



Resolución de Problemas y Algoritmos


Clase 7
Repetición condicional



Fibonacci



Dr. Alejandro J. García
http://cs.uns.edu.ar/~ajg



Departamento de Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur
Bahía Blanca - Argentina

Motivación

Existen algoritmos donde:

- se debe repetir una secuencia de acciones, pero
- no se sabe de antemano cuantas veces se van a repetir.

Por ejemplo: ➔

Algoritmo: cursada

Repetir intentar cursar materia **hasta** materia cursada

Algoritmo: envasar productos

Repetir mientras hay productos

- tomar producto
- esperar por envase vacío
- poner producto en envase
- cerrar envase

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 2

Conceptos: sentencias repetitivas en Pascal

Repetición incondicional:

```
FOR var := <ini> TO <fin> DO <sentencia>
FOR var := <ini> DOWNTO <fin> DO <sentencia>
```

Repetición condicional (depende de una condición):

WHILE <condición>
DO <sentencia>

REPEAT
<sentencias>
UNTIL <condición>

Importante: consultar los diagramas sintácticos de Pascal ([aquí](#)) para conocer los detalles de la sintaxis de todas estas sentencias.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 3

Repetición basada en condiciones

Por ejemplo, considere que se quiere validar el ingreso de datos y se quiere repetir el ingreso de datos **mientras** estos no sean correctos.

```
mostrar('Ingrese una letra mayúscula')
leer(letra)
{ validación de los datos ingresados por el usuario }
MIENTRAS ( letra < 'A' o (letra > 'Z') {i.e., no sea mayúscula}
  mostrar(' Error, ingrese una letra mayúscula')
  leer(letra)
fin repetir
{ ... Datos validados: si llega a este punto
  es porque letra tiene una mayúscula ... }
```

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 4

Repetición condicional en Pascal: WHILE

La **sentencia** de un ciclo **WHILE** se ejecutará **0 (cero) o más veces** dependiendo del resultado (true o false) que se obtiene al evaluar la expresión booleana que representa la condición.

```
sentencia WHILE → while → expresión → do → proposición →
```

Primero se evalúa la expresión booleana

```
WHILE expresión booleana
DO sentencia ;
Otra sentencia siguiente;
```

Si el resultado es FALSE, saltea (no ejecuta) la sentencia del DO y sigue en la siguiente al while.

Si el resultado es TRUE se ejecuta la sentencia que sigue al DO

Una vez ejecutada la sentencia que sigue al DO, se vuelve a evaluar la expresión booleana

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 5

"While loop": repetición condicional

Las sentencias dentro de un **WHILE** se ejecutan **0 (cero) o más veces**.

Casos de prueba: tope con -1, 0, 2

Obs: si la sentencia del while se repitió N veces, la expresión **cont < tope** se evaluó N+1 veces.

Mientras cont < tope sea true ejecuta:

Si cont < tope es false sigue en:

```
programa ejemplo;
var tope, cont: integer;
begin
  Write('Ingre un tope: ');
  readln(tope);
  cont := 0;
  WHILE cont < tope
  DO Begin
    writeln(cont);
    cont:=cont+1;
  End;
  Writeln('press enter');
  readln;
end.
```

tope	cont	cont<tope
?	?	?

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 6

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:
 "Resolución de Problemas y Algoritmos. Notas de Clase". Alejandro J. García. Universidad Nacional del Sur. (c)2016

Ejemplo de aplicación de WHILE

Validar el ingreso de datos y repetir el ingreso de datos **mientras** estos no sean correctos.

```
PROGRAM EjemploWhile;
{muestra una aplicación útil del WHILE}
VAR letra: char;
BEGIN
  writeln('Ingrese una letra mayúscula');
  readln(letra);
  { validación de los datos ingresados por el usuario}
  WHILE ( letra < 'A' OR (letra > 'Z') DO
    BEGIN writeln('Incorrecto. Ingrese letra mayúscula');
          read(letra);
    END;
  {... aquí estoy seguro que letra tiene una mayúscula y
  continúo con el resto del programa...}
```

Casos de prueba:
 (letra < 'A') OR (letra > 'Z')

letra	¿repite?
A	FALSE no
a	TRUE sí
N	FALSE no
n	TRUE sí
\$	TRUE sí
1	TRUE sí

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 7

Ejemplo de aplicación de WHILE

Validar el ingreso de datos y repetir el ingreso de datos **mientras** estos no sean correctos.

```
PROGRAM EjemploWhile;
{muestra una aplicación útil del WHILE}
VAR opcion: integer;
BEGIN
  writeln('Ingrese opción: (1) pasar a minúscula
          (2) mostrar código ASCII ');
  read(opcion);
  { validación de los datos ingresados por el usuario}
  WHILE ( opcion <> 1) and ( opcion <> 2) DO
    BEGIN write(' Incorrecto. Ingrese 1 o 2');
          read(opcion);
    END
  {... resto del programa...}
```

Casos de prueba:
 (opcion <> 1) and (opcion <> 2)

opcion	¿repite?
1	FALSE no
0	TRUE sí
2	FALSE no
5	TRUE sí

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 8

Repetición basada en condiciones

Por ejemplo, se quiere permitir al usuario **repetir** la ejecución del programa **hasta** que el decida.

REPETIR
 { esta parte del programa se repetirá hasta que el usuario indique fin}

mostrar "Pulse la letra (S) para ejecutar nuevamente o cualquier otra letra para salir de la aplicación"

leer (letra)

HASTA que se pulse letra distinta de 'S' ;
 mostrar " Muchas gracias por utilizar el programa"

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 9

Repetición condicional en Pascal

Las sentencias dentro de un **REPEAT-UNTIL** se ejecutan **1 o más veces** dependiendo del resultado (true o false) que se obtiene al evaluar la expresión booleana que representa la condición.

```
sentencia REPEAT → repeat → proposición → until → expresión
```

REPEAT
 <sentencia 1>
 <sentencia 2>
 ...
 <sentencia n>
UNTIL <expresión booleana>

Si el resultado es **FALSE** vuelve a ejecutar la secuencia a partir de la palabra reservada **REPEAT**

Si el resultado es **TRUE** no vuelve a repetir y sigue en la sentencia siguiente a **UNTIL**

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 10

"Repeat loop": repetición condicional

```
programa ejemplo2;
var tope, cont: integer;
begin
  Write('Ingrese un tope: ');
  readln(tope);
  cont := 0;
  REPEAT
    writeln(cont);
    cont:=cont+1;
  UNTIL cont >= tope;
  Writeln('press enter');
  readln;
end.
```

Las sentencias dentro de un **REPEAT** se ejecutan **1 (una) o más veces**.

Casos de prueba:
 tope con -1, 0, 2

tope	cont	Cont >= tope
?	?	?

Si el resultado es **false** vuelve a repetir desde

Si el resultado es **true** sigue en

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 11

Ejemplo con REPEAT-UNTIL

```
PROGRAM EjemploRepeat;
{muestra una aplicación útil del repeat}
VAR letra: char;
BEGIN
  REPEAT
    { esta parte del programa se repetirá
    hasta que el usuario lo indique}
    writeln('Pulse la letra (S) para ejecutar nuevamente o
            cualquier otra letra para salir de la aplicación');
    readln(letra);
    {se volverá a repetir si presiona la tecla S}
  UNTIL (letra<>'S') and (letra<>'s');
  writeln('Muchas gracias por utilizar el programa.');
```

Casos de prueba:
 (letra<>'S') and (letra<>'s')

letra	¿repite?
S	FALSE sí
s	FALSE sí
N	TRUE no
n	TRUE no
\$	TRUE no
1	TRUE no

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 12

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:
 "Resolución de Problemas y Algoritmos. Notas de Clase". Alejandro J. García. Universidad Nacional del Sur. (c)2016

Conceptos: Diferencias REPEAT y WHILE

<p>REPEAT-UNTIL</p> <ul style="list-style-type: none"> si condición es falsa sigue repitiendo. si condición es verdadera deja de repetir. repite 1 o más veces: siempre ejecuta al menos una vez la secuencia interna al repetir 	<p>WHILE</p> <ul style="list-style-type: none"> si condición es verdadera sigue repitiendo si condición es falsa deja de repetir repite 0 o más veces: puede no ejecutar nunca la secuencia interna al repetir
---	---

• **Ejercicio propuesto para practicar**: escriba las diferencias y similitudes entre las tres sentencias repetitivas FOR, WHILE y REPEAT.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 13

Repeticiones anidadas (algunos ejemplos)

REPEAT

WHILE <condición>

DO **WHILE** <condic.>

DO <sent. >

WHILE <condición>

DO FOR v:= ... TO ... **DO**

<sentencia >

REPEAT

REPEAT

<sentencia >

UNTIL <condición>

UNTIL <condición>

El límite está en la imaginación del programador 🤖👍

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 14

Conceptos: repetición condicional vs. incondicional

- La repetición **condicional** (**REPEAT** o **WHILE**) depende de una condición (expresión boolean).
- La repetición **incondicional** (**FOR**) se ejecuta un **número fijo** de veces que se conoce antes de comenzar a repetirse la sentencia.
- Toda vez** que se puede usar una repetición **incondicional** (**FOR**), el código podría **reescribirse** para usarse una repetición condicional (**while** o **repeat**).
- Pero **no todas** las repeticiones **condicionales** (**while** o **repeat**) **pueden reescribirse** para usar una incondicional (**FOR**).

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 15

Repetición condicional es **MÁS GENERAL**

a:=1;

FOR v:=1 TO 5

DO a:=a*v;

Write(a);

➔

a:=1; v:=1;

REPEAT

a:=a*v;

v:=v+1;

UNTIL v > 5

Write(a);

➔

a:=1; v:=1;

WHILE v <= 5

DO BEGIN

a:=a*v;

v:=v+1;

END

Write(a);

Write('ingrese nro. positivo: ');

Read(num);

WHILE num < 0 **DO** Read(num);

Write('ingrese nro. positivo: ');

REPEAT Read(num);

UNTIL num >= 0;

~~FOR ?? TO ?? DO~~

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 16

Concepto: CICLO INFINITO ☹️

Una repetición que se realiza infinitas veces se denomina **ciclo infinito**

- En RPA un ciclo infinito en un programa será considerado un **ERROR GRAVE** de programación.
- Cuando realice la traza de sus programas **debe asegurarse** que en las repeticiones **no exista ningún caso** en el cual el programa pueda **caer en un ciclo infinito**.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 17

Conceptos: CICLOS INFINITOS ☹️

Una repetición condicional mal programada puede caer en una **REPETICIÓN INFINITA**, lo cual es un error grave de programación

<p>{...OK...}</p> <p>v:=1; w:=1;</p> <p>REPEAT</p> <p>v:=v+1;</p> <p>UNTIL V = 3;</p> <p>Write(V);</p>	<p>{ 1. MAL }</p> <p>v:=1; w:=1;</p> <p>REPEAT</p> <p>v:=v+1;</p> <p>UNTIL w = 0;</p> <p>Write(V);</p> <p>MAL</p>	<p>{ 2. MAL }</p> <p>v:=1; w:=1;</p> <p>WHILE V<3</p> <p>DO v:=1;</p> <p>Write(V);</p> <p>MAL</p>	<p>{ 3. MAL }</p> <p>v:=1; w:=1;</p> <p>REPEAT</p> <p>v:= w-1;</p> <p>UNTIL V = 3;</p> <p>Write(V);</p> <p>MAL</p>
--	--	---	---

Algunos problemas clásicos:

- La condición de corte es errónea.
- La variable de la condición no se modifica.
- La variable de la condición se modifica mal.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 18

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:
 “Resolución de Problemas y Algoritmos. Notas de Clase”. Alejandro J. García. Universidad Nacional del Sur. (c)2016

Problema Propuesto

Escriba un programa que calcule el promedio de números reales ingresados por el usuario.

Ingrese un valor: 8.2
¿otro? s/n s
Ingrese un valor: 0.2
¿otro? s/n s
Ingrese un valor: -3.0
¿otro? s/n s
Ingrese un valor: 5.2
¿otro? s/n n
El promedio es: 2.65

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 19

Sucesión de Fibonacci

La **sucesión de Fibonacci** es una sucesión infinita de números naturales que inicia con 0 y 1, y a partir de ahí cada elemento es la suma de los dos anteriores:
0, 1, 1, 2, 3, 5, 8, 13, 21, 34, ...
Cada elemento de esta sucesión se llama **número de Fibonacci**.

anterior ← 0 mostrar(0)
ultimo ← 1 mostrar(1)
repetir
 nuevo ← ultimo + anterior
 mostrar(nuevo)
 anterior ← ultimo
 ultimo ← nuevo
hasta nuevo > tope

La sucesión fue descrita por Fibonacci, en su libro *Liber Abaci*, como la solución a un problema de la cría de conejos.
Antes de que Fibonacci escribiera su trabajo, la sucesión de los números de Fibonacci había sido descubierta por matemáticos indios tales como Gopala (antes de 1135) y Hemachandra (1150),

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 20

Información adicional


Resolución de Problemas y Algoritmos Dr. Alejandro J. García 21

Fibonacci (1170 -1250)

Leonardo de Pisa, matemático italiano. El apodo de su padre era *Bonacci* (bien intencionado) y él recibió el apodo *Fibonacci: filius* (hijo de) Bonacci. Su padre era comerciante, y Fibonacci de joven vivió en África, donde estudió con los matemáticos árabes más destacados de ese tiempo. Allí aprendió el sistema de numeración árabe (decimal). Consciente de la superioridad de este sistema comparado con el romano, en 1202, a los 32 años de edad, publicó lo que había aprendido en el **Liber Abaci** (libro del ábaco o libro de los cálculos), mediante el cual se introdujo en Europa el sistema decimal que reemplazaría al romano.
http://es.wikipedia.org/wiki/Leonardo_de_Pisa



Resolución de Problemas y Algoritmos Dr. Alejandro J. García 22



Una página del libro *Liber Abaci* en la [Biblioteca Nazionale di Firenze](http://www.biblioteca.nazionale.it) mostrando en el recuadro de la derecha la secuencia de Fibonacci en números Romanos y números Árabigos.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 23

Sistema de numeración decimal (base 10)

- El sistema de numeración decimal se considera uno de los avances más significativos de las matemáticas.
- La mayoría de los historiadores coinciden en afirmar que tuvo su origen en la India (Tamil) en 300 aC (pero también se especula que tuviera sus orígenes en China).
- Este sistema de numeración llegó a **Oriente Medio** hacia el año 670. **al-Jwarizmi** escribió el libro "Acerca de los cálculos con los números de la India" cerca de el año 825.
- En Europa se utilizaban los números Romanos, pero **Fibonacci**, que había estudiado en **Bugia** (en la actual **Argelia**), contribuyó a la difusión por Europa del sistema árabe con su libro *Liber Abaci*, publicado en 1202.

http://es.wikipedia.org/wiki/Números_arábigos

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 24

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:
"Resolución de Problemas y Algoritmos. Notas de Clase". Alejandro J. García. Universidad Nacional del Sur. (c)2016

Evolución de los símbolos de los dígitos											
1200	Europeo	0	1	2	3	4	5	6	7	8	9
	Arábico-Índico	٠	١	٢	٣	٤	٥	٦	٧	٨	٩
670	Arábico-Índico Oriental (Persa y Urdu)	۰	۱	۲	۳	۴	۵	۶	۷	۸	۹
	Devanagari (Hindi)	०	१	२	३	४	५	६	७	८	९
300aC	Tamil (India)	௦	௧	௨	௩	௪	௫	௬	௭	௮	௯

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 25

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:
 “Resolución de Problemas y Algoritmos. Notas de Clase”. Alejandro J. García. Universidad Nacional del Sur. (c)2016