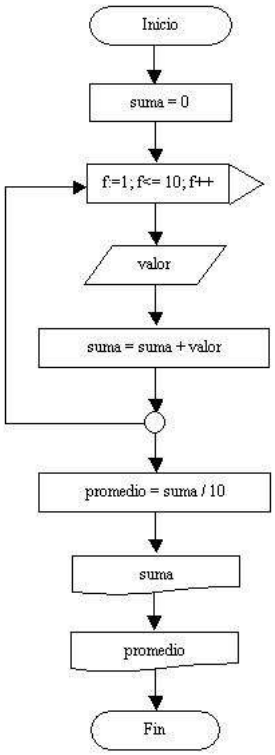
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace EstructuraRepetitivaFor1
{
    class Program
    {
        static void Main(string[] args)
        {
            int f;
            for(f=1;f<=100;f++)
            {
                Console.Write(f);
                Console.Write("-");
            }
            Console.ReadKey();
        }
    }
}
```

Problema 2:

: Desarrollar un programa que permita la carga de 10 valores por teclado y nos muestre posteriormente la suma de los valores ingresados y su promedio. Este problema ya lo desarrollamos, lo resolveremos empleando la estructura for.

Diagrama de flujo:



En este caso, a la variable del for (f) sólo se la requiere para que se repita el bloque de instrucciones 10 veces.

Programa:

Ver video

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

namespace EstructuraRepetitivaFor2
{
    class Program
    {
        static void Main(string[] args)
        {
            int suma,f,valor,promedio;
            string linea;
            suma=0;
            for(f=1;f<=10;f++)
            {
                Console.Write("Ingrese valor:");
                linea=Console.ReadLine();
                valor=int.Parse(linea);
                suma=suma+valor;
            }
            Console.Write("La suma es:");
```

```
            Console.WriteLine(suma);
            promedio=suma/10;
            Console.Write("El promedio es:");
            Console.Write(promedio);
            Console.ReadKey();
        }
    }
}
```

El problema requiere que se carguen 10 valores y se sumen los mismos.

Tener en cuenta encerrar entre llaves bloque de instrucciones a repetir dentro del for.

El promedio se calcula fuera del for luego de haber cargado los 10 valores.

Problema 3:

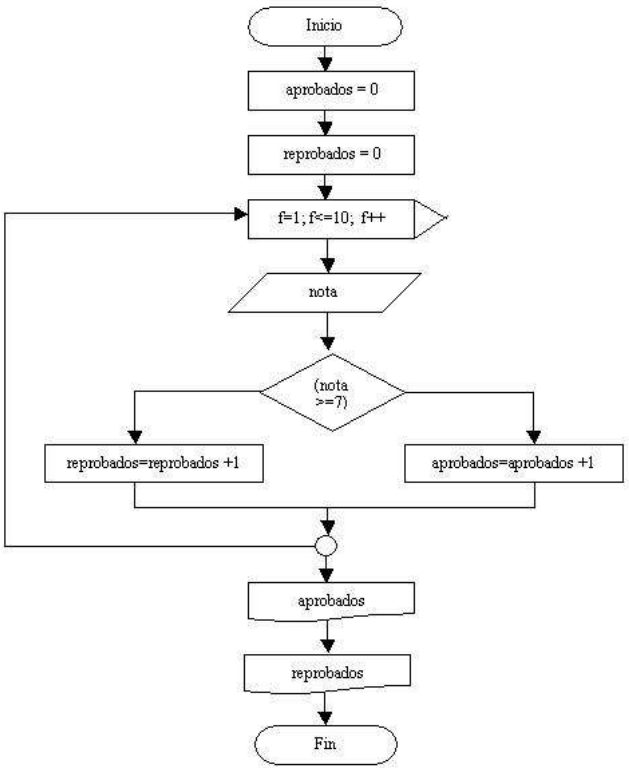
Escribir un programa que lea 10 notas de alumnos y nos informe cuántos tienen notas mayores o iguales a 7 y cuántos menores.

Para resolver este problema se requieren tres contadores:

```
aprobados (Cuenta la cantidad de alumnos aprobados)
reprobados (Cuenta la cantidad de reprobados)
f (es el contador del for)
```

Dentro de la estructura repetitiva debemos hacer la carga de la variable nota y verificar con una estructura condicional si el contenido de la variable nota es mayor o igual a 7 para incrementar el contador aprobados, en caso de que la condición retorne falso debemos incrementar la variable reprobados.

Diagrama de flujo:



Los contadores aprobados y reprobados deben imprimirse FUERA de la estructura repetitiva.

Es fundamental inicializar los contadores aprobados y reprobados en cero antes de entrar a la estructura for.

Importante: Un error común es inicializar los contadores dentro de la estructura repetitiva. En caso de hacer esto los contadores se fijan en cero en cada ciclo del for, por lo que al finalizar el for como máximo el contador puede tener el valor 1.

Programa:

Ver video

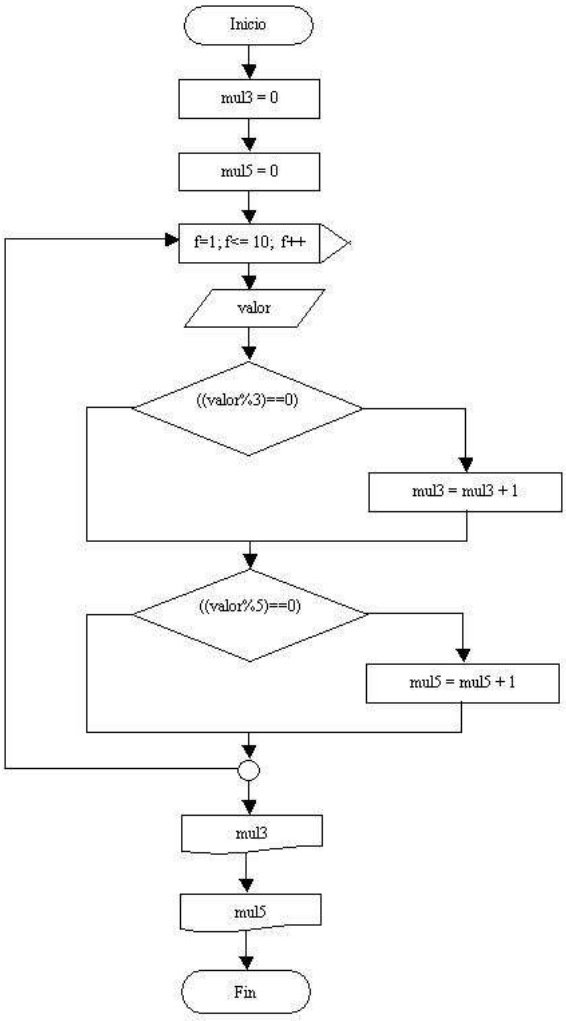
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace EstructuraRepetitivaFor3
{
    class Program
    {
        static void Main(string[] args)
        {
            int aprobados,reprobados,f,nota;
            string linea;
            aprobados=0;
            reprobados=0;
            for(f=1;f<=10;f++)
            {
                Console.Write("Ingrese la nota:");
                linea = Console.ReadLine();
                nota=int.Parse(linea);
                if (nota>=7)
                {
                    aprobados=aprobados+1;
                }
                else
                {
                    reprobados=reprobados+1;
                }
            }
            Console.Write("Cantidad de aprobados:");
            Console.WriteLine(aprobados);
            Console.Write("Cantidad de reprobados:");
            Console.Write(reprobados);
            Console.ReadKey();
        }
    }
}
```

Problema 4:

Escribir un programa que lea 10 números enteros y luego muestre cuántos valores ingresados fueron múltiplos de 3 y cuántos de 5. Debemos tener en cuenta que hay números que son múltiplos de 3 y de 5 a la vez.

Diagrama de flujo:



Tengamos en cuenta que el operador matemático % retorna el resto de dividir un valor por otro, en este caso: valor%3 retorna el resto de dividir el valor que ingresamos por teclado, por tres.
Veamos: si ingresamos 6 el resto de dividirlo por 3 es 0, si ingresamos 12 el resto de dividirlo por 3 es 0. Generalizando: cuando el resto de dividir por 3 al valor que ingresamos por teclado es cero, se trata de un múltiplo de dicho valor.

Ahora bien ¿por qué no hemos dispuesto una estructura if anidada? Porque hay valores que son múltiplos de 3 y de 5 a la vez. Por lo tanto con if anidados no podríamos analizar los dos casos.
Es importante darse cuenta cuando conviene emplear if anidados y cuando no debe emplearse.

Programa:

Ver video

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

namespace EstructuraRepetitivaFor4
{
    class Program
    {
        static void Main(string[] args)
        {
            int mul3,mul5,valor,f;
            string linea;
            mul3=0;
```

```
            mul5=0;
            for(f=1;f<=10;f++)
            {
                Console.Write("Ingrese un valor:");
                linea = Console.ReadLine();
                valor=int.Parse(linea);
                if (valor%3==0)
                {
                    mul3=mul3+1;
                }
                if (valor%5==0)
                {
                    mul5=mul5+1;
                }
            }
            Console.Write("Cantidad de valores ingresados múltiplos de 3:");
            Console.WriteLine(mul3);
            Console.Write("Cantidad de valores ingresados múltiplos de 5:");
            Console.Write(mul5);
            Console.ReadKey();
        }
    }
}
```

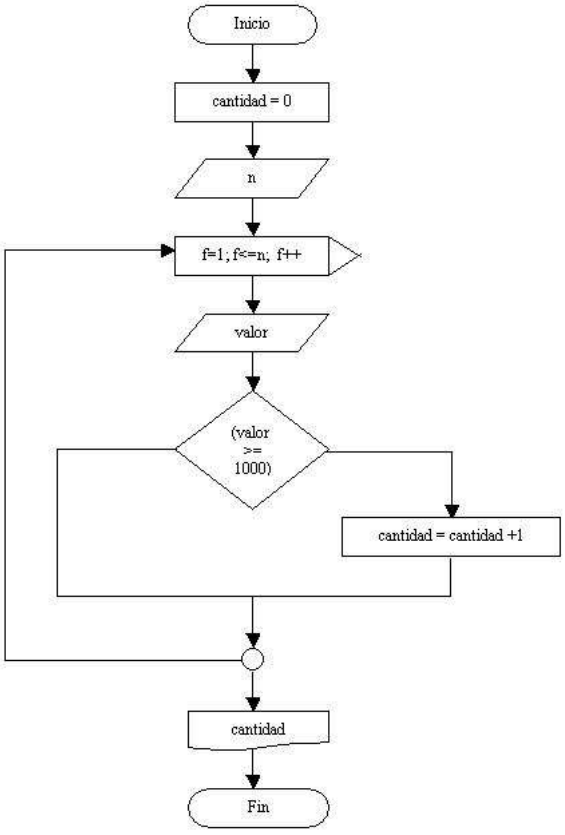
Problema 5:

Escribir un programa que lea n números enteros y calcule la cantidad de valores mayores o iguales a 1000.

Este tipo de problemas también se puede resolver empleando la estructura repetitiva for. Lo primero que se hace es cargar una variable que indique la cantidad de valores a ingresar. Dicha variable se carga antes de entrar a la estructura repetitiva for.

La estructura for permite que el valor inicial o final dependa de una variable cargada previamente por teclado.

Diagrama de flujo:



Tenemos un contador llamado cantidad y f que es el contador del for.
La variable entera n se carga previo al inicio del for, por lo que podemos fijar el valor final del for con la variable n.
Por ejemplo si el operador carga 5 en n la estructura repetitiva for se ejecutará 5 veces.
La variable valor se ingresa dentro de la estructura repetitiva, y se verifica si el valor de la misma es mayor o igual a 1000, en dicho caso se incrementa en uno el contador cantidad.
Fuera de la estructura repetitiva imprimimos el contador cantidad que tiene almacenado la cantidad de valores ingresados mayores o iguales a 1000.

Programa:

Ver video

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

namespace EstructuraRepetitivaFor5
{
    class Program
    {
        static void Main(string[] args)
        {
            int cantidad,n,f,valor;
            string linea;
            cantidad=0;
            Console.Write("Cuantos valores ingresará:");
            linea = Console.ReadLine();
            n=int.Parse(linea);
            for(f=1;f<=n;f++)
            {
                Console.Write("Ingrese el valor:");
                linea = Console.ReadLine();
                valor = int.Parse(linea);
                if (valor>=1000)
                {
                    cantidad=cantidad+1;
                }
            }
            Console.Write("La cantidad de valores ingresados mayores o iguales a 1000 sc
            Console.Write(cantidad);
            Console.ReadKey();
        }
    }
}
```

Problemas propuestos

Ha llegado nuevamente la parte fundamental, que es el momento donde uno desarrolla individualmente un algoritmo para la resolución de un problema.

1. Confeccionar un programa que lea n pares de datos, cada par de datos corresponde a la medida de la base y la altura de un triángulo. El programa deberá informar:
   a) De cada triángulo la medida de su base, su altura y su superficie.
   b) La cantidad de triángulos cuya superficie es mayor a 12.
   Ver video

2. Desarrollar un programa que solicite la carga de 10 números e imprima la suma de los últimos 5 valores ingresados.
   Ver video

3. Desarrollar un programa que muestre la tabla de multiplicar del 5 (del 5 al 50)
   Ver video

4. Confeccionar un programa que permita ingresar un valor del 1 al 10 y nos muestre la tabla de multiplicar del mismo (los primeros 13 términos)
   Ejemplo: Si ingreso 3 deberá aparecer en pantalla los valores 3, 6, 9, hasta el 39.
   Ver video

5. Realizar un programa que lea los lados de n triángulos, e informar:
   a) De cada uno de ellos, qué tipo de triángulo es: equilátero (tres lados iguales), isósceles (dos lados iguales), o escaleno (ningún lado igual)
   b) Cantidad de triángulos de cada tipo.
   c) Tipo de triángulo que posee menor cantidad.
   Ver video

6. Escribir un programa que pida ingresar coordenadas (x,y) que representan puntos en el plano.
   Informar cuántos puntos se han ingresado en el primer, segundo, tercer y cuarto cuadrante. Al comenzar el programa se pide que se ingrese la cantidad de puntos a procesar.
   Ver video

7. Se realiza la carga de 10 valores enteros por teclado. Se desea conocer:
   a) La cantidad de valores ingresados negativos.
   b) La cantidad de valores ingresados positivos.
   c) La cantidad de múltiplos de 15.
   d) El valor acumulado de los números ingresados que son pares.
   Ver video

8. Se cuenta con la siguiente información:
   Las edades de 50 estudiantes del turno mañana.
   Las edades de 60 estudiantes del turno tarde.
   Las edades de 110 estudiantes del turno noche.
   Las edades de cada estudiante deben ingresarse por teclado.
   a) Obtener el promedio de las edades de cada turno (tres promedios)
   b) Imprimir dichos promedios (promedio de cada turno)
   c) Mostrar por pantalla un mensaje que indique cual de los tres turnos tiene un promedio de edades menor.
   Ver video

Solución

Retornar