	<b>Manual de prácticas del Laboratorio de Programación básica</b>	Código:	MADO-18
		Versión:	01
		Página	134/184
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

## Guía práctica de estudio 09: Estructuras de repetición

---




---

### *Elaborado por:*


Ing. Jorge A. Solano Gálvez  
Guadalupe Lizeth Parrales Romay

### *Revisado por:*

M.C. Edgar E. García Cano

### *Autorizado por:*

M.C. Alejandro Velázquez Mena

	<b>Manual de prácticas del Laboratorio de Programación básica</b>	Código:	MADO-18
		Versión:	01
		Página	135/184
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

## Guía de práctica de estudio 09: Estructuras de repetición

### Objetivo:

Elaborar programas en FORTRAN para la resolución de problemas básicos que incluyan estructuras de repetición.


### Actividades:

- Crear programa con las estructuras *while* y *do-while*.
- Anidar estructuras de selección dentro de estructuras repetitivas.

### Introducción

Las estructuras de repetición son las llamadas estructuras cíclicas, iterativas o de bucles. Permiten ejecutar un conjunto de instrucciones de manera repetida (o cíclica) mientras que la expresión lógica a evaluar se cumpla (sea verdadera).

En lenguaje FORTRAN existen dos estructuras de repetición: *do-while* y *do*. La estructura *do-while* es una estructura repetitiva de propósito general, la estructura *do* se ejecuta un número predeterminado de veces.

	<b>Manual de prácticas del Laboratorio de Programación básica</b>	Código:	MADO-18
		Versión:	01
		Página	136/184
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

## Licencia GPL de GNU

El software presente en esta guía práctica es libre bajo la licencia GPL de GNU, es decir, se puede modificar y distribuir mientras se mantenga la licencia GPL.

```

/*
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program. If not, see <http://www.gnu.org/licenses/>.
 *
 * Author: Jorge A. Solano
 *
 */

```


## Estructura de control repetitiva DO WHILE

La estructura repetitiva (o iterativa) *do-while* primero valida la expresión lógica y si ésta se cumple (es verdadera) procede a ejecutar el bloque de instrucciones de la estructura. Si la condición no se cumple se continúa el flujo normal del programa sin ejecutar el bloque de la estructura, es decir, el bloque se puede ejecutar de cero a *n* veces. Su sintaxis es la siguiente:

```

do while (expresión_lógica)
c   Bloque de código a repetir
c   mientras que la expresión
c   lógica sea verdadera.
end do

```

	<b>Manual de prácticas del Laboratorio de Programación básica</b>	Código:	MADO-18
		Versión:	01
		Página	137/184
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

### Código (estructura de repetición while)

```

program tablaMultiplicar

c Este programa genera la tabla de multiplicar de un número dado.
c El número se lee desde la entrada estándar (teclado).

integer num, cont

cont = 0

write (*,*) '----- Tabla de multiplicar -----'
write (*,*) 'Ingrese un número:'
read (*,*) num

write (*,*) 'La tabla de multiplicar del',num,'es'

do while (cont .LT. 10)
    cont = cont + 1
    write (*,*) num,'x',cont,'=',num*cont
enddo

stop
end

```

### Código (estructura de repetición while)

```


program infinito

c El siguiente es un ciclo infinito
c por que la condición siempre es verdadera.

do while (.TRUE.)
    write (*,*) 'Ciclo infinito.'
    write (*,*) 'Para terminar presione ctrl + c'
end do

stop
end

```

	<b>Manual de prácticas del Laboratorio de Programación básica</b>	Código:	MADO-18
		Versión:	01
		Página	138/184
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

### Código (estructura de repetición do while)

```

program promCalif

c Este programa debe obtener el promedio de calificaciones
c ingresadas por el usuario. Las calificaciones se leen
c desde la entrada estándar (teclado).
c La inserción de calificaciones termina cuando el usuario
c presiona una tecla diferente a las letras 'S' o 's'

character op
real*8 sum, calif
integer veces

veces = 0
sum = 0
op = 's'


do while ((op .EQ. 'S').OR.(op .EQ. 's'))
  write (*,*) 'Suma de calificaciones'
  write (*,*) 'Ingrese calificación'
  read (*,*) calif
  veces = veces + 1
  sum = sum + calif

  write (*,*) '¿Desea sumar otra?'
  read (*,*) op
enddo

write (*,*) 'El promedio de las calificaciones es:'
write (*,*) sum/veces

stop
end

```

	<b>Manual de prácticas del Laboratorio de Programación básica</b>	Código:	MADO-18
		Versión:	01
		Página	139/184
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

## Código (estructura de repetición do while)

```

program calculadora

c Este programa genera una calculadora básica

integer op, uno, dos

do while (op .NE. 5)


  write (*,*) '----- Calculadora -----'
  write (*,*) '¿Que desea hacer?'
  write (*,*) '1) Sumar'
  write (*,*) '2) Restar'
  write (*,*) '3) Multiplicar'
  write (*,*) '4) Dividir'
  write (*,*) '5) Salir'

  read (*,*) op

  select case (op)
    case (1)
      write (*,*) 'Sumar'
      write (*,*) 'Introduzca los números a sumar'
      read (*,*) uno, dos
      write (*,*) uno, '+', dos, '=', uno+dos
    case (2)
      write (*,*) 'Restar'
      write (*,*) 'Introduzca los números a restar'
      read (*,*) uno, dos
      write (*,*) uno, '-', dos, '=', uno-dos
    case (3)
      write (*,*) 'Multiplicar'
      write (*,*) 'Introduzca los números a multiplicar'
      read (*,*) uno, dos
      write (*,*) uno, '*', dos, '=', uno*dos
    case (4)
      write (*,*) 'Dividir'
      write (*,*) 'Introduzca los números a dividir'
      read (*,*) uno, dos
      write (*,*) uno, '/', dos, '=', uno/dos
    case (5)
      write (*,*) 'Salir'

  end select
enddo
stop
end

```

	<b>Manual de prácticas del Laboratorio de Programación básica</b>	Código:	MADO-18
		Versión:	01
		Página	140/184
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

## Estructura de control de repetición DO

Lenguaje FORTRAN posee la estructura de repetición **do** la cual permite realizar repeticiones cuando se conoce el número de elementos que se quiere recorrer. La sintaxis que generalmente se usa es la siguiente:

```
do var = inicio, fin[, incremento]

    !Bloque de código
    !a ejecutar

enddo
```

La estructura do realiza 3 acciones por sí misma antes o después de ejecutar el bloque de código entre llaves. La primera parte es la inicialización, en la cual se define la variable y se inicializa su valor; esta parte solo se ejecuta una vez cuando se ingresa al ciclo. Fin marca el valor final que puede tomar la variable. El incremento indica el valor con el que la variable es modificada después de cada ejecución del ciclo DO, esta parte es opcional, cuando no aparece, su valor por defecto es 1. El incremento puede ser positivo o negativo.

### Código (estructura de repetición do)


```
program doSinIncremento

c Este programa debe obtener el promedio de 5 calificaciones
c ingresadas por el usuario. Las calificaciones se leen
c desde la entrada estándar (teclado).

real*8 sum, calif
integer veces

sum = 0

c El incremento es opcional, si no se escribe por defecto es uno
do veces = 1, 5
write (*,*) 'Suma de calificaciones'
write (*,*) 'Ingrese la calificación', veces
```

	<b>Manual de prácticas del Laboratorio de Programación básica</b>	Código:	MADO-18
		Versión:	01
		Página	141/184
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

```

        read (*,*) calif
        sum = sum + calif
    end do

    write (*,*) 'El promedio de las calificaciones es:'
    write (*,*) sum/(veces-1)

    stop
end

```

### Código (estructura de repetición do)

```

program doConIncremento

c Este programa debe obtener el promedio de 5 calificaciones
c ingresadas por el usuario. Las calificaciones se leen
c desde la entrada estándar (teclado).

real*8 sum, calif
integer veces, cont

sum = 0
cont = 1


do veces = 1, 10, 2
    write (*,*) 'Suma de calificaciones'
    write (*,*) 'Ingrese la calificación', cont
    read (*,*) calif
    sum = sum + calif
    cont = cont + 1
end do

write (*,*) 'El promedio de las calificaciones es:'
write (*,*) sum/(cont-1)

stop
end

```



	<b>Manual de prácticas del Laboratorio de Programación básica</b>	Código:	MADO-18
		Versión:	01
		Página	142/184
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

## Sentencia EXIT


La sentencia EXIT produce la salida inmediata de un ciclo. Sirve para crear una repetición controlada por una expresión lógica. Con la ayuda de las estructuras DO e IF y la sentencia EXIT se puede crear un ciclo WHILE, de la siguiente manera:

```
do
    !Bloque de código
    !a ejecutar

    IF (expresión_lógica) EXIT

    !Bloque de código
    !a ejecutar
end do
```

La estructura DO genera un ciclo que se ejecuta MIENTRAS que la expresión lógica no se cumpla (sea falsa). Cuando la expresión lógica de la estructura de selección IF se cumple (es verdadera), se ejecuta la sentencia EXIT, cuyo efecto es transferir el control fuera del ciclo DO, a la primera sentencia siguiente a END DO, de manera similar a un ciclo WHILE. La validación de la condición (estructura IF) se puede situar en cualquier parte de la estructura DO.

	<b>Manual de prácticas del Laboratorio de Programación básica</b>	Código:	MADO-18
		Versión:	01
		Página	143/184
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Código (sentencia exit)

```

PROGRAM dec2bin

c Programa que permite convertir un número en base 10 a base 2.

INTEGER decimal

WRITE (*,*) 'Ingrese un número decimal'
READ (*,*) decimal

DO
  IF (decimal .LE. 0) EXIT
  WRITE (*,*) MOD(decimal,2)
  decimal = decimal / 2
END DO

WRITE (*,*) 'El número se lee de abajo hacia arriba'

STOP
END

```

## Sentencia CYCLE

La sentencia CYCLE detiene la ejecución de la iteración actual y devuelve el control al inicio del ciclo, continuando la ejecución de la iteración siguiente. La sentencia CYCLE se puede usar dentro de cualquier ciclo iterativo:


```

do while (expresión_lógica)
  !Bloque de código
  !a ejecutar

  IF (expresión_lógica) CYCLE

  !Bloque de código
  !a ejecutar
end do

```

	<b>Manual de prácticas del Laboratorio de Programación básica</b>	Código:	MADO-18
		Versión:	01
		Página	144/184
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

Código (sentencia cycle)

```

PROGRAM sumaPares

c Programa que suma los primeros 5 números pares a partir de un número dado.

INTEGER numero, suma, contador
suma = 0
contador = 1


WRITE (*,*) 'Ingrese el número inicial'
READ (*,*) numero

DO WHILE (contador .LE. 5)
    numero = numero + 1
c si el número es par se devuelve el control al inicio del ciclo,
c es decir, no se ejecutan las líneas que están debajo de la estructura IF
    IF (MOD(numero, 2) .EQ. 1) CYCLE
    contador = contador + 1
    suma = suma + numero
END DO

WRITE (*,*) 'La suma de los números pares es: ', suma

STOP
END

```

	<b>Manual de prácticas del Laboratorio de Programación básica</b>	Código:	MADO-18
		Versión:	01
		Página	145/184
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

## Estructuras repetitivas con nombre

Se puede asignar un nombre a una estructura agregando 'nombre:' antes de la palabra reservada DO y 'nombre' después del final del ciclo (END DO). Los nombres son opcionales, pero si se usan deben ser en ambas etiquetas, es decir:

```
[nombre:] do
    !Bloque de código
    !a ejecutar

    IF (expresión_lógica) CYCLE [nombre]


    !Bloque de código
    !a ejecutar

    IF (expresión_lógica) EXIT [nombre]
end do [nombre]
```

```
[nombre:] do var = inicio, fin[, incremento]
    !Bloque de código
    !a ejecutar

    IF (expresión_lógica) CYCLE [nombre]

    !Bloque de código
    !a ejecutar
end do [nombre]
```

	<b>Manual de prácticas del Laboratorio de Programación básica</b>	Código:	MADO-18
		Versión:	01
		Página	146/184
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

### Código (ciclos con etiquetas)

**PROGRAM** etiquetas

c Programa que genera 3 números consecutivos (1, 2 y 3) y los divide  
c entre 3 números proporcionados por el usuario. Si el número  
c ingresado es cero el programa termina su ejecución.

```

REAL dividendo, divisor
INTEGER contadorExterno, contadorInterno
contadorExterno = 1

externo: DO WHILE (contadorExterno .LE. 3)
    dividendo = contadorExterno
    WRITE (*,*) 'Dividendo: ', dividendo
    contadorInterno = 0
    interno: DO
        WRITE (*,*) 'Ingrese el número inicial'
        READ (*,*) divisor


        IF (divisor .EQ. 0) THEN
            WRITE (*,*) 'No se puede dividir entre 0'
            EXIT externo
        ENDIF

        WRITE (*,*) dividendo, '/', divisor, '=', dividendo/divisor
        contadorInterno = contadorInterno + 1

        IF (contadorInterno .EQ. 2) EXIT interno
    END DO interno
    contadorExterno = contadorExterno + 1
END DO externo

STOP
END

```

	<b>Manual de prácticas del Laboratorio de Programación básica</b>	Código:	MADO-18
		Versión:	01
		Página	147/184
		Sección ISO	8.3
		Fecha de emisión	20 de enero de 2017
Facultad de Ingeniería		Área/Departamento: Laboratorio de computación salas A y B	
La impresión de este documento es una copia no controlada			

## Bibliografía

- Oracle (2010). Fortran 77 Lenguaje Reference. Consulta: Julio de 2015. Disponible en: <http://docs.oracle.com/cd/E19957-01/805-4939/>
- Stanford University (1995). Fortran 77 Tutorial. Consulta: Julio de 2015. Disponible en: [http://web.stanford.edu/class/me200c/tutorial\\_77/](http://web.stanford.edu/class/me200c/tutorial_77/)