

CLASE 4 : ESTRUCTURA DE REPETICION

1. Estructura de Control Iterativa, de Repetición o Ciclos
2. Ciclo Condicional (While - Repeat)
3. Ciclo Incondicional (For)
4. Ejemplos (While – Repeat - For)
5. Anidación de ciclos.

1. Estructura de Control Iterativa, de Repetición o Ciclos

Se utilizan para ejecutar sentencias en forma repetitiva. Si el número de repeticiones no se conoce de antemano (o sea antes que las repeticiones comiencen) se utiliza While o Repeat. En caso contrario, es conveniente implementar con For (aunque también es posible utilizar While o Repeat).

Es común que algunas de las sentencias que se ejecutan dentro del ciclo cuenten ocurrencias (de las repeticiones del ciclo o de determinados eventos) o acumulen sumas (o productos).

- Un contador es una variable entera que se incrementa (decrementa) en uno.
Cont:= Cont + 1;
- Un acumulador es una variable numérica que se incrementa (decrementa) en una cantidad variable.
Acum:=Acum + X;

2. Ciclo Condicional

La característica de estos ciclos es que están “condicionados” por una expresión lógica cuyo resultado determina la continuación o la finalización del ciclo.

Generalmente la expresión lógica utiliza operadores relacionales para enfrentar una variable a un valor preestablecido y puede presentarse de diferentes formas:

- el ciclo debe detenerse cuando una variable toma por lectura un valor “imposible”, o por cálculo un resultado límite (no se conoce la cantidad de repeticiones).
- Si el ciclo debe repetirse una determinada cantidad de veces que se conoce de antemano, se requiere evaluar si un contador de repeticiones ha alcanzado dicha cantidad.

Estas situaciones pueden combinarse mediante el uso de operadores lógicos, que potencian el control que se ejerce sobre el ciclo.

2.1. Ciclo While

While Expresión lógica do Sentencia;	While Expresión lógica do Begin Sentencia ₁ ; Sentencia ₂ ; Sentencia _n ; End;
---	---

Se evalúa la expresión lógica, si es verdadera ejecuta la sentencia (simple o compuesta) y vuelve a evaluar la expresión. Si es falsa el ciclo finaliza y continúa la ejecución de la sentencia que está físicamente abajo del ciclo.

Notar que:

- Evalúa la expresión lógica al comienzo, por lo tanto, si la primera vez resulta falsa no entra al ciclo y no se ejecuta ni una vez.
- Algunas de las variables que forman parte de la expresión lógica deben modificarse durante la ejecución del ciclo, para que la expresión pase de resultar verdadera a resultar falsa y así detener el ciclo.

2.1.1. En los siguientes ejemplos **si se conoce a-priori la cantidad de repeticiones** que deberá ejecutar el ciclo.

Para controlar la cantidad de veces que debe repetirse el ciclo, se requiere utilizar un contador de repeticiones, por lo tanto la condición que controla el ciclo While evalúa si el contador ha alcanzado dicha cantidad (ingresado por lectura), deteniendo o continuando el ciclo.

Ej 1) Leer el nombre y las 2 notas de N alumnos, calcular el promedio de cada uno e informar si aprobó o no (promedio ≥ 4).

Program Ej1;

Var

N1, N2, Cont : integer;

Prom : real;

Nomb : string;

Begin

Readln(N);

Cont:= 0;

While **Cont<N** do

Begin

Cont:= Cont + 1;

Readln(Nomb);

Readln(N1,N2);

Prom := (N1+N2)/2;

If Prom ≥ 4 then

Writeln(Nomb, 'APROBO')

else

Writeln(Nomb, 'DESAPROBO')

end

end.

El proceso debe repetirse N veces (para ésto, se utiliza una variable contadora CONT).

En cada repetición se debe leer le nombre del alumno (en Nomb), las dos notas (en N1 y N2), calcular el promedio (en Prom) e informar si aprobó o no.

2.1.2. En los siguientes ejemplos **no se conoce a-priori la cantidad de repeticiones** que deberá ejecutar el ciclo.

- **Caso 1:** La condición (expresión lógica) que controla el ciclo While verifica el valor ("imposible" o no) ingresado por lectura, deteniendo o continuando el ciclo.

Ej 2) Leer el nombre y las 2 notas de un conjunto de alumnos (*no se saben cuántos son*), calcular el promedio de cada uno e informar si aprobó o no (promedio ≥ 4). El nombre del alumno ' *** ' indica **fin de datos**.

```

.....
Aprob := 0;
Cont:= 0;
Readln (Nom);
While Nomb <> '***' do
    begin
        Readln (N1,N2);
        Cont:= Cont + 1;
        Prom := (N1+N2)/2;
        if Prom >=4 then
            begin
                Writeln (Nomb, 'APROBO');
                Aprob := Aprob + 1
            end
        else
            Writeln (Nomb, 'DESAPROBO');
        Readln (Nom);
    end;
Writeln ('El porcentaje de aprobados es', Aprob/Cont * 100:8:2);
.....

```

El primer nombre se **lee fuera del ciclo**, evaluando si es un dato válido. **Antes del fin del ciclo se vuelve a leer** el nombre del siguiente alumno para el control de la próxima iteración.

Ej3 - Leer un conjunto de números enteros cuya cantidad no se conoce de antemano (se sabe que son distintos de cero), (**fin de datos = 0**).

- a - mostrar los impares y además contar e informar cuantos son pares.
- b - calcular el promedio de positivos y negativos

```

Program Ej3a;
Var
    Num, ContP: word;
Begin
    ContP:=0;
    Write ('Ingresar un numero:');
    Readln(Num);
    While Num <> 0 do
        Begin
            If Num MOD 2 <> 0 then
                Writeln(num)
            Else
                ContP:= ContP +1;
            Writeln('Ingresar un numero:');
            Readln(Num);
        End;
    Writeln('La cantidad de pares es ', ContP);
End.

```

IMPORTANTE
Siempre las variables **contadoras** y **sumadoras** deben **inicializarse** en 0.

En el punto b) mientras los números no sean 0, por un lado se suman los positivos (en SumPos) y se cuentan (en ContPos) y por otro lado los negativos (en SumNeg y ContNeg). Por fin del proceso se muestran los promedios pedidos.

```

Program Ej3b;
Var
  Num, ContP, ContN, SumP: word;
  SumN: integer;
Begin
  ContP:=0; ContN:=0;
  SumP:=0; , SumN:=0;
  Write ('Ingresar un numero:'); Readln(Num);
  While Num <> 0 do
    Begin
      If Num > 0 then
        Begin
          ContP:= ContP +1; SumP:= SumP + Num;
        End
      Else
        Begin
          ContN:= ContN +1; SumN:= SumN + Num;
        End
      Write('Ingresar un numero:'); Readln(num);
    End;
  If ContP <> 0 then
    Writeln('Promedio positivos', SumP DIV ContP)
  Else
    Writeln('no ingresaron numeros positivos');
  If ContN <> 0 then
    Writeln('Promedio negativos', SumN DIV ContN)
  Else
    Writeln('no ingresaron numeros negativos');
End.

```

Tener en cuenta que si no hubieron negativos o positivos, hay que mostrar un informe coherente sabiendo que la división por 0 daría error.

Ej3 c) - Leer un conjunto de números enteros cuya cantidad no se conoce de antemano, calcular el promedio de positivos y negativos, además contar los ceros

En este ejercicio (parecido al Ej 3b) como todos los números hay que analizarlos porque participan de algún cálculo, no existe un valor de fin de datos, por lo tanto, la única solución posible es **preguntar al usuario** antes de cada ingreso si quiere continuar o no.

Antes de comenzar el ciclo se pregunta si hay más datos a ingresar (var. Res). Si es así, se ejecuta el ciclo, en caso contrario se continúa con las sentencias siguientes al while. Esta consulta al usuario vuelve a hacerse como última instrucción del cuerpo del ciclo mientras. Como se pide la cantidad de ceros, se llevará la cuenta en ContCeros.

```

Program Ej3c;
.....
ContPos:=0; SumPos:=0;
ContNeg:=0; SumNeg:=0; ContCeros:=0;
Writeln ('Quiere ingresar dato (S/N) ? '); Readln (Res);
While Res = 'S' do
  begin
    Readln (Nro);
    if Nro >0 then
      begin
        ContPos:=ContPos + 1;
        SumPos:=SimPos + Nro;
      end
    else
      if Nro <0 then
        begin
          ContNeg:= ContNeg+1;SumNeg:=SumNeg +Nro;
        end
      else
        ContCeros:= ContCeros+1;
    Writeln ('Quiere ingresar dato (S/N) ? '); Readln (Res)
  End;
writeln('Cantidad de ceros', ContCeros);
.....

```

Ej3 d)- Leer un conjunto de números enteros cuya cantidad no se conoce de antemano (se sabe que son distintos de cero), (**fin de datos = 0**). calcular el máximo (iniciando el máximo con un valor imposible o con el 1er valor leído)

Notar que las dos soluciones del punto c, difieren en la forma en la que la variable Max toma su primer valor. Si bien la segunda es más eficiente, no siempre es posible implementarla. Por ejemplo si se pide el máximo de los pares y los datos que se ingresan fueran: 91, 8, 45, 17, 20, 54, 0; iniciar con el primer valor no permitiría seleccionar 54 como máximo.

```

{inicia el máximo con un valor imposible}
Program Ej3c1;
Var
  Num, Max: integer;
Begin
Max:= -9999;
Write ('Ingresar un numero:');
Readln(Num);
While Num <> 0 do
  Begin
    If Num > Max then
      Max := Num;
    Write('Ingresar un numero:');
    Readln(num);
  End
Writeln('El maximo es', Max);
End.

```

```

{inicia el máximo con el 1er valor leído}
Program Ej3c2;
Var
  Num, Max: word;
Begin
Write ('Ingresar un numero:');
Readln(Num);
Max:= Num;
While Num <> 0 do
  Begin
    If Num > Max then
      Max := Num;
    Write('Ingresar un numero:');
    Readln(num);
  End
Writeln('El maximo es', Max);
End.

```

- **Caso 2:** La condición (expresión lógica) que controla el ciclo While verifica si el resultado de un cálculo ha alcanzado o no un valor límite (ingresado por lectura) deteniendo o continuando el ciclo.

Las expresiones lógicas en los ciclos condicionales pueden ser simples o compuestas utilizando operadores lógicos (and, or)

Ej4 – Leer un conjunto de números reales y sumarlos, detener el proceso cuando

a - la suma supere un cierto valor X, ingresado por teclado. Mostrar la cantidad de números sumados y la suma obtenida.

b - la suma supere un cierto valor X, o la cantidad de número leídos sea mayor a M (ambos valores ingresados por teclado). Mostrar la cantidad de números sumados y la suma obtenida.

<pre> Program Ej4a; Var Num, X, Sum: real; Cont: word; Begin Sum:=0; Cont:= 0; Writeln('Ingresar la cota de la suma:'); Readln(X); While Sum < X do Begin Writeln('Ingresar un numero:'); Readln(Num); Sum:= Sum + Num; Cont:= Cont +1; End; Writeln('Son', Cont, 'numeros'); Writeln('Su suma es', Sum:8:2); End.</pre>	<pre> Program Ej4b; Var Num, X, Sum: real; M, Cont: word; Begin Sum:=0; Cont:= 0; Writeln('Ingresar la cota de la suma y de los numeros:'); Readln(X, M); While (Sum <= X) and (Cont <= M) do Begin Writeln('Ingresar un numero:'); Readln(Num); Sum:= Sum + Num; Cont:= Cont +1; End; Writeln('Son', Cont, 'numeros'); Writeln('Su suma es', Sum:8:2); End.</pre>
---	---

IMPORTANTE:

Es importante que alguna de las variables que intervienen en la expresión lógica que controla el ciclo While, cambien su valor dentro del ciclo, en caso contrario el ciclo es infinito (no se detiene).

Si las siguientes sentencias faltaran en los ejemplos anteriores,

Readln(num); Sum:= Sum + Num; Cont:= Cont +1;

se obtendran **ciclos infinitos**.

Desarrollar programas Pascal que resuélva los problemas propuestos. Proponer juegos de datos y verificar su funcionamiento

1.-Dado un conjunto de números enteros distintos de cero, informar el porcentaje de pares e impares..

2.-Leer dos valores reales A y B ($A < B$), luego N números reales, calcular e informar el promedio de los que pertenecen al intervalo $[A, B]$

Ejemplos: A= - 2.5 y B=30.8

Los números son:

a) -1.3, 0, 50, -3.7, 5.5, 30.9 → promedio = 1.4

b) -13, 80.3, 50, -3.7, 55, 30.9 → no hay números en el intervalo

3.-Dado un conjunto de números enteros distintos de cero, informar el menor y el mayor de ellos.

4.-Dado un conjunto de números enteros distintos de cero, informar el mayor múltiplo de 3.

5.-Ingresar largo y ancho de un conjunto de rectángulos, para cada uno calcular e informar la superficie. Detener el proceso cuando se hayan ingresado N datos o el perímetro del rectángulo ingresado supere un valor X. (N y X son valores ingresados por teclado)

6.- Dado un conjunto de figuras geométricas y sus respectivas áreas:

- Figura ('T'=triángulo, 'C'=cuadrado, 'R'=rectángulo, 'F'=Fin de datos

- Area

Informar:

a) Porcentaje de cada tipo de figura.

b)Figura con mayor área (suponer única).

2.1.3. Ciclo de lectura de datos desde archivo

Un caso particular es el ciclo que lee datos de un archivo, la función booleana eof() que controla si se alcanzó o no el fin de archivo constituye la expresión lógica que controla el fin o la continuación del ciclo, respectivamente.

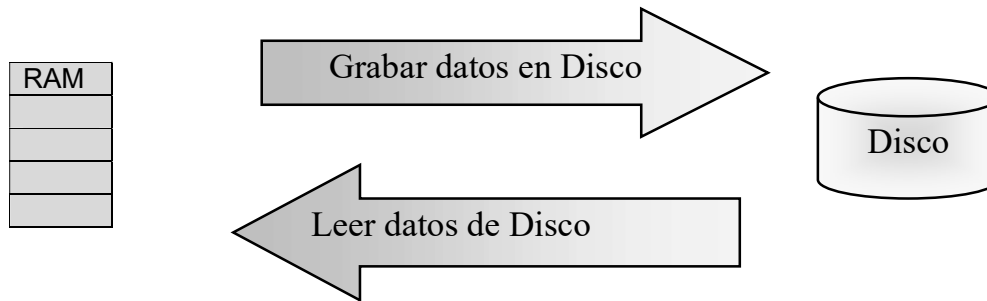
Tanto la entrada como la salida estándar, usan los archivos del sistema **input** y **output** por defecto, o sea que no es necesario especificar el teclado y la pantalla respectivamente.

Si se requiere lectura o escritura sobre archivo es necesario indicar el nombre del archivo (y en el caso de que éste no se encuentre en la misma ubicación del programa, también indicar unidad y carpeta que lo contiene), como así también la operación a realizar sobre el mismo (leer o escribir).

Consideraciones previas para lectura o escritura desde un almacenamiento secundario con archivos de texto (text)

Si el programador desea ingresar datos desde el disco rígido o removible (donde fueron almacenados previamente como un **archivo de texto**), debe indicar al programa el nombre del archivo que contiene los datos y su intención de leer datos desde ese archivo. De manera similar, puede crearse y almacenar datos en un archivo de texto. Los archivos de texto en

Pascal requieren de variables de tipo **text** . Estos archivos son secuenciales, es decir se acceden en forma progresiva desde el comienzo al final, esta característica los hace lentos y se recomienda su uso con información que será procesada en su totalidad.



El programa Pascal se ejecuta desde la memoria de acceso aleatorio (o RAM). Si el programador desea almacenar datos en un archivo de texto en el disco, estos deben ser grabados con sentencias específicas de escritura. Por otro lado, si desea leer o cargar datos ya previamente almacenados en el disco para llevarlos a la memoria, deben ser recuperados con otras sentencias específicas. Un archivo de texto es utilizado para leer o grabar, en forma excluyente o sea no admite ambas operaciones simultáneamente.

Los datos en un archivo de texto se almacenan usando el código ASCII, en el cual cada carácter es representado por un simple byte. Debido a que los archivos de texto utilizan el código ASCII, se pueden ver o imprimir. En este tipo de archivos, todos sus datos se almacenan como cadenas de caracteres, es decir, los números se almacenan con su representación ASCII y no su representación numérica, por lo tanto no se pueden realizar operaciones matemáticas directamente con ellos. Pero, si el dato es leído desde disco y almacenado en una variable numérica, puede operarse normalmente.

Luego de declarar la variable de tipo archivo en la sección

Var

NombreArch:text;

Deben utilizarse las siguientes sentencias, de cada uno se describe la sintaxis y su función:

Sentencia	Sintaxis	Función
assign	assign(nombre archivo Pascal, nombre archivo en el disco)	Enlaza la variable de tipo text con el archivo físico en el disco. Necesario para lectura/ escritura.
reset	reset(nombre del archivo Pascal)	Prepara al archivo para la lectura, ubicándose al comienzo.
read/ readln	read/readln(nombre archivo Pascal, lista de variables)	Se leen del archivo los valores y se almacenan en las variables indicadas en la lista.
rewrite	rewrite(nombre del archivo Pascal)	Prepara al archivo para la escritura, ubicándose al comienzo.
write/ writeln	write/writeln(nombre del archivo Pascal, lista de variables)	Se graban en el archivo los valores contenidos en las variables de la lista.
close	close(nombre del archivo Pascal)	Cierra el archivo.
eof	eof(nombre de archivo Pascal)	Función booleana que detecta el fin del archivo

Ha de tenerse cuidado con la sentencia **rewrite**, pues si el archivo no existe, se crea uno nuevo, pero si ya existe, *su contenido se pierde al sobrescribir los nuevos valores*.

Ej5 - A continuación ejemplos utilizando archivos de texto (se reformula el 3-b)

a - Ingresar números reales distintos de 0 desde un archivo Numeros.txt (no se sabe cuántos son), están grabados uno por línea. Calcular el porcentaje de negativos y positivos.

b - En el archivo Datos.txt se han grabado datos numéricos de tipo entero. Mostrar y sumar los positivos. Los números han sido grabados en una única línea.

```
program Ej5a;
var
  Dato, Sum:integer;
  Arch:text;
Begin
  Sum:=0;
  Assign (Arch,'Numeros.txt');      {vincula el archivo Numeros.txt con la variable Arch}
  Reset (Arch);                    {prepara el archivo para lectura}
  While not eof(Arch) do           {controla que no haya alcanzado el fin de archivo}
  Begin
    Readln(Arch,Dato);              {lee un número en del archivo y pasa a la línea siguiente}
    If Dato > 0 then
      begin
        Sum:= Sum + Dato;
        Writeln(Dato);
      end;
    End;
  Close (Arch);                    {cierra el archivo y desvincula Arch con Numeros.TXT}
  Writeln('la suma es', Sum);
End.
```

```
program Ej5b;
var
  Dato, Sum:integer;
  Arch:text;
Begin
  Sum:=0;
  Assign (Arch,'Datos.txt');        {vincula el archivo Datos.txt con la variable Arch}
  Reset (Arch);                    {prepara el archivo para lectura}
  While not eof(Arch) do
  Begin
    Read(Arch,Dato);                {lee un número del archivo y queda en la misma línea}
    If Dato > 0 then
      begin
        Sum:= Sum + Dato;
        Writeln(Dato);
      end;
    End;
  Close (Arch);
  Writeln('la suma es', Sum);
End.
```

Ej5 c - En el archivo Clima.txt se ha grabado información climática, en cada línea:
Fecha (dd/mm/aaaa) máxima mínima

Se pide leer el archivo y grabar en otro archivo Amplitud.txt las fechas que registren amplitud térmica mayor a 10.

```
program E4c;
var
  fecha:string[10];
  max, min:real;
  ArchE, ArchS:text; {se declaran dos archivos, se usara uno de entrada y otro de salida}
Begin
  Assign (ArchE,'Clima.txt'); Reset (Arch);           {prepara el archivo para lectura}
  Assign (ArchS,'Amplitud.txt'); Rewrite(ArchS);      {prepara el archivo para escritura}
  While not eof(ArchE) do
    Begin
      Readln(ArchE, fecha, max, min);                 {lee en una línea una medición }
      If max - min > 10 then
        Writeln(ArchS, fecha);                       {si la amplitud es mayor a 10 graba la fecha}
      End;
    Close (ArchE); Close (ArchS);                     {cierra los archivos}
  End.
```

Desarrollar programas Pascal que resuelva los problemas propuestos. Proponer juegos de datos y verificar su funcionamiento

7.-Rehacer el ejercicio 2. Ahora los datos están en el archivo 'Numeros.TXT' donde en la primer línea se encuentran los valores de A y B (con $A < B$), no se sabe cuántos números reales hay, calcular e informar el promedio de los que pertenecen al intervalo $[A, B]$

8.- Leer números desde un archivo e informar la cantidad de veces en que un número es igual al que le antecede.

9.- Rehacer el ejercicio 4 (leyendo desde archivo) Ingresar un valor entero X por teclado y luego:

- a) Informar el mayor múltiplo de X, poniendo un cartel aclaratorio si no existiera.
- b) Generar un Archivo 'Multiplo.TXT' con todos los múltiplos de X que hubiera.

2.2. Ciclo Repeat

REPEAT

Sentencia/s;
UNTIL expresión lógica;

Se ejecutan las sentencias que contiene la estructura y luego evalúa la condición, si es falsa repite el ciclo hasta que la condición sea verdadera, cuando el ciclo se detiene continúa la ejecución de la sentencia que esta físicamente abajo del mismo.

Notar que:

- Evalúa la condición después de ejecutar las sentencias que contiene, por lo tanto se ejecuta al menos una vez.
- Alguna de la variables que forman parte de la expresión lógica deben modificarse durante la ejecución del ciclo, para que la expresión pase de resultar falsa a resultar verdadera y así detener el ciclo.
- A diferencia del ciclo While que itera mientras la condición es verdadera, el Repeat itera si la condición resulta falsa (o sea se detiene cuando es verdadera).
- Para codificar una estructura While como una estructura Repeat, basta con negar la expresión lógica. Pero no es totalmente equivalente ya que la primera puede no ejecutarse nunca y la segunda se ejecuta al menos una vez.
- Las consideraciones hechas para la estructura While sobre las expresiones lógicas que controlan las repeticiones se aplican a la estructura Repeat (valor imposible por lectura, valor límite por cálculo, contador alcanza una cantidad establecida)

Se reprograman los ejemplos vistos anteriormente.

Ej6 - Leer un conjunto de

a - números enteros cuya cantidad no se conoce de antemano (se sabe que son distintos de cero), mostrar los impares,

b - números reales y sumarlos, detener el proceso cuando la suma supere un cierto valor X, ingresado por teclado. Mostrar la cantidad de números sumados y la suma obtenida.

<pre>{Caso1: Finaliza por valor 'imposible' ingresado } Program E6a; Var Num: word; Begin Write ('Ingresar un numero:'); Readln(Num); Repeat If Num MOD 2 <> 0 then Writeln(num); Writeln('Ingresar un numero:'); Readln(Num); Until Num = 0; End.</pre>	<pre>{ Caso 2 : Finaliza por valor calculado} Program Ej6b; Var Num, X, Sum: real; Cont: word; Begin Sum:=0; Cont:= 0; Writeln('Ingresar la cota de la suma:'); Readln(X); Repeat Writeln('Ingresar un numero:');Readln(Num); Sum:= Sum + Num; Cont:= Cont +1; Until Sum >= X ; Writeln('Son', Cont, 'numeros'); Writeln('Su suma es', Sum:8:2); End.</pre>
--	--

Se presentarán situaciones donde es más eficiente utilizar *Repeat* que la sentencia *While*:

- ✓ Para Validación de datos
- ✓ En Menú de Opciones

Ej7 - Leer un mes (1 .. 12) luego indicar a que trimestre pertenece (1 .. 4) . Validar la lectura hasta que ingrese un mes correcto.

```
Program Ej7;  
var  
  Mes:byte;  
Begin  
  Repeat  
    writeln('ingrese mes');  
    readln(Mes);  
  Until (1<=Mes) and (Mes<=12);  
  Writeln ('pertenece al trimestre ', (Mes-1) div 3  
+1)  
End.
```

Ej8 - Leer dos números reales, y luego utilizar en forma repetitiva un **menú** para obtener uno o más de los siguientes resultados: suma, resta, producto o división.

```
Program Ej8;  
var  
  x, y: real;  
  Op: char;  
Begin  
  writeln('ingrese dos números distintos de cero'); readln(x, y);  
  Repeat  
    writeln('S - Suma');  
    writeln('R - Resta');  
    writeln('P - Producto');  
    writeln('D - Division');  
    writeln('F - Finalizar');  
    writeln('Ingrese opcion');  
    Readln(Op);  
    Case op of  
      'S': writeln( x + y :8:2);  
      'R': writeln( x - y :8:2);  
      'P': writeln( x * y :8:2);  
      'D': Writeln ( x / y :8:2);  
    End;  
  Until Op = 'F';  
End.
```

menú

Se presenta un Menú de
Opciones y se elije una de ellas

Los dos últimos ejemplos muestran las aplicaciones más comunes de la estructura Repeat.

3. Ciclo Incondicional

La característica de este ciclo es que se ejecuta una cantidad “conocida” de veces, por lo tanto el control de las repeticiones no depende de una “condición” (expresión lógica), sino que es asumido por la propia estructura a través de la variable de control de ciclo (VC) que toma valores ordinales progresivos (o regresivos), determinados entre un límite inicial y otro final (VI y VF son expresiones ordinales).

Según sea la secuencia ascendente o descendente

3.1 FOR ascendente (For – to ; VI<=VF)

```
For VC := VI to VF do
    Sentencia;
```

VC: variable de control.
 VI: valor inicial
 VF: valor final
 Sentencia simple o compuesta

La variable VC toma el valor VI y se incrementa en 1 en cada iteración hasta alcanzar el valor VF,

Previo a cada iteración se verifica que se cumpla $VC \leq VF$. Estas acciones las realiza la estructura en forma automática.

El funcionamiento de la estructura For es equivalente a la siguiente estructura While

```
VC:= VI;
while VC<= VF do
    begin
        Sentencia;
        VC := VC + 1;
    End;
```

3.2 FOR descendente (For-downto ; VI>=VF)

```
For VC := VI downto VF do
    Sentencia;
```

El cambio que conlleva con respecto al anterior es que en cada iteración decrementa en 1 la variable de control (VC)

Previo a cada iteración se verifica que se cumpla $VC \geq VF$.

Consideraciones:

- El valor inicial y el valor final se evalúan al comienzo (una sola vez) y si el valor inicial es mayor (o menor en el downto) que el final no se ejecuta el ciclo.
- La variable de control de ciclo (VC) y los límites inicial y final (VI y VF) NO deben ser alterados dentro del ciclo For.
- La variable de control queda indefinida luego de terminar For.
- Es posible calcular la cantidad de repeticiones a partir de VI, VF y el Paso (1 ó -1).
 $\text{CantRepeticiones} = (VF - VI) * \text{Paso} + 1$
- Si se requiere un incremento diferente a 1 ó -1 debe implementarse a través de un ciclo While.

Ej9 – (corresponde al Ej3 y 4 codificado con For en vez de While)
 Ingresar N números reales
 a – Calcular el porcentaje de negativos y positivos.
 b – Calcular el mínimo.

<pre> Program Ej9a; Var X: real; i, Contp, N: word; Begin Writeln('Ingresar la cantidad de números:'); Readln(N); ContP:=0; For i:= 1 to N do Begin Writeln('Ingresar un numero:'); Readln(X); If X > 0 then ContP:= ContP +1; End; Writeln('% de positivos', ContP*100/N:8:2); Writeln('% de negativos',(N -ContP)*100/N:8:2); End.</pre>	<pre> Program Ej9b; Var Min, X: real; i, N: word; Begin Writeln('Ingresar la cantidad de números:'); Readln(N); Writeln('Ingresa el 1er. numero:'); Readln(Min); For i:= 2 to N do Begin Writeln('Ingresar un numero:'); Readln(X); If X <Min then Min:= X End; Writeln('El minimo es ', Min); End.</pre>
---	--

Desarrollar programas Pascal que resuelva los problemas propuestos. Proponer juegos de datos y verificar su funcionamiento.

10.-Leer un número, calcular e informar la suma de los números naturales hasta ese número.

11.-Leer N números enteros, calcular el mínimo de los impares y la cantidad de repeticiones del mismo.

12.- Un comercio realiza el 10% de descuento por ventas mayores a \$1000. Leer la cantidad de ventas realizadas, y luego el importe de cada una, calcular e informar el importe total que se otorgo en concepto de descuento y la cantidad de ventas que no tuvieron dicho beneficio. Cómo cambiaría el código si en vez de 10% y \$ 1000, dichos valores fueran variables (igual para todas las ventas).

13.- Para N personas, ingresar sexo (F : femenino, M : masculino) y edad (en años).





Calcular e informar:

a) Porcentaje de cada sexo.

b) Promedio general de las edades ingresadas, el promedio de edad de los varones y el promedio de edad de las mujeres. ¿Qué control puede hacer para darle robustez al algoritmo y evitar que cancele por división por 0?.

4. Ejemplos (While – Repeat - For)

Ej10 - La admisión a la facultad depende del resultado obtenido en los exámenes de tres materias. Se tienen las notas (0-100) correspondientes a N aspirantes a ingresar a la universidad, por cada uno :

-  Nombre
-  Nota Matemáticas
-  Nota Física
-  Nota Química

Ingresa el alumno que haya cumplido uno de estos tres requisitos :

C1- Las tres notas mayores o iguales a 40

C2- Matemáticas ≥ 60 y Química ≥ 80

C3- Física + Química ≥ 180

Desarrollar una solución que ingrese los N datos, liste los nombres de los ingresantes y la condición que cumplen, e informe el % de los mismos.

Se deben ingresar exactamente N datos, para ello se usa el ciclo For. En cada iteración se ingresan los datos de un aspirante, (nombre y 3 notas), y se analizan para determinar si cumple alguno de los tres requisitos (C1, C2 o C3). Considerar como precondition N > 0 .

Program Ej10;

Var

I, N, NM, NF, NQ, ContIng : byte;

Nomb: string[25];

begin

Readln(N);

ContIng:= 0;

For I := 1 to N do

begin

Readln(Nomb);

Readln(NM, NF, NQ);

If (NM ≥ 40) AND (NF ≥ 40) AND (NQ ≥ 40) then

Begin

writeln (Nomb, 'C1');

ContIng:= ContIng + 1;

end

else

if (NM ≥ 60) AND (NQ ≥ 80) then

Begin

writeln (Nomb, 'C2');

ContIng:= ContIng + 1;

end

else

if NF + NQ ≥ 180 then

Begin

writeln (Nomb, 'C3');

ContIng:= ContIng + 1;

End

End;

writeln ('El % de ingresantes es ', ContIng * 100/ N: 8:2);

end.

Ej11 - En un estudio de mercado (para el posicionamiento de un producto), se encuestó una cantidad no conocida de personas. Por cada una se obtuvo una de estas tres posibles respuestas:

N - No lo conoce

C - Lo conoce y lo consume

A - Lo conoce y no lo consume

Se pide desarrollar una solución que ingrese las respuestas (F es fin de datos), calcule e informe:


- ✓ % de personas que no lo consumen, sobre el total de encuestados
- ✓ % de personas que lo consumen, sobre el total de personas que lo conocen
- ✓ % de personas que no lo conocen, sobre el total de encuestados

Ejemplo: C C N C N A C N F

ContN= 3 NoConsumen = (3 + 1)*100/8

ContC= 4 ConsumenConocen = 4*100/(4 + 1)

ContA= 1 NoConocen = 3*100/8



Program Ej11;

Var

ContT, ContC, ContN, ContA : word;

Res : char;

Begin

ContC:= 0;

ContN:= 0;

ContA:= 0;

Writeln('C- lo conoce y consume ; N – no lo conoce ; A - lo conoce y no lo consume; F - Fin');

Readln(Res);

Res:= UPCASE (Res);

While Res <> 'F' do

begin

If Res = 'C' then

ContC:= ContC + 1

Else

If Res = 'N' then

ContN:= ContN + 1

Else

ContA:= ContA + 1;

Writeln('C- lo conoce y consume ; N – no lo conoce ; A - lo conoce y no lo consume; F – fin');

Readln(Res);

Res:= UPCASE (Res);

End;

ContT:= ContC + ContN + ContA;

writeln('% de personas que no lo consumen', (ContN + ContA)/ContT * 100 :8:2);

writeln ('% de personas que lo consumen', ContC/(ContC + ContA) * 100 :8:2); (*)

writeln ('% de personas que no lo conocen', ContN /ContT * 100 :8:2);

end.

{otra posibilidad}

Case Res of

'C' : ContC:= ContC + 1;

'N' : ContN:= ContN + 1;

'A' : ContA:= ContA + 1;

End;

(*) Si la encuesta solo registra datos de personas que no lo conocen, esta sería una división por cero. Por lo tanto debería controlarse esta situación utilizando una estructura alternativa (If).

Ej12 - Ingresar una secuencia de N números enteros (ordenados en forma descendente), informar cual es la máxima diferencia entre dos números consecutivos y qué posición ocupa el par dentro de la secuencia.

Ej : 25 19 11 9 3 -3, la máxima diferencia es 8 y corresponde al par 2

La diferencia se calcula sobre un par de números, por lo tanto deben estar en memoria dos valores consecutivos (Pri y Seg), se actualizan en cada repetición.

a- inicializar con un valor imposible	b-inicializar con el primer valor
<pre> Readln (N); Readln(Seg); DifMax := -99; For I := 2 to N do begin Pri := Seg; Readln(Seg); Dif := Pri - Seg; If Dif > Dif Max then begin Dif Max := Dif; PosPar := I - 1; end; End; writeln(DifMax, 'es la diferencia máxima y está en el par ', PosPar) </pre>	<pre> Readln (N); Readl(Pri, Seg); DifMax := Pri - Seg; PosPar:= 1; For I := 3 to N do begin Pri := Seg; Readln(Seg); Dif := Pri - Seg; If Dif > Dif Max then begin Dif Max := Dif; PosPar := I - 1; end; End; writeln(DifMax, 'es la diferencia máxima y está en el par ', PosPar);</pre>

Ej13 – Ingresar desde un archivo un conjunto de números naturales, mostrar los que tienen los factores primos (de 1 sólo dígito, 2, 3, 5, 7) coincidentes con los del primero leído.

Ejemplo: **12**, 18, 21, 36, 42, 15, 54 → sus factores son solo 2 y 3

Program Ej13;

Var

Fact2, Fact3, Fact5, Fact7 : Boolean;

N: word;

Arch : text;

Begin

Assign (Arch,'Datos.txt'); Reset (Arch); Read(Arch, N);

Fact2:= N mod 2 = 0;

Fact3:= N mod 3 = 0;

Fact5:= N mod 5 = 0;

Fact7:= N mod 7 = 0;

While not eof(Arch) do

Begin

Read(Arch,N);

If (Fact2 = (N mod 2 = 0)) and (Fact3 = (N mod 3 = 0)) and (Fact5 = (N mod 5 = 0)) and
(Fact7 = (N mod 7 = 0)) then

Writeln(N);

End;

Close(Arch);

End.

lee y establece los divisores
del primer número

Desarrollar un programa Pascal que resuelva:

14.-Leer las N ventas efectuadas por una farmacia Social. Por cada una se ingresa el monto y un código indicador del rubro:

L - venta libre

M - medicamentos (23% de descuento)

P - perfumería (promoción 10% de descuento).

A - accesorios (promoción, si el monto supera \$X corresponde un 5% de descuento)

Se pide informar:

- por cada venta el importe a pagar (con el descuento efectuado, si corresponde)
- Importe total bonificado en concepto de descuentos
- Total de operaciones e importe total en venta libre
- Monto de venta máximo y a que rubro pertenece

15.-En un entrenamiento de ciclismo cada participante, efectúa una vuelta a la pista y al terminar la misma se registra el *nombre* (** fin de datos), la *velocidad máxima alcanzada* y sus *pulsaciones*.

Se pide desarrollar un programa Pascal que lea de teclado los datos mencionados, obtenga e informe:

- La categoría de cada ciclista y su nivel de riesgo
- La cantidad de participantes en cada categoría
- El porcentaje de participantes cuyo nivel de riesgo es 3

Calcule CATEGORIA y NIVEL DE RIESGO según las siguientes tablas:

Pulsaciones	Nivel de Riesgo
hasta 120	1
Entre 121 y 160	2
Más de 160	3

Velocidad Máxima	Categoría
Más de 50km/h	A
Entre 40km/h y 49km/h	B
Menos de 40km/h	C

Como modificaría el código si se pidiera además

- Promedio de velocidades máximas
- Cantidad de corredores de categoría A con nivel de riesgo 1
- Informar categorías sin participantes

6. Ciclos Anidados

Ciclos Secuenciales

Son independientes, después que finaliza el primer ciclo, comienza el segundo.

Ejemplo:

```
Program CiclosSecuenciales;
Var
  Letra: Char;
  Nro:byte;
Begin
  For Nro:= 1 to 5 do
    Write (Nro:3);
  Writeln;
  For Letra:= 'a' to 'z' do
    Write (letra:3);
  End.
```

1	2	3	4	5	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Total de repeticiones 5 + 26

Ciclos Anidados

Es cuando dentro de un ciclo, hay una sentencia que es otro ciclo.

Son dependientes, por cada iteración del ciclo externo se ejecuta íntegramente el ciclo interno.

Ejemplo:

```
Program CiclosAnidados;
Var
  Letra: Char;
  Nro:byte;
Begin
  For Nro:= 1 to 5 do
    begin
      Write (Nro:3);
      For Letra:= 'a' to 'z' do
        Write (letra:3);
      Writeln;
    End;
  End.
```

1	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
2	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
3	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
4	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
5	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z

Total de repeticiones 5 * 26

Un ciclo puede contener otro ciclo, se dice que están “anidados”, estos encajes se establecen con los tres ciclos vistos. Un For puede contener otro For, un While o Repeat, todas las combinaciones están permitidas y deben responder al control de las repeticiones que requiere la solución.

Ejemplo 14.- Utilizando la solución del ejercicio 1 (Leer un número, calcular e informar la suma de los números naturales hasta ese número), modificarla para que realice sobre un conjunto cuya cantidad no se conoce. Dar por terminada la entrada cuando el número sea 0.

Al final del proceso informar cuantos números se procesaron y cuál fue el máximo.

Program Ej14;

Var

Nro, i, Max, Cont : byte;

Sum : word;

Begin

Max:=0; Cont :=0;

Write ('Ingrese un numero, 0 para finalizar'); Readln(Nro);

While Nro <> 0 do

Begin

Sum :=0;

For i:=1 to Nro do

Sum := Sum + i;

Writeln('La suma de los primeros ',Nro,' naturales es:', Sum);

Cont := Cont +1;

If Nro > Max then

Max := Nro;

End;

Writeln('Se procesaron ',Cont,' números y el máximo fue ',Max);

End.

Ej15 – Se tiene un conjunto de números primos y por cada uno de ellos N números no primos, este agrupamiento se mantiene hasta que ingresa un cero. O sea se repite el siguiente esquema:

- P (número primo ó 0 que indica fin de datos)
- N (cantidad entera)
- y a continuación N números no primos.

Se pide ingresar dichos datos, calcular e informar:

- Para cada primo porcentaje de múltiplos entre los N no primos.
- Número primo con mayor porcentaje de múltiplos.
- Cantidad de primos sin múltiplos.

Ejemplo: a.

11, 5, 24, <u>44</u> , 34, <u>121</u> , 98	→ 40%
3, 3, 35, 25, 14	→ 0%
5, 4, <u>15</u> , <u>100</u> , <u>30</u> , <u>85</u>	→ 100%
2, 8, <u>4</u> , 63, <u>18</u> , 27, <u>32</u> , 21, <u>6</u> , 81	→ 50%
0	

b. 5

c. 1

```

Program Ej15;
Var
  P, N, NP, ContM, ContSM, i, Max, MaxP : word;
  Porc. Real;
Begin
  ContSM:= 0; Max:=0;
  Writeln('Ingrese un número primo'); Readln(P);
  While P <> 0 do
    begin
      Writeln('Ingrese la cantidad de no primos'); Readln(N);
      ContM:= 0;
      For i := 1 to N do
        Begin
          Writeln('Ingrese un número no primo'); Readln(NP);
          If NP mod P =0 then
            ContM:= ContM +1;
          End;
        Porc:= ContM*100/N;
        Writeln('el % de múltiplos de ', P, 'es',Porc: 8:2);
        If ContM = 0 then
          ContSM:= ContSM + 1
        Else
          If Porc > Max then
            Begin
              Max:= Porc; MaxP:= P;
            End;
          Writeln('Ingrese un número primo'); Readln(P);
        End;
      writeln (ContSM, 'numeros primos no registraron multiplos');
      if Max <> 0 then
        writeln ('El numero primo con mas múltiplos es', MaxP)
      else
        writeln ('ningun numero primo tuvo múltiplos ');
    end.

```

Ej16 – N camiones tiene asignados una cantidad no conocida de bultos para ser cargados (no necesariamente podrán ser cargados en su totalidad). Cada camión tiene una capacidad en Tns. y cada bulto un peso en kgs. Los datos descriptos están organizados de la siguiente forma para cada uno de los N camiones:

- Capacidad del camión
- y a continuación el peso de cada bulto en kgs. (0 es fin de bulto para dicho camión)

Se pide ingresar dichos datos, calcular e informar:

- a. Para cada camión si pudo cargar todos los bultos.
- b. Peso total de los bultos que quedaron sin poder cargarse (en todos los camiones)
- c. Cuantos camiones completaron su capacidad en un 90 % o más (sin importar si quedaron o no bultos sin cargar).

```

Program Ej16;
Var
  N, Cont90, i, : word;
  Suma, Capacidad, Bulto, SinCargar: Real;
Begin
  Cont90:= 0; SinCargar:=0;
  Writeln('Ingrese cantidad de camiones'); Readln(N);
  For i := 1 to N do
    begin
      Writeln('Ingrese capacidad del camión'); Readln(Capacidad);
      Suma:= 0;
      Repeat
        Writeln('Ingrese peso del bulto'); Readln(Bulto);
        Bulto:= Bulto/1000;
        Suma:= Suma + Bulto;
      Until (Bulto = 0) or (Suma > Capacidad);
      If Suma > Capacidad then
        Begin
          Suma:= Suma - Bulto; {queda lo que realmente cargó}
          Writeln('No se cargaron todos los bultos para la capacidad', Capacidad);
          While Bulto <> 0 do
            Begin
              SinCargar:= SinCargar + Bulto;
              Writeln('Ingrese peso del bulto'); Readln(Bulto);
              Bulto:= Bulto/1000;
            End;
          End
        Else
          Writeln('Se cargaron todos los bultos para la capacidad', Capacidad);
      If Suma >= 0.9 * Capacidad then {completó su capacidad en un 90% o más}
        Cont90:= Cont90 + 1;
      End;
    Writeln (Cont90, 'camiones completaron el 90% o más de su capacidad');
    Writeln ('quedaron ', SinCargar , 'Tns.' );
  end.

```

Desarrollar un programa Pascal que resuelva:

Ej 16.-En una competencia participan N deportistas que se identifican con su n° de Documento y lanzan una JABALINA.

Una vez lanzada, se registra el documento y la Distancia del lanzamiento.

Se requiere:

- Conocer el Documento del Ganador y su Distancia.
- Conocer la Distancia Media de la prueba.

Ej 17.-En un sorteo de lotería se extraen 6 números enteros entre 1 a 40.

Se pide ingresar los datos de cada uno de los últimos N sorteos e informar:

- La cantidad de sorteos que han tenido al menos un número impar.
- El número par más alto de cada sorteo, indicar cuándo no haya habido

Ejemplo:

Si N fuera 3 y los datos de los sorteos fueran:

2 5 1 19 3 12

3 13 9 1 7 29

2 8 1 14 37 16

Debería responder

a) En 3 sorteos hubo al menos un número impar

b) El número par más alto del sorteo 1 fue 12

No hubo pares en el sorteo 2

El número par más alto del sorteo 3 fue 16

Ej18.-Leer un conjunto de números enteros, la presencia de ceros intermedios indican fin de un subconjunto y comienzo del siguiente (el fin de datos está indicado con dos ceros consecutivos). Se pide calcular e informar: total de números para cada subconjunto y orden del conjunto más numeroso.

Ejemplo : 8, 9, 0, 7, 2, 7, 4, 0, 5, 6, 1, 0, 0

Respuesta:

Subconjunto	Cantidad
1	2
2	4
3	3

Subconjunto con más elementos 2

Ej19.-Leer un conjunto de N números enteros, se sabe que hay subconjuntos formados por números que difieren del anterior en ± 1 , el no cumplimiento de esta relación indica fin de un subconjunto y comienzo del siguiente. Se pide calcular e imprimir la cantidad de subconjuntos.

Ejemplo : N= 9

6, 7, 8, 6, 5, 6, 7, 1, 2

Respuesta :

Son 3 subconjuntos

Son 3 subconjuntos