TEMAS de la CLASE

1. Concepto de Estructura de Control
2. Clasificación de las estructuras de control
   - Selección Simple
   - Selección Múltiple
   - Iterativa Precondicional
   - Iterativa Postcondicional
   - Repetitiva

Concepto de Estructura de control

Todos los lenguajes de programación tienen un conjunto mínimo de instrucciones que permiten especificar el **control** del algoritmo que se quiere implementar.

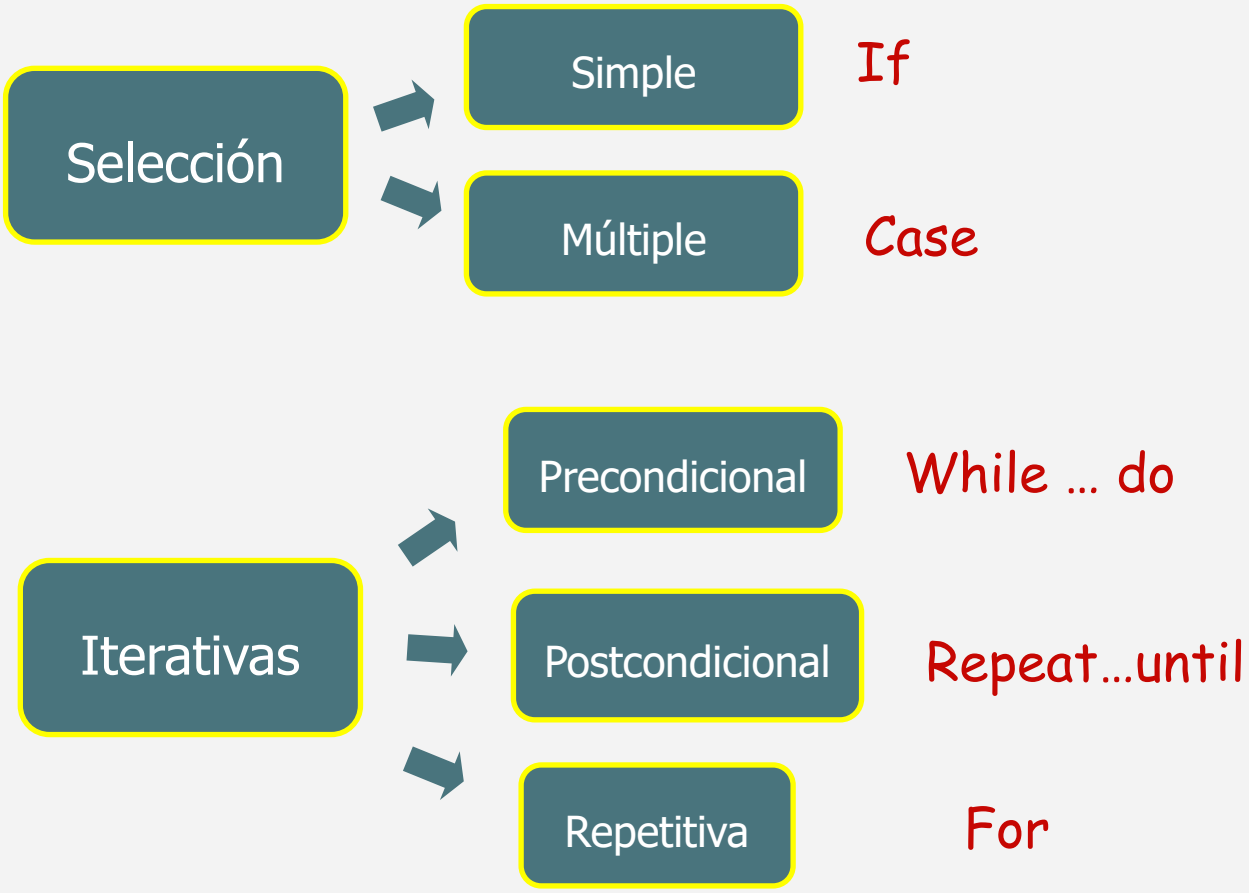
Veremos que este conjunto **debe contener como mínimo:**

- ✓ **Selección**
- ✓ **Iteración**

¿Para qué nos sirven las estructuras de control?

Las estructuras de control permiten modificar el flujo de ejecución de las instrucciones de un programa.

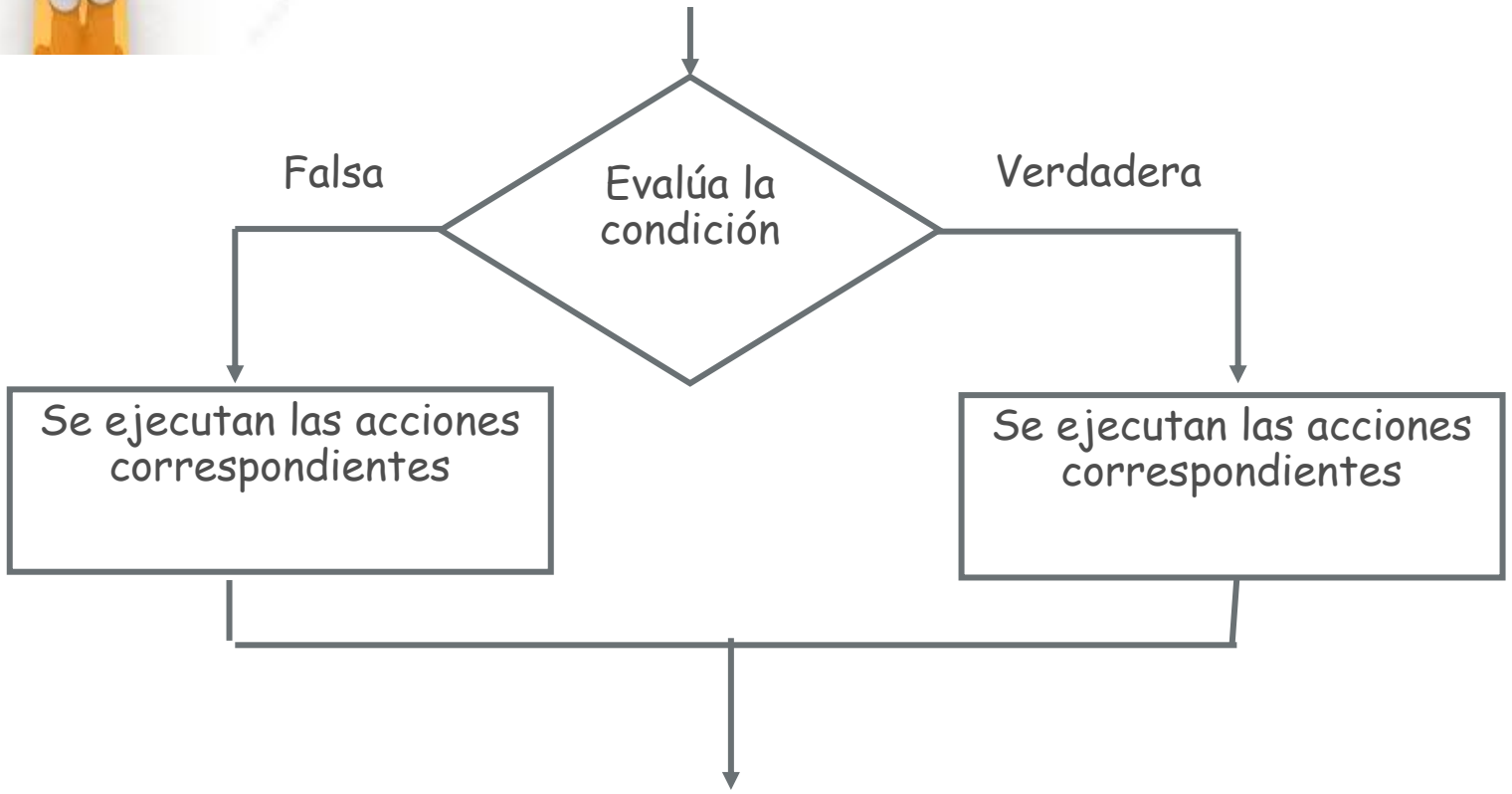
Clasificación de las Estructuras de control



Estructura de control: Selección simple (en Pascal)

Puede ocurrir que en un problema real sea necesario elegir una alternativa entre 2 posibles.

La estructura de selección simple se representa simbólicamente:



Estructura de control: Selección simple (en Pascal)

```
If  (condición)  then begin
                    Acciones_por_Condición_Verdadera;
                 end
                 else begin
                    Acciones_por_Condicion_Falsa;
                 end;
```

Caso Especial (sin else)

```
If  (condición)  then begin
                    Acciones_por_Condición_Verdadera;
                 end;
```

Estructura de control: Selección simple (en Pascal)

Ejercicio 1: Se quiere comprar un terreno solo si su superficie es mayor a 900 mts 2.

Implementar un programa que calcule la superficie del terreno cuyas dimensiones se leen e informe si lo compra o no.

Algoritmo:

- *Leer dimensiones*
- *Calcular superficie*
- *Si (superficie > 900) entonces*
- *– Mostrar Comprar*
- *sino*
- *– Mostrar No comprar*

```
Program Ejercicio1;
  var frente, fondo: real;
        sup: real;
Begin
  readln (frente); readln (fondo);
  sup := frente * fondo;
   if (sup > 900) then
                        write ('Comprar Terreno');
                    else
                        write ('No Comprar Terreno);
End.
```

Estructura de control: Selección simple (en Pascal)

Ejercicio 2: Escribir un programa que permita que se ingrese un tipo de impuesto y en función del tipo de impuesto se muestre a que caja debe dirigirse. Unicamente se cobran impuestos de los tipos "I" en la caja 1 y "M" en la caja 2.

Algoritmo:

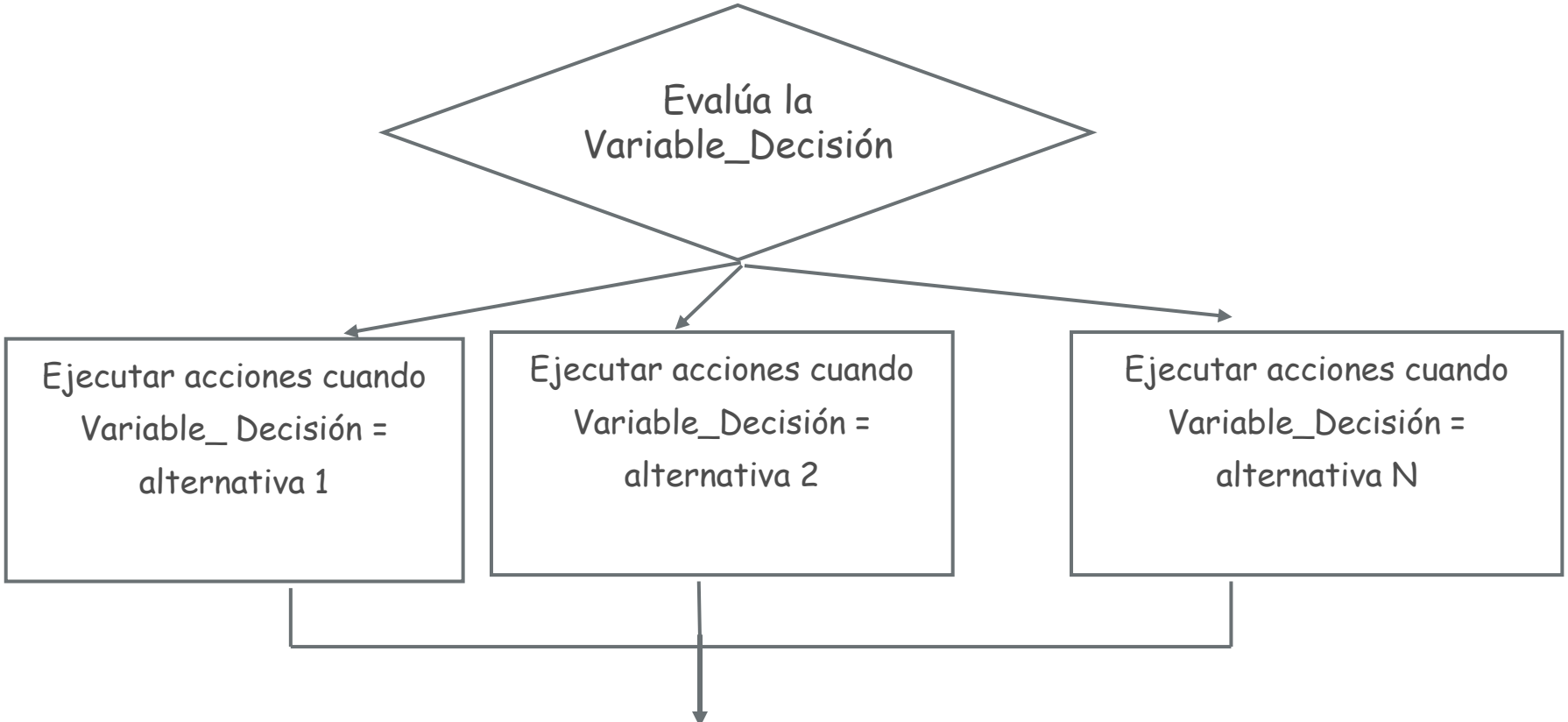
- *Leer impuesto*
- *Si es I -> Caja 1*
  sino -> Caja 2

```
Program Ejercicio2;
  var impuesto: char;

Begin
 readln (impuesto);
 if (impuesto = 'I') then
                      write ('Ir a Caja 1');
                    else
                      write ('Ir a Caja 2');
End.
```

Estructura de control: Selección múltiple

Puede ocurrir que en un problema real sea necesario elegir una alternativa entre varias posibles en función del problema a resolver.

La estructura de selección múltiple se representa simbólicamente:



Estructura de control: Selección múltiple (en Pascal)

Puede ocurrir que en un problema real sea necesario elegir una alternativa entre varias posibles en función del problema a resolver.

```
Case variable decision of
   alternativa 1 : Acciones;
   alternativa 2 : Acciones;
     ……..
   else otras acciones
End;
```

Estructura de control: Selección múltiple (en Pascal)

Ejercicio 3: Escribir un programa que permita que se ingrese un tipo de impuesto y en función del tipo de impuesto se muestre a que caja debe dirigirse. Se cobran impuestos de los tipos "I", "M", "P" y "S" , en las cajas 1 a 4 respectivamente. Para otros impuestos no hay cajas disponibles.

Algoritmo:

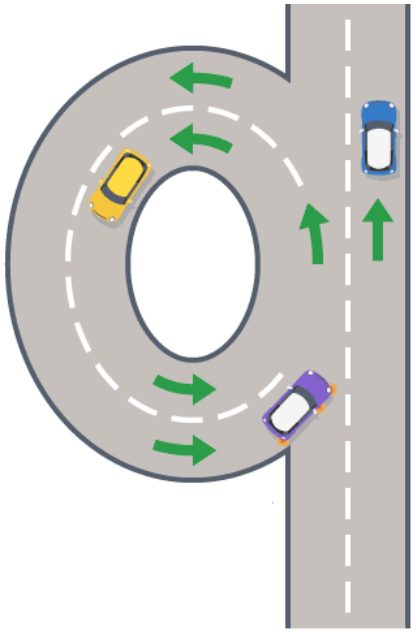
Leer impuesto
Verificar tipo
Si es I -> Caja 1
sino si es M -> Caja 2
sino si es P …

```
Program Ejercicio3;
   var impuesto : char;

begin
  readln (impuesto);
  case  impuesto of
   'I': writeln ('Dirigirse a Caja 1');
   'M': writeln ('Dirigirse a Caja 2');
   'P': writeln ('Dirigirse a Caja 3');
   'S': writeln ('Dirigirse a Caja 4');
   else write ('Acá no se cobra ese impuesto')
  end;
 end.
```

Estructuras de Control Iterativas

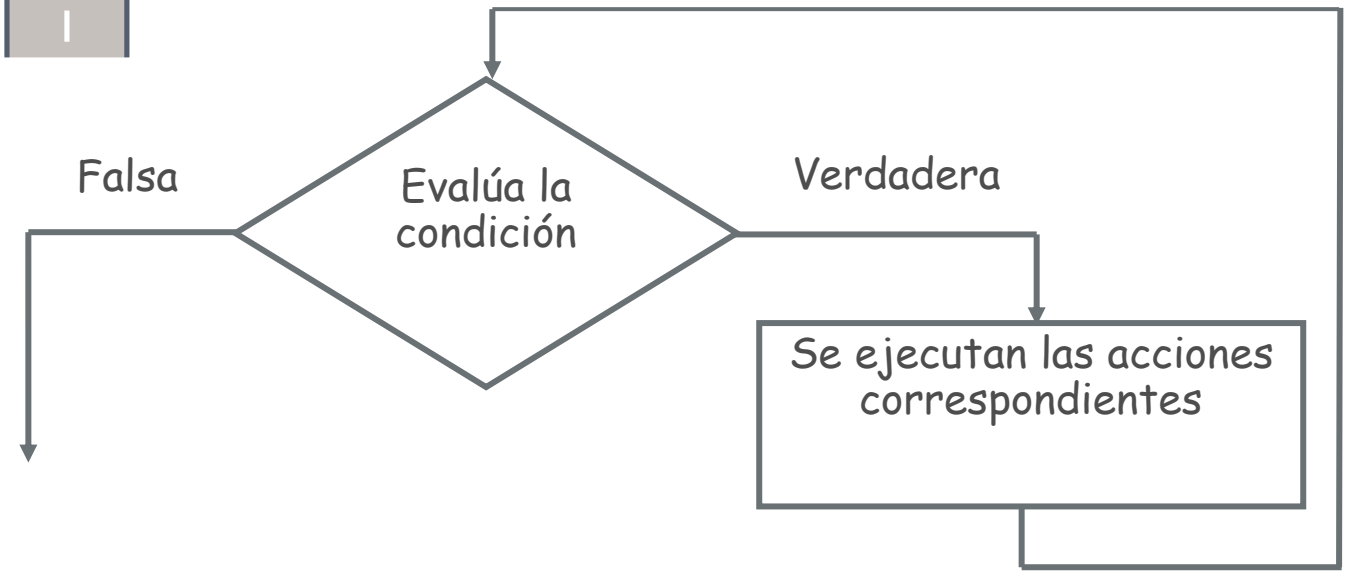
- Puede ocurrir que se desee ejecutar un bloque de instrucciones desconociendo el número exacto de veces que se ejecutan.
- Para estos casos existen en la mayoría de los lenguajes de programación estructurada las **estructuras de control iterativas condicionales.**
- Como su nombre lo indica las acciones se ejecutan dependiendo de la evaluación de la condición.
- Estas estructuras se clasifican en **pre-condicionales** y **post-condicionales**.

Estructura de Control: Iterativa precondicional



- Las **estructuras de control iterativas precondicionales** primero evalúan la condición y si es verdadera se ejecuta el bloque de acciones. Dicho bloque se puede ejecutar 0, 1 ó más veces.

- Importante: el valor inicial de la condición debe ser conocido o evaluable antes de la evaluación de la condición.



Estructura de Control: Iterativa precondicional (en Pascal)

- Las **estructuras de control iterativas precondicionales** primero evalúan la condición y si es verdadera se ejecuta el bloque de acciones. Dicho bloque se puede ejecutar 0, 1 ó más veces.

- **Importante**: el valor inicial de la condición debe ser conocido o evaluable antes de la evaluación de la condición.

```
While (condición) do
 begin
    Acciones a realizar
 end;
```

Estructura de Control: Iterativa precondicional (en Pascal)

En el ejercicio de los terrenos, si se piensa que hay varios terrenos y se quiere informar la superficie de cada uno, ¿Cómo lo resolveríamos sabiendo que si el largo es 0 se termina el cálculo?

Ejercicio 4: Escribir un programa que permita ingresar los datos (frente y fondo) de los terrenos e informe la superficie de cada uno. El ingreso de los datos del terreno finaliza cuando se ingresa el fondo igual a 0.

Algoritmo:

- *Leer fondo*
- *Mientras el fondo sea distinto de 0*
  - *Leer frente*
  - *Calcular superficie*
  - *Mostrar superficie*
  - *Leer fondo*

Fin mientras

```
Program Ejercicio 4;
   var frente, fondo, sup : real;

begin
 readln (fondo);
 while (fondo <> 0) do begin
    readln (frente);
    sup:= frente * fondo;
    writeln ('La superficie es: ', sup);
    readln (fondo);
 end;
end.
```

Estructura de Control: Iterativa precondicional (en Pascal)

Si volvemos al caso del pago de impuestos y ahora se piensa que hay varios clientes que van a pagar un impuesto…

Ejercicio 5: Escribir un programa que permita ingresar el tipo de impuesto a pagar por cada cliente y en función del tipo de impuesto se muestre a que caja debe dirigirse. El ingreso de impuestos finaliza cuando se ingresa el impuesto '@'.

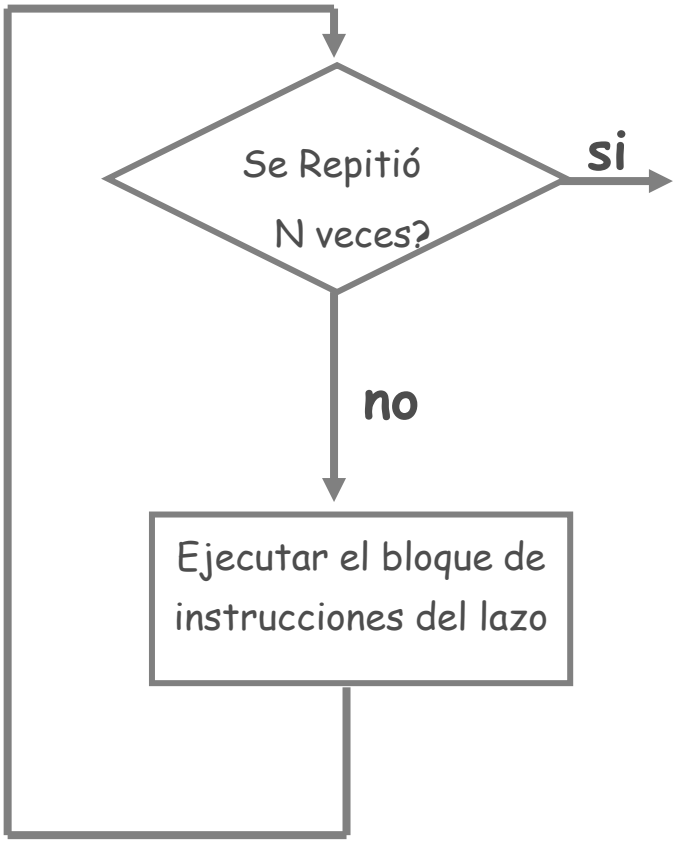
Algoritmo:

- *Leer impuesto*
- *Mientras el impuesto sea distinto de '@'*
  - *Verificar tipo*
  - *Si es I -> Caja 1*
    sino si es M -> Caja 2
    sino si es P …
  - *Leer impuesto*

Fin mientras

```
Program Ejercicio 5;
    var impuesto : char;

begin
 readln (impuesto);
 while (impuesto <> '@') do begin
  case  impuesto of
    'I': writeln ('Dirigirse a Caja 1');
    'M': writeln ('Dirigirse a Caja 2');
    'P': writeln ('Dirigirse a Caja 3');
    'S': writeln ('Dirigirse a Caja 4');
     else write ('Acá no se cobra ese impuesto')
   end;
   readln (impuesto);
 end;
end.
```

Estructura de control: Iterativa repetitiva



- Puede ocurrir que se desee ejecutar un bloque de instrucciones conociendo el número exacto de veces que se ejecutan. Es decir ***repetir N veces un bloque de acciones.***

- Este número de veces que se deben ejecutar las acciones es fijo y conocido de antemano.

Se muestra el diagrama esquemático de la ***repetición***

Estructura de control: Iterativa repetitiva

```
For indice := valor_inicial to valor_final do
    begin
        Acciones a realizar
    End;
```

Acerca de la variable índice en Pascal:

- La variable de control debe ser de tipo ordinal (entero, boolean, char)

- NO debe modificarse dentro del lazo.
- Los incrementos ó decrementos y testeos son implícitos.
- Al terminar el ciclo, la variable índice no tiene un valor definido (su uso se limita a la repetición).

Estructura de control: Iterativa repetitiva

```
For indice := 'A'  to  'H' do
   begin
      Acciones a realizar
   End;
```

```
For  índice := false to true do
   Begin
        Acciones a realizar
    End;
```

```
For índice := 18 to 18 do
   begin
     Acciones a realizar
   End;
```

```
For índice := 20 downto 18 do
    begin
      Acciones a realizar
    End;
{este bloque se ejecuta 3 veces}
```

¿De qué tipo es la variable índice?

¿Qué valores toma la variable índice?

Estructura de control: Iterativa repetitiva

Ejercicio 6: Escribir un programa que informe la superficie total de una parcela de terrenos cuyas dimensiones se leen de teclado. La parcela se compone de 15 terrenos.

Algoritmo:

Inicializar total de sup.

Repetir 15 veces

Leer frente y fondo

Calcular superficie

Sumar al total

Mostrar Resultado

```
Program Ejercicio 6;
var
  i : integer;
  total, sup, frente, fondo: real

begin
  total := 0;
  for i:= 1 to 15 do begin
     read (frente);
     read (fondo);
     sup:= frente * fondo;
     total := total + sup;
  end;
  write (total);
end.
```

Estructura de control: Iterativa repetitiva

Ejercicio 7: Reescribir el programa anterior para que además informe la superficie máxima calculada.

Algoritmo:

Inicializar total de sup.

Inicializar valor máximo

Repetir 15 veces

Leer frente y fondo

Calcular superficie

Sumar al total

Si superficie > valor máximo

actualizar valor máximo

Mostrar Resultados

```
Program Ejercicio 7;
var
  i : integer;
  total, sup, frente, fondo, max: real;

begin
  total := 0;
  max := 0;
  for i:= 1 to 15 do begin
     read (frente);
     read (fondo);
     sup:= frente * fondo;
     total := total + sup;
     if sup > max then max := sup;
  end;
  write ('Superficie parcela: ',total);
  write ('Superficie máxima: ', max)
end.
```

Estructuras de Control

Ejercicio 8: Escribir un programa que informe la suma de 20 números enteros que se leen de teclado y además que informe la cantidad de nros pares entre los nros leídos..

Algoritmo:

Inicializar la suma

Inicializar contador de pares

Repetir 20 veces

Leer un número

Sumar el número leído

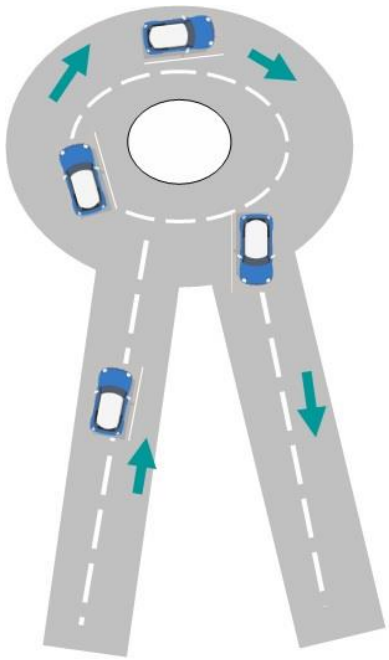
Verificar si es par

incrementar contador

Mostrar Resultados

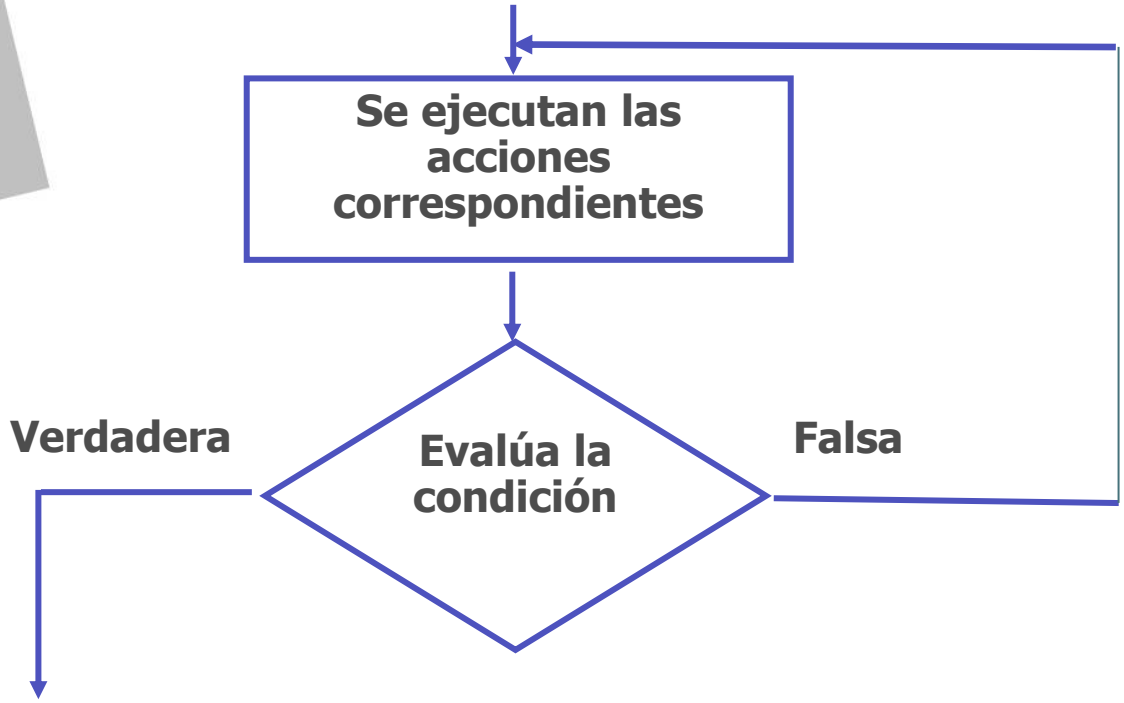
```
Program Ejercicio 8;
var
  suma, cont_pares, num, i : integer;

begin
  suma := 0;
  cont_pares := 0;
  for i:= 1 to 20 do begin
    read (num);
    suma := suma + num;
    if num mod 2 =0 then
       cont_pares := cont_pares + 1
  end;
  write (suma, cont_pares);
end.
```

Estructura de Control: Iterativa postcondicional



- Las **estructuras de control iterativas postcondicionales** primero ejecutan el bloque de acciones y luego evalúan la condición. A diferencia de la estructura iterativa pre-condicional, el bloque de acciones se ejecuta 1 ó más veces.

- Importante: el valor inicial de la condición debe ser conocido o evaluable antes de la evaluación de la condición.

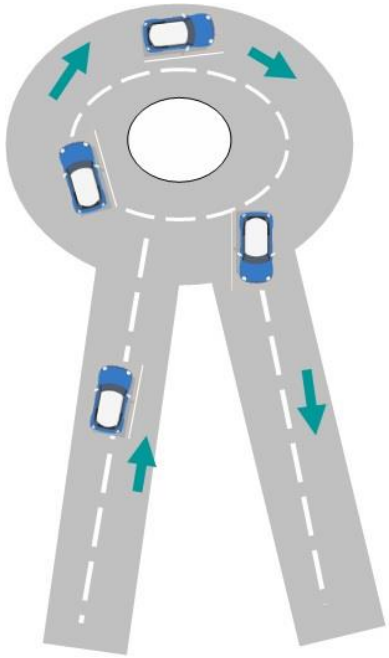


Estructura de Control: Iterativa postcondicional

- Las **estructuras de control iterativas postcondicionales** primero ejecutan el bloque de acciones y luego evalúan la condición. A diferencia de la estructura iterativa pre-condicional, el bloque de acciones se ejecuta 1 ó más veces.

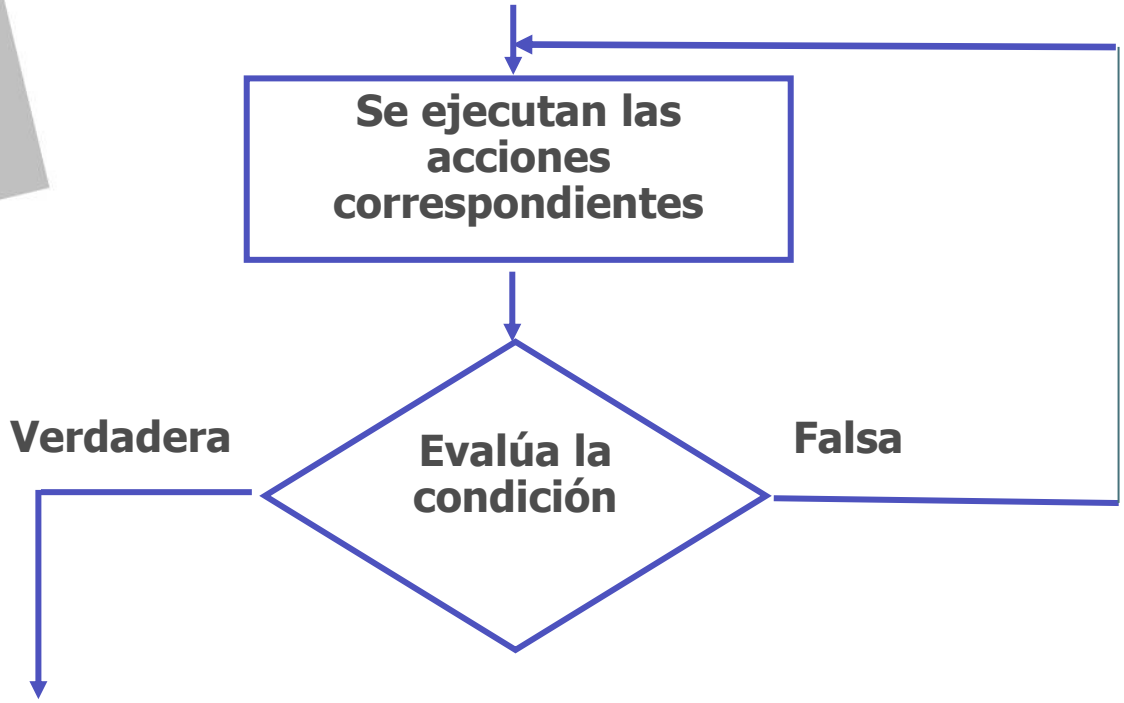
```
repeat
        Acciones a realizar
until ( condicion);
```

Estructura de Control: Iterativa postcondicional (en Pascal)



- Las **estructuras de control iterativas postcondicionales** primero ejecutan el bloque de acciones y luego evalúan la condición. A diferencia de la estructura iterativa pre-condicional, el bloque de acciones se ejecuta 1 ó más veces.

- Importante: el valor inicial de la condición debe ser conocido o evaluable antes de la evaluación de la condición.



Estructura de Control: Iterativa postcondicional (en Pascal)

Ejercicio 9: Escribir un programa que informe las superficies de terrenos cuyas dimensiones se leen de teclado. El ingreso de datos finaliza cuando se ingresa un terreno cuya superficie supera los 1000 mt2.

```
Algoritmo
   Repetir
        Leer frente
        Leer fondo
        Calcular Superficie
        Mostrar Superficie
    Hasta que Superficie > 1000
Fin
```

```
Program Ejercicio 9;
  var frente, fondo, sup: real;

Begin
 repeat
   readln(frente);
   readln (fondo);
   sup := frente * fondo;
   write ('Superficie = ', sup);
 until (sup > 1000)
End.
```

Estructura de Control: Iterativa postcondicional (en Pascal)

Ejercicio 10: Escribir un programa que lea una secuencia de caracteres terminada en punto. Se debe informar la cantidad de caracteres '%' que aparecen en la secuencia, la cantidad total de caracteres leídos y mostrar una línea que contenga tantos '*' como '%' se haya contado.

Ejemplo:

asdfoi&%(uewr9832/()?237/&ASD %%.

Cantidad de '%' = 3

Cantidad de caracteres = 37

***

Estructura de Control: Iterativa postcondicional (en Pascal)

Ejercicio 10: Escribir un programa que lea una secuencia de caracteres terminada en punto. Se debe informar la cantidad de caracteres '%' que aparecen en la secuencia, la cantidad total de caracteres leídos y mostrar una línea que contenga tantos '*' como '%' se haya contado.

```
Algoritmo
Inicializar contador de caracteres
Inicializar contador de '%'
Leer caracter
Mientras carácter <> '.'
    incrementar en 1 el contador de caracteres
    si carácter leído es ='%'  entonces
                                    incrementar en 1 el contador de '%'
    leer carácter
Informar la cantidad de '%' y la cantidad total de caracteres
Repetir contador de '%'
   mostrar '*'
```

Analicemos los datos...

Estructura de Control: Iterativa postcondicional (en Pascal)

Ejercicio 10: Escribir un programa que lea una secuencia de caracteres terminada en punto. Se debe informar la cantidad de caracteres '%' que aparecen en la secuencia, la cantidad total de caracteres leídos y mostrar una línea que contenga tantos '*' como '%' se haya contado.

Algoritmo
Inicializar contador de caracteres
Inicializar contador de '%'
Leer caracter
Mientras carácter <> '.'
    incrementar en 1 el contador de caracteres
    si carácter leído es ='%'
      entonces incrementar en 1 el contador de '%'
    leer carácter
Informar la cantidad de '%' y la cantidad total de caracteres
Repetir contador de '%'
   mostrar ''*

```
Program Ejercicio10;
var i, contadorcar, cont%:integer;
      car: char;
Begin
  contadorcar:=0;
  cont%:=0;
  readln (car);
  While ( car <>'.') do begin
    contadorcar:=contadorcar+1;
   if car='%' then cont%:=cont%+1;
    readln (car);
  end;
  writeln ('Cantidad de %= ', conta%);
  writeln ('Cant.de caracteres= ', contadorcar);
  for i:= 1 to cont% do
      write ('*');
end.
```

Estructuras de Control: Resumen

Todos los lenguajes de programación tienen un conjunto mínimo de instrucciones que permiten especificar el **control** del algoritmo que se quiere implementar.

Decisión (if)

iteración

- Precondicional (while)
- Selección múltiple (case)
- Repetición (For)
- Postcondicional (repeat ... until)

EJERCITACIÓN CLASE 2

Un centro de deportes quiere procesar la información de sus clientes y los 4 tipos de actividades que ofrece: 1 (Musculación), 2 (Spinning), 3 (Cross Fit) y 4 (Libre).

Para ello, se debe leer y guardar el precio mensual de cada actividad. Además, se debe leer para cada cliente el DNI y el número de actividad elegida (1 ..4). La lectura finaliza cuando llega el DNI 0.

Se pide, informar para cada cliente, el monto mensual a pagar y la cantidad de clientes que realizan la actividad 3.

Se sabe que cada cliente elige una sola actividad.