Apunte: Estructura de Control Repetitiva

Tecnicaturas

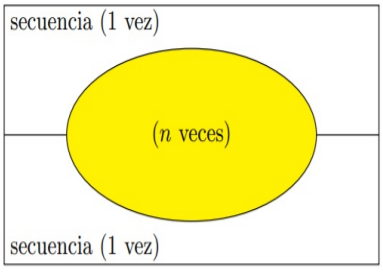
26 septiembre, 2023

1. Introducción

Definición: CONJUNTO DE DATOS DE ESTRUCTURA REPETITIVA

> Una **estructura repetitiva** es aquella en la que un conjunto de acciones se ejecutan un número determinado de veces y dependen de un valor predefinido o el cumplimiento de una determinada condición. En este último caso, para que la sucesión de acciones termine, dentro del conjunto de acciones a repetir se debe modificar por lo menos una de las variables que intervienen en la condición. Cada vez que se ejecuta el conjunto de acciones a repetir, diremos que se ejecuta una iteración del bucle.

Eso sí, la iteración no puede ser infinita. Las estructuras repetitivas son otra forma de estructura de control. El tratamiento de los datos por una estructura repetitiva se muestra en el siguiente gráfico. Note que un conjunto de datos es manipulado en forma reiterada un cierto número de veces:



2. Estructuras Repetitivas

Toda estructura repetitiva consta de un cláusula de control de la misma, y de un **cuerpo de la estructura repetitiva**, también denominada **bucle** de primitivas a repetir, ó **bloque** de primitivas a repetir. El bloque de instrucciones es una unidad que se ejecutará en forma completa si la cláusula de control lo permite. En pseudocódigo lo mostramos con su indentación correspondiente, y en PHP entre llaves.

2.1 Estructura Repetitiva PARA

Una estructura Repetitiva PARA se utiliza cuando la cantidad de veces a repetir es finita, y estamos en presencia de una cantidad definida de repeticiones (ej 3, 100, *n*-veces, *cantidad de alumnos*-veces).

Una estructura Repetitiva PARA cuenta con una variable iteradora, que es inicializada a un valor inicial (o **limite inferior**), y esta variable tomará los valores desde ese limite hasta el valor del **limite superior** inclusive (descripto en la cláusula HASTA). La variable iteradora se incrementará la cantidad de veces que indique la **cláusula PASO**, una vez que finaliza el bloque de sentencias a repetir. Si al ser incrementada la variable iteradora, este valor supera el valor del limite superior, la estructura repetitiva PARA finaliza.

Si la cláusula PASO, es PASO 1, el caso más habitual, esta puede omitirse . En otras palabras, si encontramos una estructura repetitiva PARA, sin cláusula PASO, esto implica que la variable iteradora se incrementará en una unidad cada vez que finalice un bucle de la estructura repetitiva.

Observación: El limite superior debe ser superior o igual al limite inferior, en valor absoluto, de otro modo la estructura estaría mal definida. Si el limite superior es menor en valor absoluto que el limite inferior se supone que el valor de la primitiva paso es negativo, con el fin de decrementar la variable iteradora.

```
ALGORITMO....
    secuencia
    PARA varIter← limInferior HASTA limSuperior PASO varIncr HACER
        Cuerpo de la
        Estructura
        Repetitiva
    FIN PARA
    secuencia
FIN ALGORITMO....
```

2.2 Estructura Repetitiva MIENTRAS

Una estructura Repetitiva MIENTRAS primero evalúa una expresión booleana, si la expresión booleana es verdadera ejecuta el bloque de sentencias a repetir. Una vez ejecutado el bloque o bucle, se vuelve a evaluar la expresión booleana y se opera en forma similar. Si la expresión booleana se evalúa a falso, se finaliza la ejecución de la estructura repetitiva.

Debido a que en la primera evaluación de la expresión booleana, puede retornar como resultado falso, esta estructura admite el caso de que se itere cero veces. A diferencia de la estructura repetitiva REPETIR MIENTRAS que al menos ejecuta una vez el bucle.

Por ello decimos que la estructura repetitiva permite 0 o más iteraciones. Cero, en el caso que la primera evaluación de la condición booleana sea falsa.

MIENTRAS

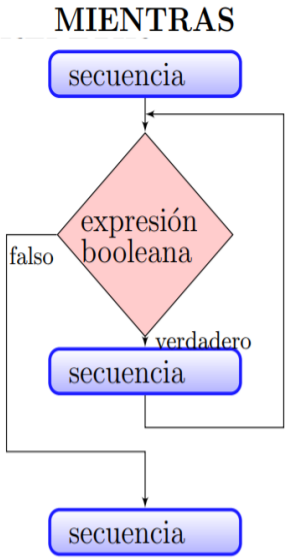
```
ALGORITMO....
    secuencia
    MIENTRAS (expresión booleana)
        cuerpo de la
        estr. rep. Se ejecuta
        si exp.bool es true
    FIN MIENTRAS
    secuencia
FIN ALGORITMO....
```

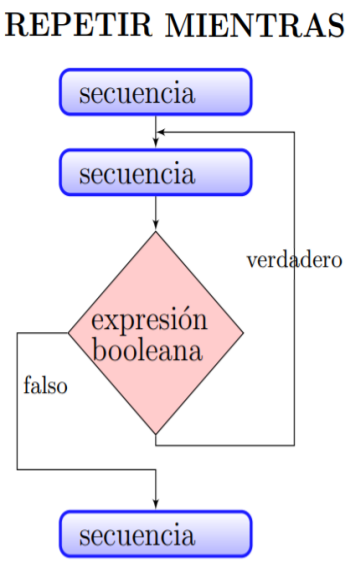
REPETIR MIENTRAS

```
ALGORITMO....
    secuencia
    REPETIR
        Cuerpo de la
        estr. rep. Se ejecuta
        si exp. booleana es  true
    MIENTRAS (expresion booleana)
    secuencia
FIN ALGORITMO....
```

2.3 Estructura REPETIR MIENTRAS

Una estructura Repetitiva REPETIR MIENTRAS primero ejecuta el bucle de sentencias a repetir y luego evalúa una expresión booleana. En pseudocódigo decimos que la estructura repetitiva REPETIR MIENTRAS, vuelve a ejecutar el conjunto de sentencias del bucle si la expresion booleana es VERDADERA. Y terminará de repetir cuando esta arroje resultado FALSO. En PHP, es idéntico, la instrucción de PHP repite mientras la expresion booleana sea verdadera.





2.4 ¿Cuando utilizar cada estructura repetitiva?

Existen variantes de una estructura repetitiva, pero todas ellas permiten que un conjunto de acciones se ejecuten entre 0 y n veces. La estructura **MIENTRAS** se puede ejecutar entre **0 y n veces**, la estructura **REPETIR MIENTRAS** se puede ejecutar entre **1 y n veces**. Utilizamos la estructura Repetir PARA cuando la cantidad de veces a repetir está determinada por un valor (es una cantidad definida de veces), esto representa un ciclo definido. Si la cantidad de veces a repetir es indefinida podemos utilizar un MIENTRAS o REPETIR MIENTRAS.

2.5 Estructuras Repetitivas en PHP

A continuación mostramos los constructores de estructuras repetitivas en PHP.

En el caso de la Estructura repetitiva PARA se implementa en java utilizando la palabra reservada **for** y a continuación, entre paréntesis, se especifican tres componentes, separados por punto y coma:

1. El valor inicial de la variable iteradora.
2. La condición que debe cumplir la variable iteradora. Condición que se evalúa al finalizar cada iteración del bucle.
3. El incremento de la variable iteradora

El bucle o cuerpo de sentencias a repetir se encierra entre llaves. Por ser este cuerpo o bucle un bloque de sentencias, en PHP es importante dejar sangría, espaciado o indententación para resaltar que ese conjunto es el cuerpo de la estructura repetitiva.

Indentar: significa hacer espacios hacia la derecha para mover una línea de código, lo podés hacer usando la barra espaciadora o con la tecla de tabulación (es la tecla que está justo arriba de tu tecla Bloquear Mayúsculas). Indentación es un anglicismo (de la palabra inglesa **indentation**) de uso común en informática; no es un término reconocido por la Real Academia Española .

MIENTRAS

```
while (condición)
{
   //acciones del bucle
}
```

REPETIR MIENTRAS

```
do
  {
  //acciones del bucle
  } while (condición);
```

REPETIR PARA

```
for (inicio;condicion;incremento)
  {
  //acciones del bucle
  }
```

3. Problemas con REPETIR PARA

Se muestran en esta sección algunos ejemplos que requieren de la estructura repetitiva PARA.

3.1 Tabla de Multiplicar

Problema: Solicitar al usuario un número entre 1 y 9 y mostrar la tabla de multiplicar de dicho número:

```
ALGORITMO tablaDeMultiplicar() RETORNA ∅
  (* dado un número entero entre 1 y 9
     muestra la tabla de multiplicar
     de ese nro *)
  ENTERO numero, j
  ESCRIBIR("Ingrese un número entre 1 y 9: ")
  LEER(numero)
  ESCRIBIR("Tabla de Multiplicar de:"+ numero)
  SI (numero>0 AND numero<=9) ENTONCES
     PARA j<-1 HASTA 10 HACER
       ESCRIBIR(j+"*"+numero +"=" + numero*j )
     FIN PARA
  SINO
     ESCRIBIR("Número no válido")
  FIN SI
FIN ALGORITMO tablaDeMultiplicar
```

```
<?php
  // principal
  // INT $numero
  // INT $j
  // Entrada de datos
  echo "Ingrese un numero entre 1 y 9";
  $numero = trim(fgets(STDIN));
  echo "Tabla de Multiplicar de:".$numero."\n";
  if ($numero>=0 & $numero<=9){
    for ($j=1; $j<=10; $j++){
      echo $j."*".$numero."=".$numero*$j."\n";
    }
  }
?>
```

Descargar Archivo EjTablaMultiplicar.php (https://opendata.fi.uncoma.edu.ar/rpa/EjTablaMultiplicar.php)

La traza del algoritmo suponiendo que el usuario ingresa el valor 3:

tablaDeMultiplicar:

numero	j	salida
3	~~1~~	Ingrese un número entero entre 1 y 9:
.	~~2~~	Tabla de multiplicar de: 3
.	~~3~~	1 * 3 = 3
.	~~4~~	2 * 3 = 6
.	~~5~~	3 * 3 = 9
.	~~6~~	4 * 3 = 12
.	~~7~~	5 * 3 = 15
.	~~8~~	6 * 3 = 18
.	~~9~~	7 * 3 = 21
.	~~10~~	8 * 3 = 24
.	11	9 * 3 = 27
.		10 * 3 = 30

3.2 Edad Promedio de un número determinado de alumnos

Problema: Solicitar al usuario el número determinado de alumnos y calcular la edad promedio de alumnos

```
ALGORITMO edadPromedio() RETORNA ∅
  (* calcula la edad promedio de un
     numero determinado de alumnos*)
  ENTERO cantAlumnos, itAl, edadAlumno
  REAL acumEdades
  acumEdades <- 0.0
  ESCRIBIR("Ingrese la cantidad de Alumnos: ")
  LEER(cantAlumnos)
  PARA itAl <- 1 HASTA cantAlumnos PASO 1 HACER
    ESCRIBIR("Ingrese la edad del alumno"+itAl)
    LEER(edadAlumno)
    acumEdades <- acumEdades + edadAlumno
  FIN PARA
  ESCRIBIR("La edad promedio es:"+acumEdades/cant
Alumnos)
FIN ALGORITMO edadPromedio
```

```
<?php
   /* principal
  calcula la edad Promedio de un
  numero determinado de alumnos
  INT $cantAlumnos, $itAl, $edadAlumno
  REAL $acumEdades */
  $acumEdades = 0.0;
  echo "Ingrese la cantidad de Alumnos: ";
  $cantAlumnos = trim(fgets(STDIN));
  for ($itAl= 1; $itAl<=$cantAlumnos; $itAl++){
    echo "Ingrese la edad del alumno".$itAl;
    $edadAlumno = trim(fgets(STDIN));
    $acumEdades = $acumEdades + $edadAlumno;
  }
  echo "La edad promedio es:".($acumEdades/$cantA
lumnos);
?>
```

Descargar Archivo

- EdadPromedio.php (https://opendata.fi.uncoma.edu.ar/rpa/EdadPromedio.php)
- Traza del pseudocódigo (https://opendata.fi.uncoma.edu.ar/algoritmos/files/Traza.docx)

edadPromedio:

cantAlumnos	itAl	edadAlumno	acumEdades	salida
3	~~1~~	~~2~~	~~0,0~~	Ingrese la cantidad de alumnos:
.	~~2~~	~~4~~	~~2,0~~	Ingrese la edad del alumno 1:
.	~~3~~	6	~~6,0~~	Ingrese la edad del alumno 2:
.	4		12,0	Ingrese la edad del alumno 3:
.				La edad promedio es: 4

3.3 Obtener el valor mínimo de un número determinado de números

Problema: Solicitar al usuario una cantidad determinada de alumnos y obtener la menor edad del grupo

```
ALGORITMO menorEdad() RETORNA ⌀
  (* este alg. obtiene la edad del
     menor alumno de un conj. det. de alumnos*)
  ENTERO cantAlumnos, itAl, edadAlumno, menorEdad
  menorEdad <- 120
  ESCRIBIR("Ingrese la cantidad de Alumnos: ")
  LEER(cantAlumnos)
  PARA itAl <- 1 HASTA cantAlumnos PASO 1 HACER
    ESCRIBIR("Ingrese la edad del alumno"+itAl)
    LEER(edadAlumno)
    SI (edadAlumno < menorEdad) ENTONCES
        menorEdad <- edadAlumno
    FIN SI
  FIN PARA
  ESCRIBIR("La menor edad es:"+menorEdad)
FIN ALGORITMO menorEdad
```

```
<?php
   /* principal
  este alg. obtiene la edad del
  menor alumno de un conj. det.
  de alumnos
  INT $cantAlumnos, $itAl, $edadAlumno, $menorEda
d */
  $menorEdad = 120;
  echo "Ingrese la cantidad de Alumnos: ";
  $cantAlumnos = trim(fgets(STDIN));
  for ($itAl = 1; $itAl <= $cantAlumnos; $itAl++)
{
    echo "Ingrese la edad del alumno".$itAl.": ";
    $edadAlumno = trim(fgets(STDIN));
    if ($edadAlumno < $menorEdad) {
       $menorEdad= $edadAlumno;
    }
   }
   echo "La menor edad es:".$menorEdad;
?>
```

Descargar Archivo MenorEdad.php (https://opendata.fi.uncoma.edu.ar/ip/MenorEdad.php)

edadPromedio:

cantAlumnos	itAl	edadAlumno	menorEdad	salida
3	~~1~~	~~4~~	~~100~~	Ingrese la cantidad de alumnos:
.	~~2~~	~~2~~	~~4~~	Ingrese la edad del alumno 1:
.	~~3~~	6	2	Ingrese la edad del alumno 2:
.	4			Ingrese la edad del alumno 3:
.				La menor edad es: 2

4. Problemas con MIENTRAS

4.1 Edad Promedio de un número indeterminado de alumnos (Texto SI-NO)

Problema: Supongamos que el usuario indicará la edad de un alumno y luego le consultaremos *¿desea continuar? (SI/NO)* y el deberá contestar SI o NO.

```
ALGORITMO EdadPromedioCI() RETORNA ∅
(* calcula la edad promedio de un
  numero indeterminado de Alumnos *)
 ENTERO cantAlumnos, j, edadAlumno
 REAL acumEdades
 TEXTO continuar
 acumEdades <- 0.0
 ESCRIBIR("Desea ingresar un alumno? SI/NO")
 LEER(continuar)
 j <- 1
 MIENTRAS ( continuar="SI") HACER
   ESCRIBIR("Ingrese la edad del alumno" + j)
   LEER(edadAlumno)
   acumEdades <- acumEdades + edadAlumno
   ESCRIBIR("Desea continuar? SI/NO")
   LEER(continuar)
   SI  ( continuar="SI") ENTONCES
         j<-j+1
   FIN SI
 FIN MIENTRAS
 ESCRIBIR("La edad promedio es:" + acumEdades /j)

FIN ALGORITMO EdadPromedioCI
```

```
<?php
   /* principal
  calcula la edad promedio de un
  numero indeterminado de Alumnos
  INT $cantAlumnos, $j, $edadAlumno
  FLOAT $acumEdades
  STRING continuar */
  $acumEdades = 0;
  echo "Desea ingresar un alumno? SI/NO \n";
  $continuar = trim(fgets(STDIN));
  $j = 1;
  while ( $continuar=="SI") {
    echo "Ingrese la edad del alumno".$j;
    $edadAlumno = trim(fgets(STDIN));
    $acumEdades = $acumEdades + $edadAlumno;
    echo "Desea continuar? SI/NO \n";
    $continuar = trim(fgets(STDIN));
    if  ($continuar =="SI") {
        $j = $j + 1;
    }
    }
    echo "La edad promedio es:".($acumEdades /
$j);
?>
```

Descargar Archivo EdadPromedioCI.php (https://opendata.fi.uncoma.edu.ar/rpa/EdadPromedioCI.php)

- Ejercicio: Modifique el pseudocódigo y PHP de este último ejercicio para tener en cuenta que el usuario puede contestar alguna de estas alternativas: SI, Si, si, sI, NO, No, no, nO.
- ¿Qué sucede si no contesta ninguna de ellas?

4.3 Las joyas de la Abuela

Problema: La Sra. Medina tiene una herencia en joyas de gran valor. Pero lo que a esta señora le interesa es comprarse un departamento, por lo cual decide vender algunas de sus joyas. El joyero le paga por cada gramo de oro dependiendo de su calidad que puede ser: Calidad 1: $80 el gramo. Calidad 2: $120 el gramo. Calidad 3: $300 el gramo. La Sra. Medina elije al azar las joyas cuyos valores sumen al menos $200000 que es lo que necesita para comprarse el departamento. Realizar un algoritmo que lea los datos de cada una de las joyas hasta alcanzar una suma mayor o igual al valor requerido. El algoritmo debe mostrar el monto total que el joyero pagará a la Sra. Medina por sus joyas.

```
MODULO obtenerValor(ENTERO calidad,REAL gramos) R
ETORNA REAL
  (* obtiene el valor de joyas a partir
      de su peso y calidad de oro *)
  REAL valor
  SI (calidad = 1) ENTONCES
      valor <- 80 * gramos
  OTRO-SI (calidad = 2) ENTONCES
      valor <- 120 * gramos
  OTRO-SI (calidad = 3) ENTONCES
      valor <- 300 * gramos
  SINO
      valor <- 0
  FIN SI
  RETORNAR valor
FIN MODULO obtenerValor

ALGORITMO joyas() RETORNA ⌀
  (* este alg. determina el valor que
     pagará el joyero a una abuela *)
  REAL acumJoyas, valorJoya, gramosJoya
  ENTERO calidadJoya
  acumJoyas <- 0.0

  MIENTRAS (acumJoyas < 200000) HACER
    ESCRIBIR("Ingrese los datos de la joya ")
    ESCRIBIR("¿Cuál es la calidad de la joya? 1,
2 ó 3")
    LEER(calidadJoya)
    ESCRIBIR("¿Cuántos gramos pesa la joya?")
    LEER(gramosJoya)
    valorJoya <- obtenerValor(calidadJoya, gramos
Joya)
    acumJoyas <- acumJoyas + valorJoya
  FIN MIENTRAS
  ESCRIBIR("El joyero pagará :"+acumJoyas+"$")
FIN ALGORITMO joyas
```

```php
<?php
  /**
   Obtiene el valor de joyas a partir
   de su peso y calidad de oro
   @param INT $calidad
   @param FLOAT $calidad
   @return FLOAT
  */
function obtenerValor($calidad, $gramos){
    // REAL $valor
    if ($calidad == 1) {
        $valor = 80 * $gramos;
    }
    elseif ($calidad == 2) {
        $valor = 120 * $gramos;
    }
    elseif ($calidad == 3) {
        $valor = 300 * $gramos;
    }
    else
    $valor = 0;
    return $valor;
}

/* este alg. determina el valor que
pagará el joyero a una abuela
REAL acumJoyas, valorJoya, gramosJoya
ENTERO calidadJoya */
$acumJoyas = 0.0;
while ($acumJoyas < 200000) {
  echo "Ingrese los datos de la joya \n";
  echo "¿Cuál es la calidad de la joya? 1, 2 ó
3";
  $calidadJoya = trim(fgets(STDIN));
  echo "¿Cuántos gramos pesa la joya?";
  $gramosJoya  = trim(fgets(STDIN));
  $valorJoya = obtenerValor($calidadJoya, $gramos
Joya);
  $acumJoyas = $acumJoyas + $valorJoya;
}
  echo"El joyero pagará :". $acumJoyas . "$";

  ?>
```

Descargar Archivo Joyas.php (https://opendata.fi.uncoma.edu.ar/rpa/Joyas.php)

4.4 Determinar si un número es primo

```
ALGORITMO primo() RETORNA ∅
  (* Determina si un numero es par *)
  ENTERO numero
  ESCRIBIR("Ingrese un número entero: ")
  LEER(numero)
  SI (numero > 1) ENTONCES
    SI (esPrimo(numero)) ENTONCES
      ESCRIBIR("El número es primo")
    SINO
      ESCRIBIR("El número no es primo")
    FIN SI
  FIN SI
FIN ALGORITMO primo

MÓDULO esPrimo(ENTERO numero) RETORNA LÓGICO
(* determina si un numero es primo
  numero: el numero a analizar
  retorna true si es primo, false en caso contrar
io*)
  ENTERO num
  LOGICO resultado
  num <- numero - 1
  resultado <- VERDADERO
  MIENTRAS (num != 1 AND num != 0 AND
            !esDivisible(numero, num)) HACER
      num <- num - 1
  FIN MIENTRAS
  SI (num != 1) ENTONCES
      resultado <- FALSO
  FIN SI
  RETORNAR resultado
FIN MÓDULO esPrimo

MÓDULO esDivisible(ENTERO num1, num2) RETORNA LÓG
ICO
  (* determina si un numero es divisible por otro
  num1 y num2: los números a analizar
  retorna VERDADERO si son divisibles
  y FALSO en caso contrario*)
  RETORNAR ((num1 MOD num2)==0)
FIN MÓDULO esPrimo
```

```
<?php
   /** determina si un numero es divisible por ot
ro
   num1 y num2: los números a analizar
   retorna VERDADERO si son divisibles
   y FALSO en caso contrario
   @param INT $num1
   @param INT $num2
   @return BOOLEAN
   */
  function esDivisible($num1, $num2) {
     return (($num1 % $num2) == 0);
  }

  /** determina si un numero es primo
  numero: el numero a analizar
  retorna true si es primo,
  false en caso contrario
  @param INT $numero
  @return BOOLEAN */
  function esPrimo($numero){
  // INT $num
  // BOOLEAN $resultado
  $num = $numero - 1;
  $resultado = TRUE;
  while (($num != 1 & $num != 0 ) & !(esDivisible
($numero, $num))) {
    $num = $num - 1;
  }
  if ($num != 1) {
    $resultado = FALSE;
  }
  return $resultado;
  }

  // principal
 /* Determina si un numero es par
  INT $numero1 */
  echo "Ingrese un número entero: ";
  $numero = trim(fgets(STDIN));
  if ($numero > 1) {
     if (esPrimo($numero)) {
        echo "El número es primo";
     } else {
        echo "El número no es primo";
     }
  }

  ?>
```

Descargar Archivo primo.php (https://opendata.fi.uncoma.edu.ar/rpa/primo.php)

5. Problemas REPETIR MIENTRAS

5.1 Edad Promedio de un número indeterminado de alumnos (1 o más)

Problema:

```
ALGORITMO edadPromedioCI() RETORNA ∅
(* calcula la edad promedio de un
  numero indeterminado de 1 ó mas alumnos *)
 ENTERO cantAlumnos, j, edadAlumno
 REAL acumEdades
 TEXTO continuar
 acumEdades <- 0.0
 j <- 1
 REPETIR
   ESCRIBIR("Ingrese la edad del alumno" + j)
   LEER(edadAlumno)
   acumEdades <- acumEdades + edadAlumno
   ESCRIBIR("Desea ingresar un alumno? SI/N0")
   LEER(continuar)
   SI ( continuar ="SI" ) ENTONCES
      j<- j + 1
   FIN SI
 MIENTRAS ( continuar = "NO" )
 ESCRIBIR("La edad promedio es:" + acumEdades /
j);
FIN ALGORITMO edadPromedioCI
```

```
<?php
 // principal
 /* calcula la edad promedio de un
  numero indeterminado de 1 ó mas alumnos
  INT $cantAlumnos, $j, $edadAlumno
  FLOAT $acumEdades
  STRING $continuar */
  $acumEdades = 0.0;
  $j = 1;
  do {
     echo "Ingrese la edad del alumno ".$j."\n";
     $edadAlumno = trim(fgets(STDIN));
     $acumEdades = $acumEdades + $edadAlumno;
     echo "¿Desea ingresar otro alumno? SI/NO
\n";
     $continuar = trim(fgets(STDIN));
     if ($continuar === "SI") {
        $j = $j +1;
     }
  } while ($continuar === "SI");
  echo "La edad promedio es: ". ($acumEdades/$j);

?>
```

Descargar Archivo EjRM_EdadPromedio.php (https://opendata.fi.uncoma.edu.ar/rpa/EjRM_EdadPromedio.php)

5.2 Obtener la cantidad de números pares ingresados por el usuarios hasta que el usuario ingrese un cero.

Problema: Solicitar al usuario números enteros, hasta que el mismo ingrese el numero cero. Determine cuántos de los números ingresados son números pares.

```
ALGORITMO determinarPares() RETORNA ∅
(* solicita al usuario numeros enteros hasta que
  que se ingrese un cero, finalmente
  muestra la cantidad de numeros pares.
  El usuario ingresará al menos un
  numero distinto de cero*)
  ENTERO numero, cantPares
  LÓGICO resultado
  cantPares <- 0
  ESCRIBIR("Ingrese un número entero \n"
     + "Para finalizar ingrese un cero")
  LEER(numero)
  REPETIR
    SI (esPar(numero)) ENTONCES
        cantPares <- cantPares + 1
    FIN SI
    ESCRIBIR("Ingrese otro número entero. "
        + "Para finalizar ingrese un cero");
    LEER(numero)
  HASTA (numero = 0)
  (* incrementamos una vez mas porque
  el ultimo número no verificado es cero,
  que sabemos que es par *)
  cantPares <- cantPares +1
  ESCRIBIR("La cantidad de numero pares es: "
       + cantPares)
FIN ALGORITMO determinarPares

MODULO esPar(ENTERO num) RETORNA LOGICO
  (* determina si un numero es par o no
     num: entero recibido como parametro
     retorna verdadero si num es par,
            falso en caso contrario*)
  RETORNAR ((num MOD 2) == 0)
FIN MODULO esPar
```

```
<?php

/**
 * determina si un numero es par o no
 *    num: entero recibido como parametro
 *    retorna verdadero si num es par
 * @param INT $num
 * @return BOOLEAN
 */
function esPar($num) {
    return (($num % 2)==0);
}

/* principal
* solicita al usuario numeros enteros hasta que
* que se ingrese un cero, finalmente
*  muestra la cantidad de numeros pares.
*  El usuario ingresará al menos un
*  numero distinto de cero
* INT $numero, $cantPares
* BOOLEAN $resultado */
 $cantPares = 0;
 echo "Ingrese un número entero \n";
 echo "Para finalizar ingrese un cero";
 $numero = trim(fgets(STDIN));
 do {
    if (esPar($numero)){
        $cantPares = $cantPares+1;
    }
    echo "Ingrese otro número entero";
    echo "Para finalizar ingrese un cero";
    $numero = trim(fgets(STDIN));
 } while ($numero != 0);
  /** incrementamos una vez mas porque
  * el ultimo número no verificado es cero,
  *que sabemos que es par */
  $cantPares = $cantPares +1;
  echo "La cantidad de numero pares es: ". $cantP
ares;

?>
```

Descargar Archivo EjRM_CantPares.php (https://opendata.fi.uncoma.edu.ar/rpa/EjRM_CantPares.php)

6. Pirámides

Problema: Supongamos que deseamos mostrar por pantalla una pirámide como la siguiente:
1
1 2
1 2 3
1 2 3 4

donde la cantidad de filas la determina el usuario. Construya una solución en el cual la pirámide se construye en un módulo.

```
ALGORITMO Ejpiramide() RETORNA ∅
(* solicita al usuario la cant de filas
   de una pirámide y luego muestra la misma*)
   ENTERO niveles
   ESCRIBIR("Ingrese la cantidad de filas")
   LEER(niveles)
   piramide(niveles);
FIN ALGORITMO Ejpiramide

MODULO piramide(ENTERO n) RETORNA ∅
  (* construye una piramide
     num: cantidad de filas de la piramide*)
     ENTERO fila, col;
     PARA fila = 1 HASTA n PASO 1 HACER
        PARA col = 1 HASTA fila PASO 1 HACER
           ESCRIBIR(col + " ") (* sin salto *)
        FIN PARA
        ESCRIBIR(" ") (* con salto de linea *)
     FIN PARA
FIN MODULO piramide
```

```php
<?php

/**
 * construye una piramide
 * de acuerdo a la cantidad de filas
 * @param INT $niveles
 */
function piramide($n) {
   // INT fila, col
   for ($fila = 1 ; $fila <= $n; $fila++) {
       for ($col = 1 ; $col <= $fila; $col++) {
         echo $col." " ;
       }
       echo "\n";
   }
}

/* principal
* solicita al usuario la cant de filas
* de una pirámide y luego muestra la misma
* INT $niveles */
echo "Ingrese la cantidad de filas";
$niveles = trim(fgets(STDIN));
piramide($niveles);

?>
```

Descargar Archivo Piramide.php (https://opendata.fi.uncoma.edu.ar/rpa/Piramide.php)

Ejercicio: Introduzca cambios al código anterior de tal forma que el algoritmo muestra una pirámide solo si la cantidad de filas es inferior a 10, y en caso contrario muestre un mensaje de error.

Ejercicio: Imagina más pirámides y para cada una de ellas, detalla el pseudocódigo y luego implementa en PHP.