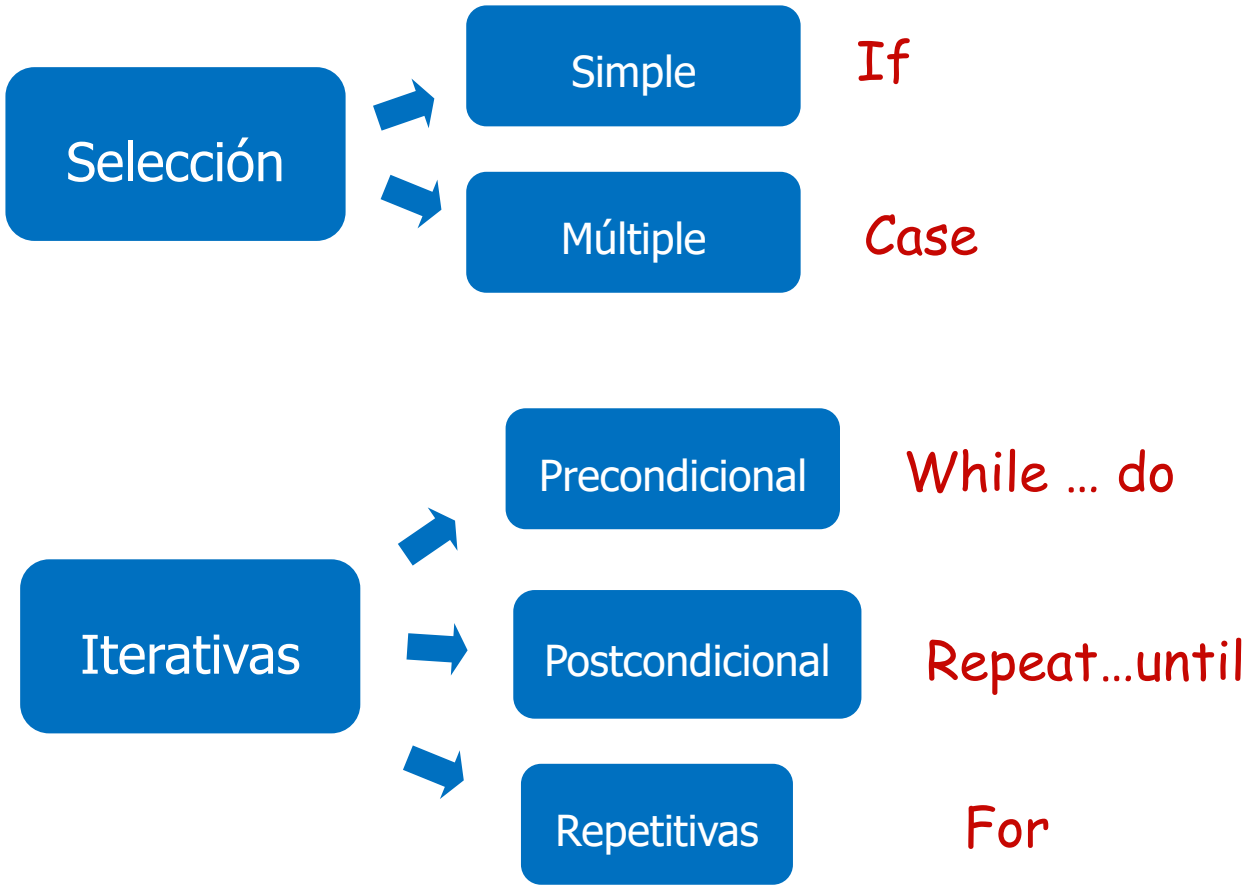
TEORIA 3

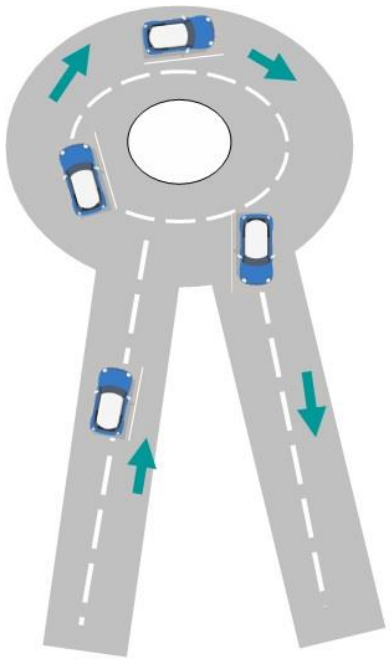
TEMAS de la CLASE

1. Estructuras de control
   - Iterativa Postcondicional
   - Repetitiva

Clasificación de las Estructuras de control.

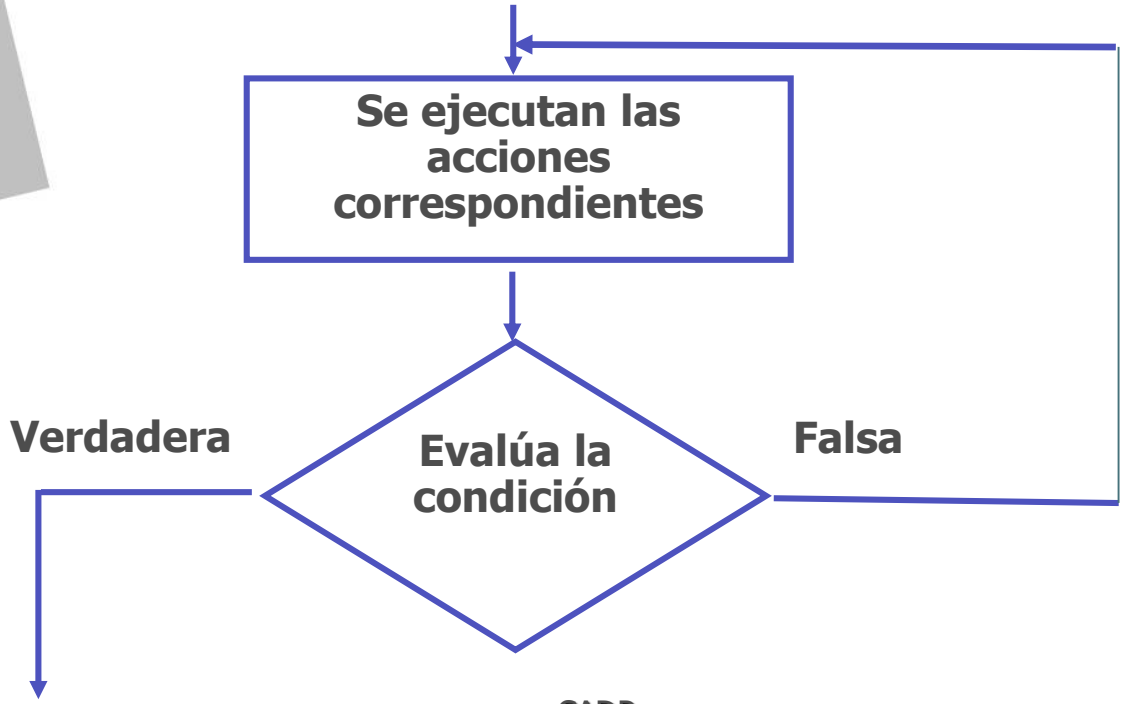


Estructura de Control iterativa postcondicional



- 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 (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.

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

Ejercicio

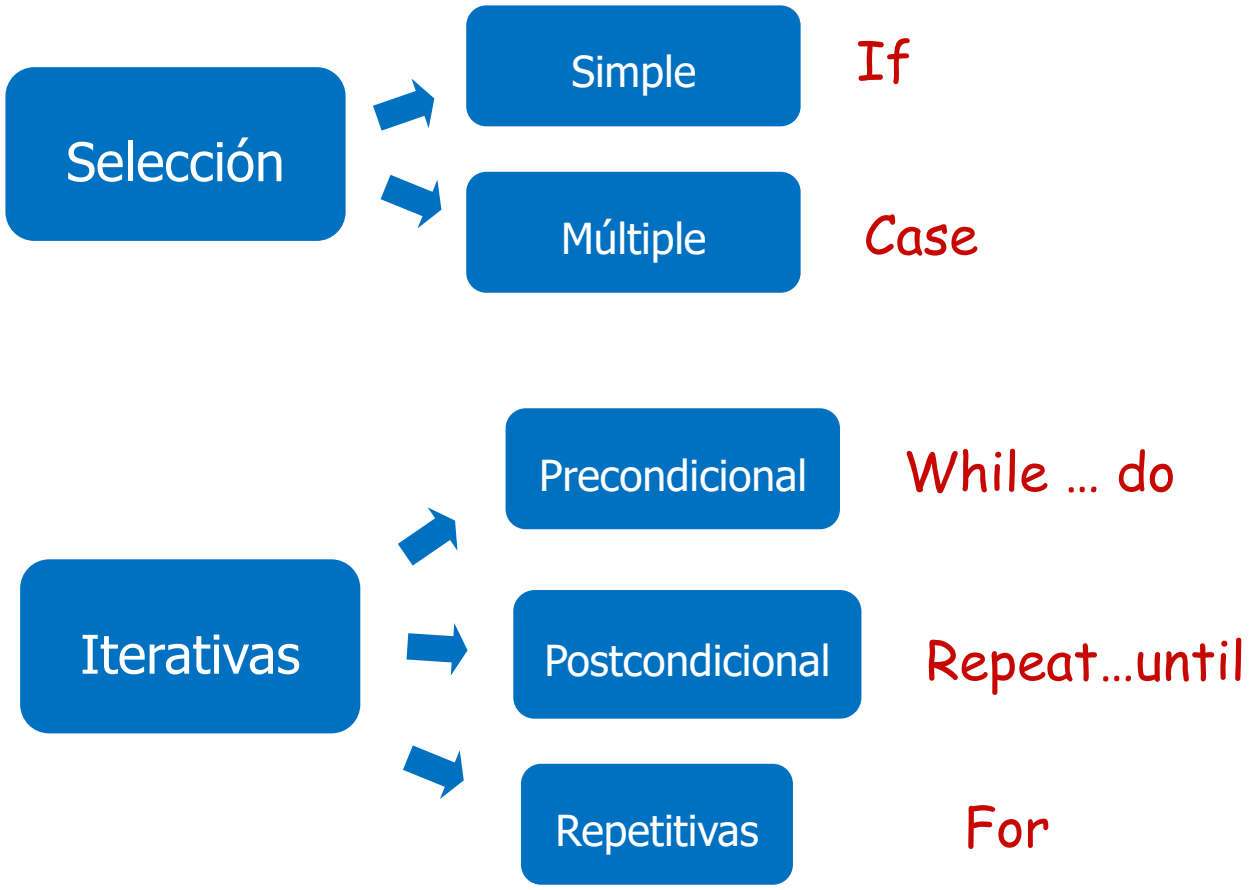
Escribir un programa que informe las superficies de terrenos cuyas dimensiones se leen de teclado. El ingreso de datos finaliza cuando se ingresa un terreno cuya superficie supera los 1000 mt2.

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

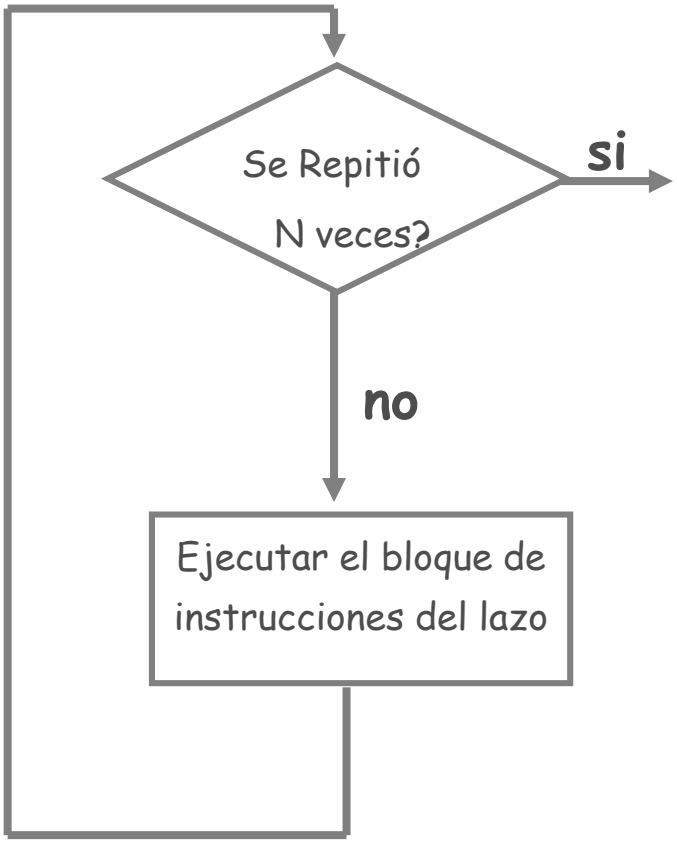
```
Program superficie;
  var frente, fondo, sup: real;

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

Clasificación de las Estructuras de control.



Estructura de control 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 repetitiva (en Pascal)

```
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 repetitiva (Ejemplos)

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

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

```
For índice := 20 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?

Ejercicio

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 suma15;
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 (suma);
end.
```

Ejercicio

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 sumaypares;
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.
```

Ejercicio

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

***

Ejercicio

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...

Ejercicio

Escribir un programa que lea
informar la cantidad de carac
caracteres leídos y mostrar u

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

```
Program ejemplo;
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.
```

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)

Precondicional (while)

Selección múltiple (case)

iteración

Repetición (For)

Postcondicional (repeat ... until)