

ANÁLISIS Y DESARROLLO DE SOFTWARE

LÓGICA PROPOSICIONAL

ESTRUCTURAS CÍCLICAS

6. Estructuras cíclicas indeterminadas.

A continuación, se definen los tipos de estructuras de programación cíclicas, las características de cada estructura y ejemplos de las mismas.



Estructuras Cíclicas Determinadas: son aquellas estructuras donde el número de ejecuciones (iteraciones) se conoce antes de ejecutar el ciclo.

Este tipo de estructura tiene la siguiente forma:

```

PARA <variable> <expresion1> hasta <expresion2> paso <expresion3> haga
    <acciones a repetir>
FIN PARA
  
```

Y tiene el siguiente significado:

Dado un valor inicial expresion1 que es asignado a la variable, esta se irá aumentando o disminuyendo de acuerdo a la expresion3 hasta llegar a la expresion2. Tener en cuenta que, si se omite el paso, significa que la variable aumentará de uno en uno (valor por defecto).

Estructuras Cíclicas Indeterminadas: son aquellas estructuras donde el número de ejecuciones (iteraciones) no se conoce con exactitud, y depende la cantidad de las ejecuciones de una condición o variable dentro del ciclo.

Este tipo de estructura tiene la siguiente forma:

```

MIENTRAS QUE <condición> haga
    <acciones a repetir>
FIN MIENTRAS
  
```

Y tiene el siguiente significado:

Dada una condición, que debe de cumplirse para que se siga ejecutando las acciones dentro del ciclo. Una vez la condición deja de cumplirse, el ciclo ya no se ejecuta.

Los ciclos se deben de programar para que se ejecuten un número finito de veces, de lo contrario nos encontraremos con un ciclo infinito y el algoritmo no funcionará.

Para conseguir que el ciclo sólo se repita un número finito de veces, debe de existir una condición de salida del mismo, es decir, una instrucción o situación en la que ya no sea necesario seguir repitiendo las instrucciones.

Con la herramienta LPP se pueden crear aplicaciones que hagan uso de las estructuras cíclicas PARA, MIENTRAS y REPITA, permitiendo adquirir los fundamentos necesarios para el manejo de estas estructuras de programación.

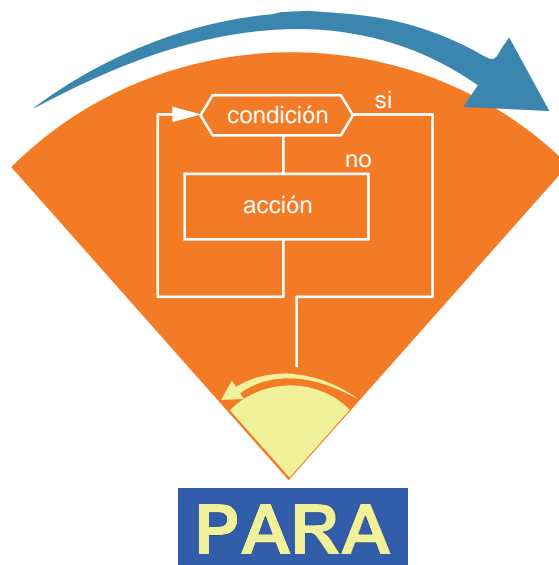
6.1. La estructura cíclica “PARA”.

La estructura cíclica PARA, permite ejecutar una serie de instrucciones un número determinado de veces. Es habitual en programación que existan instrucciones que se deben ejecutar cíclicamente cierta cantidad de veces. Gracias a la estructura PARA, estas instrucciones se escriben una sola vez dentro de la estructura cíclica y la configuración de esta estructura es la que determina cuántas veces se deben ejecutar.

Sintaxis de una estructura Cíclica “PARA”.

Sintaxis LPP
Para variable <- valor Inicial Hasta valor Final Haga //código que se desea repetir Fin Para
Ejemplo
Para x<--- 1 Hasta 100 Haga Escriba “Hola” Fin Para

El ejemplo anterior presenta 100 veces la palabra Hola en pantalla.



EJEMPLO: programa para el uso de ciclo “PARA”.

Se requiere una aplicación que lea el nombre de 3 estudiantes de un salón de clase, las 2 notas parciales de cada uno y presente un mensaje con sus nombres y notas finales. Si la nota final es inferior a 3, presentar el mensaje “REPROBADO”, en caso contrario presentar el mensaje “APROBADO” a cada estudiante.

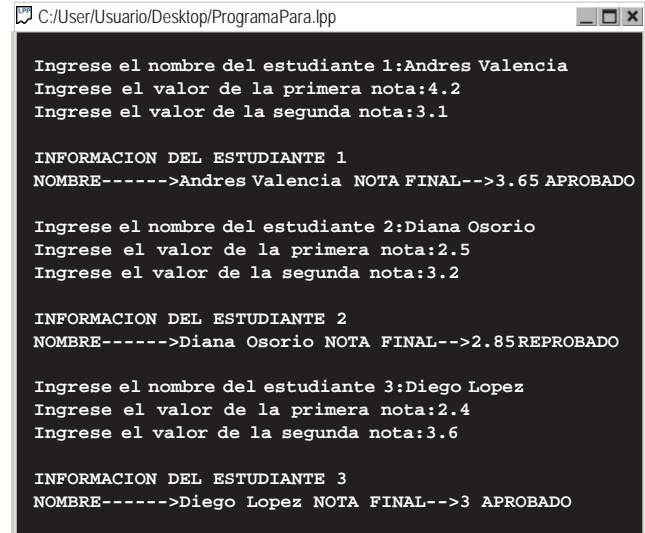
```
//Declaración de Variables
Cadena [25] nombre
Real nota1, nota2, nota3, notaFinal

Inicio
//Configuración del Ciclo PARA
//La variable estudiante es la que controla el
ciclo

Para estudiante <---- 1 Hasta 3 Haga
//Lectura de los datos de entrada
escriba "Ingrese el nombre del estudiante:"
,estudiante,":
lea nombre
escriba "Ingrese el valor de la primera nota:"
lea nota1
escriba "Ingrese el valor de la segunda nota:"
lea nota2

//Cálculo de la nota final
notaFinal <- (nota1 + nota2) / 2
//Escritura de la salida
llamar nueva_linea
escriba "INFORMACION DEL ESTUDIANTE"
,estudiante
llamar nueva_linea
escriba "NOMBRE ----->",nombre
llamar nueva_linea
escriba "NOTA FINAL-->",notaFinal
//Estructurar Condicional Doble
Si notaFinal < 3 Entonces
escriba "REPROBADO"
Sino
escriba "APROBADO"
Fin Si
llamar nueva_linea
llamar nueva_linea
//Fin del ciclo PARA
Fin
```

El resultado del programa para el uso de ciclo "PARA" es el siguiente:



```
C:/User/Usuario/Desktop/ProgramaPara.lpp

Ingrese el nombre del estudiante 1:Andres Valencia
Ingrese el valor de la primera nota:4.2
Ingrese el valor de la segunda nota:3.1

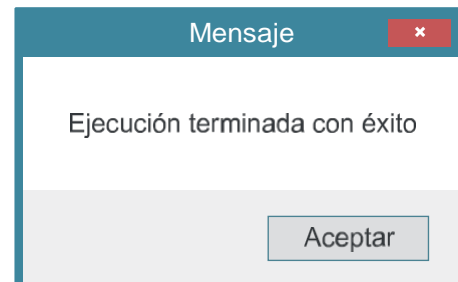
INFORMACION DEL ESTUDIANTE 1
NOMBRE----->Andres Valencia NOTA FINAL-->3.65 APROBADO

Ingrese el nombre del estudiante 2:Diana Osorio
Ingrese el valor de la primera nota:2.5
Ingrese el valor de la segunda nota:3.2

INFORMACION DEL ESTUDIANTE 2
NOMBRE----->Diana Osorio NOTA FINAL-->2.85 REPROBADO

Ingrese el nombre del estudiante 3:Diego Lopez
Ingrese el valor de la primera nota:2.4
Ingrese el valor de la segunda nota:3.6

INFORMACION DEL ESTUDIANTE 3
NOMBRE----->Diego Lopez NOTA FINAL-->3 APROBADO
```



Se puede observar que el programa se ejecutó las tres veces indicadas en el ciclo para. Para este ejemplo, cada vez que se ejecute el programa, se repetirán tres veces las instrucciones indicadas en el ciclo.

Para cambiar el número de ejecuciones deberá cambiar la condición del ciclo PARA.

6.2. La estructura cíclica "MIENTRAS".

La estructura cíclica MIENTRAS, permite ejecutar una serie de instrucciones un número indeterminado de veces. La cantidad de veces que se repite el ciclo MIENTRAS depende del cumplimiento de una condición, por esta razón es frecuente que el programador no conozca de antemano cuántas veces el ciclo será ejecutado y esta es la principal diferencia con el Ciclo PARA.

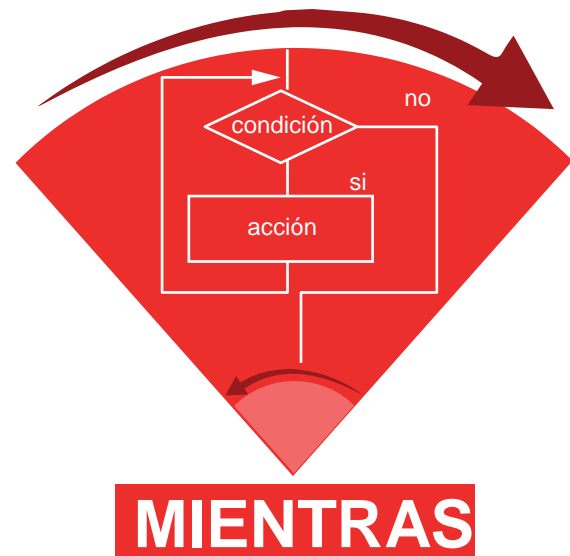
¿Qué pasaría si no quisiéramos que nuestro Robot Cíclico realice su recorrido de manera automática un número determinado de veces, sino que lo hiciera hasta que una condición suceda, por ejemplo, una orden del usuario?

La estructura cíclica MIENTRAS soluciona esta necesidad, donde cada vez que se ejecuta el ciclo, se evalúa si la condición todavía es verdadera para saber si se debe realizar un nuevo ciclo o no.

Sintaxis de una estructura cíclica “MIENTRAS”.

Sintaxis LPP
Mientras condición Haga //código que se repite mientras la condición sea //verdadera Fin Mientras
Ejemplo
Mientras ahorro<10000 Haga Lea Dinero Ahorro<-ahorro+dinero Fin Mientras

El ejemplo anterior se ejecuta hasta que el ahorro es de \$100.000 o mas.



EJEMPLO: programa para el uso de ciclo “MIENTRAS”.

Se requiere una aplicación que lea constantemente el nombre de los estudiantes de un salón de clase, las 2 notas parciales de cada uno y presente un mensaje con sus nombres y notas finales. Si la nota final es inferior a 3, presentar el mensaje “REPROBADO”, en caso contrario presentar el mensaje “APROBADO” a cada estudiante. Repetir este proceso hasta que alguno de ellos tenga una nota final por debajo de 2

```
//Declaración de Variables
Cadena [25] nombre
Real nota1, nota2, nota3, notaFinal

Inicio
    notaFinal <- 99 //se da un valor cualquiera mayor a 2 para que entre al ciclo
    //Configuración del Ciclo MIENTRAS
    Mientras notaFinal >= 2 Haga
        //Lectura de los datos de entrada
        //Lectura de los datos de entrada
```



```

escriba "Ingrese el nombre del estudiante:"
lea nombre
escriba "Ingrese el valor de la primera nota:"
lea nota1
escriba "Ingrese el valor de la segunda nota:"
lea nota2
//Cálculo de la nota final
notaFinal <--- (nota1 + nota2) / 2
//Escritura de la salida
llamar nueva_linea
escriba "INFORMACION DEL ESTUDIANTE"
llamar nueva_linea
escriba "NOMBRE ----->", nombre
llamar nueva_linea
escriba "NOTA FINAL-->", notaFinal
//Estructurar Condicional Doble
Si notaFinal < 3 Entonces
escriba "REPROBADO"
Sino
escriba "APROBADO"
Fin Si
llamar nueva_linea
llamar nueva_linea
//Fin del ciclo MIENTRAS
Fin

```

El resultado del programa para el uso de ciclo "MIENTRAS" es el siguiente:

```

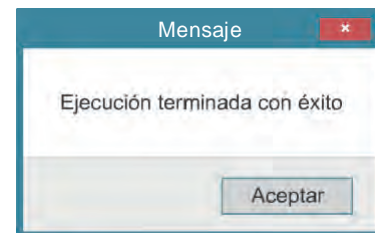
C:/User/Usuario/Desktop/ProgramaPara.lpp
Ingrese el nombre del estudiante :Juan Arias
Ingrese el valor de la primera nota:4
Ingrese el valor de la segunda nota:2

INFORMACION DEL ESTUDIANTE
NOMBRE----->Juan Arias NOTA FINAL-->3 APROBADO

Ingrese el nombre del estudiante :Fernando Cardona
Ingrese el valor de la primera nota:1
Ingrese el valor de la segunda nota:2.3

INFORMACION DEL ESTUDIANTE
NOMBRE----->Fernando Cardona NOTA FINAL-->1.65 REPROBADO

```



Se puede observar que el programa se ejecutó solamente dos veces porque la nota final del segundo estudiante fue menor a 2. Pero si este programa se ejecuta de nuevo, la cantidad de veces de ejecución del ciclo puede ser diferente, dependiendo de los datos ingresados por el usuario

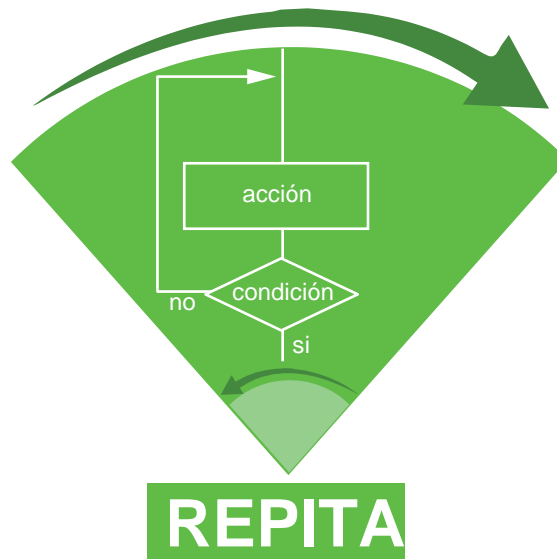
6.3. La estructura cíclica "REPITA".

La estructura cíclica REPITA, al igual que la estructura cíclica MIENTRAS, se ejecuta un número indeterminado de veces, estas dos estructuras tienen un comportamiento similar, presentando su principal diferencia en el lugar de la estructura donde se evalúa la condición, dado que la estructura MIENTRAS evalúa la condición del ciclo al inicio del mismo y la estructura REPITA lo hace al final del mismo, de este modo, en la estructura cíclica REPITA, el programador garantiza que el ciclo se ejecuta al menos una vez.

Sintaxis de una estructura Cíclica "REPITA".

Sintaxis LPP	Ejemplo
Repita //código que se repite mientras la condición sea //verdadera Fin Repita	Repita Lea Dinero Ahorro<-ahorro+dinero Hasta ahorro >= 100000

El ejemplo anterior se ejecuta hasta que el ahorro es de \$100.000 o más.



EJEMPLO: Programa para el uso de ciclo “MIENTRAS”.

```
//Declaración de Variables
Cadena [25] nombre
Real nota1, nota2, nota3, notaFinal

Inicio
  //Inicio del ciclo Repita
  Repita
  //Lectura de los datos de entrada
  llamar nueva_linea
  escriba "Ingrese el nombre del estudiante:"
  lea nombre
  escriba "Ingrese el valor de la primera nota:"
  lea nota1
  escriba "Ingrese el valor de la segunda nota:"
  lea nota2
  //Cálculo de la nota final
  notaFinal <- (nota1 + nota2) / 2
  //Escritura de la salida
  llamar nueva_linea
  llamar nueva_linea
  escriba "INFORMACION DEL ESTUDIANTE"
  llamar nueva_linea
  escriba "NOMBRE ----->", nombre
  llamar nueva_linea
  escriba "NOTA FINAL-->", notaFinal
  Si notaFinal < 3 Entonces
    escriba "REPROBADO"
  Sino
    escriba "APROBADO"
  Fin Si
  Hasta notaFinal < 2
  //Fin del ciclo REPITA
Fin
```

El resultado del programa para el uso de ciclo “MIENTRAS” es el siguiente:

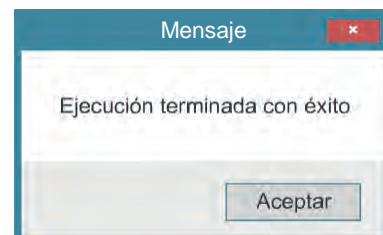
```
C:/User/Usuario/Desktop/ProgramaPara.lpp

Ingrese el nombre del estudiante:Diego
Ingrese el valor de la primera nota:2
Ingrese el valor de la segunda nota:3.2

INFORMACION DEL ESTUDIANTE NOMBRE ----- >Diego
NOTA FINAL-->2.6 REPROBADO
Ingrese el nombre del estudiante:Carlos
Ingrese el valor de la primera nota:3
Ingrese el valor de la segunda nota:4

INFORMACION DEL ESTUDIANTE NOMBRE ----- >Carlos
NOTA FINAL-->3.5 APROBADO
Ingrese el nombre del estudiante:Jorge
Ingrese el valor de la primera nota:1
Ingrese el valor de la segunda nota:1.5

INFORMACION DEL ESTUDIANTE NOMBRE ----- >Jorge
NOTA FINAL-->1.25 REPROBADO
```



El programa con el ciclo REPITA presenta el mismo comportamiento que el programa con el ciclo MIENTRAS, el cambio está en el código utilizado en cada una de las estructuras; por lo tanto, el uso de un ciclo o el otro es indiferente cuando se requiere una estructura cíclica indeterminada.

7. Armando el rompecabezas con estructuras de programación.

Cómo vimos, las estructuras de programación son de tipo secuenciales, condicionales y cíclicas y con estos 3 tipos de estructuras se pueden desarrollar una gran cantidad de aplicaciones de software, la clave está en la organización lógica que se le den a las instrucciones y estructuras para resolver un problema mediante un programa de computador.

La organización de las estructuras en un programa depende exclusivamente del problema que se desea resolver, por lo tanto, es posible tener un programa con una estructura condicional al interior de una estructura cíclica o un ciclo al interior de otro (ciclos anidados), o una estructura cíclica al interior de una condicional y estructuras secuenciales al interior o por fuera de estructuras cíclicas o condicionales. Por esta razón, el desarrollo de un programa es similar a la construcción de un rompecabezas donde cada elemento debe estar en el lugar adecuado para interactuar con los demás elementos y así construir una solución integral a un problema determinado.

