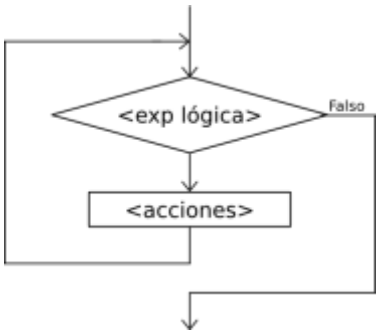
Estructuras repetitivas:

Mientras



La instrucción `Mientras` ejecuta una secuencia de instrucciones mientras una condición sea verdadera.

```
Mientras <condición> Hacer
    <instrucciones>
FinMientras
```

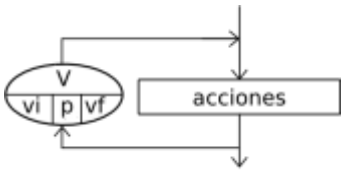
- Al ejecutarse esta instrucción, la condición es evaluada. Si la condición resulta verdadera, se ejecuta una vez la secuencia de instrucciones que forman el cuerpo del ciclo. Al finalizar la ejecución del cuerpo del ciclo se vuelve a evaluar la condición y, si es verdadera, la ejecución se repite. Estos pasos se repiten mientras la condición sea verdadera.
- Se puede dar la circunstancia que las instrucciones del bucle no se ejecuten nunca, si al evaluar por primera vez la condición resulta ser falsa.
- Si la condición siempre es verdadera, al ejecutar esta instrucción se produce un ciclo infinito. A fin de evitarlo, las instrucciones del cuerpo del ciclo deben contener alguna instrucción que modifique la o las variables involucradas en la condición, de modo que ésta sea falsificada en algún momento y así finalice la ejecución del ciclo.

Ejemplo:

Crea un programa que pida al usuario una contraseña, de forma repetitiva mientras que no introduzca "asdasd". Cuando finalmente escriba la contraseña correcta, se le dirá "Bienvenido" y terminará el programa.

```
Proceso login
        Definir secreto, clave como cadena;
        secreto <- "asdasd";
        Escribir "Dime la clave:";
        Leer clave;
        Mientras clave<>secreto Hacer
                Escribir "Clave incorrecta!!!";
                Escribir "Dime la clave:";
                Leer clave;
        FinMientras
        Escribir "Bienvenido!!!";
        Escribir "Programa terminado";
FinProceso
```

Para



La instrucción `Para` ejecuta una secuencia de instrucciones un número determinado de veces.

```
Para <variable> <- <inicial> Hasta <final> [Con Paso <paso>] Hacer
        <instrucciones>
FinPara
```

- Al ingresar al bloque, la variable `<variable>` recibe el valor `<inicial>` y se ejecuta la secuencia de instrucciones que forma el cuerpo del ciclo.
- Luego se incrementa la variable `<variable>` en `<paso>` unidades y se evalúa si el valor almacenado en `<variable>` superó al valor `<final>`.
- Si esto es falso se repite hasta que `<variable>` supere a `<final>`.
- Si se omite la cláusula `Con Paso <paso>`, la variable `<variable>` se incrementará en 1.

Ejemplo

Escribir en pantalla del 1 al 10.

```
Proceso Contar
        Definir var como Entero;
        Para var<-1 Hasta 10 Hacer
                Escribir Sin Saltar var," ";
        FinPara
FinProceso
```

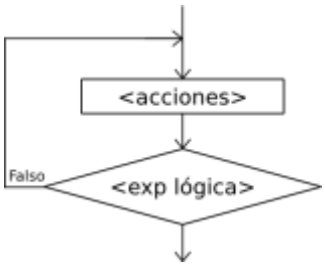
Escribir en pantalla de 10 al 1.

```
Proceso ContarDescesdente
        Definir var como Entero;
        Para var<-10 Hasta 1 Con Paso -1 Hacer
                Escribir Sin Saltar var," ";
        FinPara
FinProceso
```

Escribir los número pares desde el 2 al 10.

```
Proceso ContarPares
        Definir var como Entero;
        Para var<-2 Hasta 10 Con Paso 2 Hacer
                Escribir Sin Saltar var," ";
        FinPara
FinProceso
```

Repetir - Hasta Que



La instrucción `Repetir-Hasta Que` ejecuta una secuencia de instrucciones hasta que la condición sea verdadera.

```
Repetir
    <instrucciones>
Hasta Que <condición>
```

- Al ejecutarse esta instrucción, la secuencia de instrucciones que forma el cuerpo del ciclo se ejecuta una vez y luego se evalúa la condición. Si la condición es falsa, el cuerpo

del ciclo se ejecuta nuevamente y se vuelve a evaluar la condición. Esto se repite hasta que la condición sea verdadera.

- Note que, dado que la condición se evalúa al final, las instrucciones del cuerpo del ciclo serán ejecutadas al menos una vez.
- Además, a fin de evitar ciclos infinitos, el cuerpo del ciclo debe contener alguna instrucción que modifique la o las variables involucradas en la condición de modo que en algún momento la condición sea verdadera y se finalice la ejecución del ciclo.

Ejemplo:

Crea un programa que pida al usuario una contraseña, de forma repetitiva mientras que no introduzca "asdasd". Cuando finalmente escriba la contraseña correcta, se le dirá "Bienvenido" y terminará el programa.

```
Proceso login
        Definir secreto, clave como cadena;
        secreto <- "asdasd";
        Repetir
                Escribir "Dime la clave:";
                Leer clave;
                Si clave<>secreto Entonces
                        Escribir "Clave incorrecta!!!";
                FinSi
        Hasta Que clave=secreto
        Escribir "Bienvenido!!!";
        Escribir "Programa terminado";
FinProceso
```

Uso específico de variables: contadores, acumuladores e indicadores

Contadores

Un contador es una variable entera que la utilizamos para contar cuando ocurre un suceso. Un contador:

- Se **inicializa** a un valor inicial.
- `cont <- 0;`
- Se **incrementa**, cuando ocurre el suceso que estamos contado se le suma 1.
- `cont <- cont + 1;`

Ejemplo

Introducir 5 número y contar los números pares.

```
Proceso ContarPares
        Definir var,cont,num como Entero;
        cont<-0;
        Para var<-1 Hasta 5 Hacer
                Escribir Sin Saltar "Dime un número:";
                Leer num;
                Si num % 2 = 0 Entonces
                        cont<-cont+1;
                FinSi
        FinPara
        Escribir "Has introducido ",cont," números pares.";
FinProceso
```

Acumuladores

Un acumulador es una variable numérica que permite ir acumulando operaciones. Me permite ir haciendo operaciones parciales. Un acumulador:

- Se **inicializa** a un valor inicial según la operación que se va a acumular: a 0 si es una suma o a 1 si es un producto.
- Se **acumula** un valor intermedio.
- `acum <- acum + num;`

Ejemplo

Introducir 5 número y sumar los números pares.

```
Proceso SumarPares
        Definir var,suma,num como Entero;
        suma<-0;
        Para var<-1 Hasta 5 Hacer
                Escribir Sin Saltar "Dime un número:";
                Leer num;
                Si num % 2 = 0 Entonces
                        suma<-suma+num;
                FinSi
        FinPara
        Escribir "La suma de los números pares es ",suma;
FinProceso
```

Indicadores

Un indicador es una variable lógico, que usamos para recordar o indicar algún suceso. Un indicador:

- Se **inicializa** a un valor lógico que indica que el suceso no ha ocurrido.

  indicador <- Falso

- Cuando ocurre el suceso que queremos recordar cambiamos su valor.

  indicador <- Verdadero

Ejemplo

Introducir 5 número e indicar si se ha introducido algún número par.

```
Proceso RecordarPar
        Definir var,num como Entero;
        Definir indicador como Logico;
        indicador <- Falso;
        Para var<-1 Hasta 5 Hacer
                Escribir Sin Saltar "Dime un número:";
                Leer num;
                Si num % 2 = 0 Entonces
                        indicador <- Verdadero;
                FinSi
        FinPara
        Si indicador Entonces
                Escribir "Has introducido algún número par";
        SiNo
                Escribir "No has introducido algún número par";
        FinSi
FinProceso
```

Ejercicios estructuras repetitivas

Ejercicio 1

Crea una aplicación que pida un número y calcule su factorial (El factorial de un número es el producto de todos los enteros entre 1 y el propio número y se representa por el número seguido de un signo de exclamación. Por ejemplo 5! = 1x2x3x4x5=120),

Ejercicio 2

Crea una aplicación que permita adivinar un número. La aplicación genera un número aleatorio del 1 al 100. A continuación va pidiendo números y va respondiendo si el número a adivinar es mayor o menor que el introducido,a demás de los intentos que te quedan (tienes 10 intentos para acertarlo). El programa termina cuando se acierta el número (además te dice en cuantos intentos lo has acertado), si se llega al limite de intentos te muestra el número que había generado.

Ejercicio 3

Algoritmo que pida números hasta que se introduzca un cero. Debe imprimir la suma y la media de todos los números introducidos.

Ejercicio 4

Realizar un algoritmo que pida números (se pedirá por teclado la cantidad de números a introducir). El programa debe informar de cuantos números introducidos son mayores que 0, menores que 0 e iguales a 0.

Ejercicio 5

Algoritmo que pida caracteres e imprima 'VOCAL' si son vocales y 'NO VOCAL' en caso contrario, el programa termina cuando se introduce un espacio.

Ejercicio 6

Escribir un programa que imprima todos los números pares entre dos números que se le pidan al usuario.

Ejercicio 7

Realizar una algoritmo que muestre la tabla de multiplicar de un número introducido por teclado.

Ejercicio 8

Escribe un programa que pida el limite inferior y superior de un intervalo. Si el límite inferior es mayor que el superior lo tiene que volver a pedir. A continuación se van introduciendo números hasta que introduzcamos el 0. Cuando termine el programa dará las siguientes informaciones:

- La suma de los números que están dentro del intervalo (intervalo abierto).
- Cuantos números están fuera del intervalo.
- He informa si hemos introducido algún número igual a los límites del intervalo.

Ejercicio 9

Escribe un programa que dados dos números, uno real (base) y un entero positivo (exponente), saque por pantalla el resultado de la potencia. No se puede utilizar el operador de potencia.

Ejercicio 10

Algoritmo que muestre la tabla de multiplicar de los números 1,2,3,4 y 5.

Ejercicio 11

Escribe un programa que diga si un número introducido por teclado es o no primo. Un número primo es aquel que sólo es divisible entre él mismo y la unidad. Nota: Es suficiente probar hasta la raíz cuadrada del número para ver si es divisible por algún otro número.

Ejercicio 12

Realizar un algoritmo para determinar cuánto ahorrará una persona en un año, si al final de cada mes deposita cantidades variables de dinero; además, se quiere saber cuánto lleva ahorrado cada mes.

Ejercicio 13

Una empresa tiene el registro de las horas que trabaja diariamente un empleado durante la semana (seis días) y requiere determinar el total de éstas, así como el sueldo que recibirá por las horas trabajadas.

Ejercicio 14

Una persona se encuentra en el kilómetro 70 de una carretera, otra se encuentra en el km 150, los coches tienen sentido opuesto y tienen la misma velocidad. Realizar un programa para determinar en qué kilómetro de esa carretera se encontrarán.

Ejercicio 15

Una persona adquirió un producto para pagar en 20 meses. El primer mes pagó 10 €, el segundo 20 €, el tercero 40 € y así sucesivamente. Realizar un algoritmo para determinar cuánto debe pagar mensualmente y el total de lo que pagó después de los 20 meses.

Ejercicio 16

Una empresa les paga a sus empleados con base en las horas trabajadas en la semana. Realice un algoritmo para determinar el sueldo semanal de N trabajadores y, además, calcule cuánto pagó la empresa por los N empleados.

Ejercicio 17

Una empresa les paga a sus empleados con base en las horas trabajadas en la semana. Para esto, se registran los días que trabajó y las horas de cada día. Realice un algoritmo para determinar el sueldo semanal de N trabajadores y además calcule cuánto pagó la empresa por los N empleados.

Ejercicio 18

Hacer un programa que muestre un cronometro, indicando las horas, minutos y segundos.

Ejercicio 19

Realizar un ejemplo de menú, donde podemos escoger las distintas opciones hasta que seleccionamos la opción de "Salir".

Ejercicio 20

Mostrar en pantalla los N primero número primos. Se pide por teclado la cantidad de números primos que queremos mostrar.