compAs

Grupo de capacitación e investigación pedagogica

LA PROGRAMACIÓN: EL INICIO DE LA COMPUTACIÓN

Maribel Revelo Espinosa
Catalina Vargas Pérez
Jaime Gabriel Espinosa Izquierdo
John Fernado Granados Romero
Francisco Lenín Moran Peña

LA PROGRAMACIÓN: EL INICIO DE LA COMPUTACIÓN

Maribel Revelo Espinosa
Catalina Vargas Pérez
Jaime Gabriel Espinosa Izquierdo
John Fernado Granados Romero
Francisco Lenín Moran Peña

LA PROGRAMACIÓN: EL INICIO DE LA COMPUTACIÓN

compAS
Grupo de capacitación e investigación pedagógica

Título original:
LA PROGRAMACIÓN: EL INICIO DE LA COMPUTACIÓN

Primera edición: marzo 2020

© 2020, Maribel Revelo Espinosa
Catalina Vargas Pérez
Jaime Gabriel Espinosa Izquierdo
John Fernado Granados Romero
Francisco Lenín Moran Peña
Docentes de la Universidad de Guayaquil

Publicado por acuerdo con los autores.
© 2020, Editorial Grupo Compás
Guayaquil-Ecuador

Grupo Compás apoya la protección del copyright, cada uno de sus textos han sido sometido a un proceso de evaluación por pares externos con base en la normativa del editorial.

El copyright estimula la creatividad, defiende la diversidad en el ámbito de las ideas y el conocimiento, promueve la libre expresión y favorece una cultura viva. Quedan rigurosamente prohibidas, bajo las sanciones en las leyes, la producción o almacenamiento total o parcial de la presente publicación, incluyendo el diseño de la portada, así como la transmisión de la misma por cualquiera de sus medios, tanto si es electrónico, como químico, mecánico, óptico, de grabación o bien de fotocopia, sin la autorización de los titulares del copyright.

Editado en Guayaquil - Ecuador

ISBN: 978-9942-33-187-8

compÁs
Grupo de capacitación e investigación pedagógica

Cita.
M. Revelo, C. Vargas, J. Espinosa, J. Granados, F. Moran (2020) LA PROGRAMACIÓN: EL INICIO DE LA COMPUTACIÓN , Editorial Grupo Compás, Guayaquil Ecuador, 69 pag

ÍNDICE

UNIDAD I

Generalidades.

Programación, definición.
Programa, definición.
Lenguaje de Programación.
Clasificación de los Lenguajes de Programación.
Lenguaje de Máquina.
Lenguaje de Bajo Nivel.
Lenguaje de Alto Nivel.

UNIDAD I

Programación, definición

La programación es la técnica mediante la cual se escriben *programas*. ¿Pero qué es un *programa*? Veamos la definición de este término.

El término *programa* es utilizado en diversos aspectos. Por ejemplo, los canales de televisión emiten *programas* cada media hora o cada hora, uno después del otro, en secuencia. Cuando se va a realizar alguna sesión solemne o acto cívico, el secretario encargado o el maestro de ceremonias lee también un *programa*.

Cuando un amigo le cuenta a otro que el fin de semana saldrá a comer y a bailar con alguna chica, le dice que tiene un programa para el viernes.

Fíjese bien. Los tres tipos de programas mencionados, tienen algunas cosas en común. Lo principal es su estricto orden. Los programas de TV son a las 07h00: las noticias, 08h00: los deportes, 09h00: cosas de casas, 09h30: etc… El programa del maestro de ceremonias es: 1ª) Ingreso de las Autoridades, 2ª) Himno Nacional, 3ª) Palabras de Inauguración, 4ª) etc… El programa para el viernes es: Primero: la invito a comer, Segundo: vamos a bailar, Tercero: ¡¡¡a lo que la noche nos inspire (!!!), etc...

Programa

Es la secuencia ordenada y lógica de pasos a seguir para la solución de un problema. Un programa para la computadora tiene una serie de órdenes o instrucciones que el computador ejecuta paso a paso para obtener un resultado. Estas instrucciones del programa deben estar escritas o "codificadas" en un determinado lenguaje. Esto es lo que se conoce con el nombre Lenguaje de Programación.

Lenguaje de programación

Un lenguaje de programación es un conjunto de palabras reservadas, denominadas instrucciones, mediante las cuales se diseñan los programas para el computador; así como el lenguaje: inglés, francés, alemán o español; es usado para que la gente se comunique entre sí, los programadores necesitan a los lenguajes de programación para comunicarse con el computador. Todo programador requiere aprender un determinado Lenguaje de programación para de esta manera, poder escribir sus programas.

Clasificación de los lenguajes de programación

Los lenguajes de programación se clasifican en tres (3) grandes grupos:

- **Lenguaje de máquina.** 010000001
- **Lenguaje de bajo nivel.** Códigos Nemotécnicos * Código Binario
- **Lenguaje de alto nivel**. Fortran, Cobol, Pascal, VFP, Visual Basic

Lenguaje de Máquina. - Son específicos de cada modelo o fabricante de computador. Están basados fundamentalmente en el código binario, es decir, en representaciones de ceros (0) y unos (1). Las ventajas de los programas escritos en lenguaje de máquina; son la, posibilidad de cargar datos y procesos directamente a la memoria del computador sin la necesidad de traducción alguna, debido a que están dados en el propio lenguaje de la máquina, lo cual supone una velocidad de ejecución superior a la de cualquier otro lenguaje de programación.

Los inconvenientes de los lenguajes de máquina superan a sus ventajas, debido a la extrema dificultad para entenderlos, codificarlos, y peor aún, modificarlos. Además, son programas esclavos de la máquina, es decir, que un mismo programa, no puede funcionar de igual manera en otro computador. Este lenguaje solo lo utilizan los técnicos y expertos en la fabricación de computadoras. La siguiente tabla muestra las representaciones en Lenguaje de máquina (código binario) de las seis primeras letras del alfabeto:

CÓDIGO BINARIO	REPRESENTACIÓN
01000001	A
01000010	B
01000011	C
01000100	D
01000101	E
01000110	F

Por ejemplo, la palabra CAFÉ en Lenguaje de máquina se escribiría así:

01000011	01000001	01000110	01000101
C	A	F	E

Representando cada letra de la palabra CAFÉ por su respectiva equivalencia en el código binario. Claro está la dificultad de codificar programas utilizando el lenguaje de máquina.

Lenguaje de Bajo Nivel. - Los lenguajes de bajo nivel utilizan códigos nemotécnicos en lugar de código binario, y los direccionamientos tanto de datos como de instrucciones, los realiza en la memoria en forma simbólica. Son menos difíciles de usar que los lenguajes de la máquina, pero al igual que ellos, dependen de la máquina en particular.

Como la máquina no puede trabajar con códigos simbólicos, estos deben ser previamente traducidos a lenguaje de máquina para poder procesarlos. Los lenguajes de bajo nivel presentan ventaja frente a los lenguajes de máquina, debido a su mayor facilidad de codificación, y en general a su velocidad de cálculo.

Sus principales inconvenientes son entre otros, la dependencia total de la máquina, lo cual imprime la transportabilidad de los programas (posibilidad de ejecutar un programa en diferentes máquinas). Además, la formación de los programadores a los programas de alto nivel, ya que existe no solamente las técnicas de

programación simbólica o nemotécnica, sino también, el conocimiento interno del computador.

Uno de los lenguajes de bajo nivel más conocidos es el **Assembler.** (Ensamblador). El Assembler fue el lenguaje de bajo nivel utilizado para la codificación de los sistemas operativos, bases de datos y demás programas de control. Esto debido a que el Assembler podía maximizar y explotar al máximo los recursos del computador. Sin embargo, los programas en Assembler son muy extensos y dependen estrictamente del modelo del computador.

Lenguaje de Alto Nivel. - Los lenguajes de alto nivel son los **más utilizados** por los programadores. Están diseñados para que las personas escriban y entiendan los programas de un modo mucho **más fácil** que los programas escritos en lenguaje de máquina o bajo nivel.

Otra razón es que los programas escritos en lenguaje de alto nivel son **independientes de la máquina;** es decir, pueden funcionar indistintamente en cualquier otro computador compatible.

Las ventajas de los programas codificados en **lenguaje de alto nivel** son entre otras: el tiempo de formación de los programadores es relativamente más corto comparado con el de los programadores en otros lenguajes, gracias a la facilidad de sus **instrucciones,** que no son otra cosa que **simples palabras en inglés** técnico.

Además, diseñar programas en lenguaje de alto nivel, es mucho más rápido y menos costoso que codificar en bajo nivel o máquina.

Pero estos lenguajes también poseen ciertas desventajas. Por ejemplo, todo programa codificado en un lenguaje de alto nivel, antes de ser utilizado por el computador, debe ser **traducido o compilado,** esto es, interpretado al lenguaje de máquina: el código binario. Esto presume, cierta demora en la ejecución de algún proceso.

Estos programas también ocupan demasiada memoria, y no aprovechan al máximo los recursos del computador de la manera como lo hacen los lenguajes de máquina y bajo nivel.

Sin embargo, los lenguajes de alto nivel son los más utilizados por los programadores, debido a su facilidad de manejo y aprendizaje. En la actualidad, todas las **aplicaciones de computación,** son desarrolladas en lenguaje de programación de alto nivel.

UNIDAD II

Introducción a los Lenguajes de Programación de Alto Nivel.

Clasificación de los Lenguajes de Programación de Alto Nivel.

Lenguajes de Programación Científicos (FORTRAN, ALGOL, PASCAL, C).

Lenguajes de Programación Comercial (COBOL, FOXPRO, VISUAL FOXPRO, VISUAL BASIC).

Lenguajes de Programación General (LOGO, BASIC).

UNIDAD II

CLASIFICACIÓN DE LOS LENGUAJES DE PROGRAMACIÓN DE ALTO NIVEL

A su vez, los lenguajes de programación de alto nivel se clasifican en **tres** grupos:

- **Lenguaje científico.**
- **Lenguaje comercial.**
- **Lenguaje general.**

Lenguajes de Programación Científicos. - Son aquellos que se utilizan en el campo de la **investigación** científica. Este tipo de lenguaje se caracteriza por efectuar **abundante cantidad de cálculos** numéricos y lógicos con los datos. Entre los principales lenguajes de programación científicos están el FORTRAN, el ALGOL, PASCAL y C.

FORTRAN (**FOR**mula **TRAN**slate = Traductor de fórmulas). - El **primer lenguaje** de programación de alto nivel y compilador, desarrollado en **1954** por la IBM.

Originalmente fue diseñado para expresar **formulas matemáticas** y a pesar de que ocasionalmente es usado para aplicaciones comerciales, es todavía un lenguaje usado para problemas **científicos,** de ingeniería y de matemática. Las instrucciones usadas en FORTRAN son muy similares al **Lenguaje aritmético.** El siguiente fragmento de programa muestra cómo se transforman grados Fahrenheit a centígrados en FORTRAN:

```
WRITE (6,*) 'Enter Fahrenheit '
READ (5,*) XFAHR
XCENT  =  (XFAHR - 32) * 5 / 9
WRITE (6,*) 'Celsius is  ', XCENT
END
```

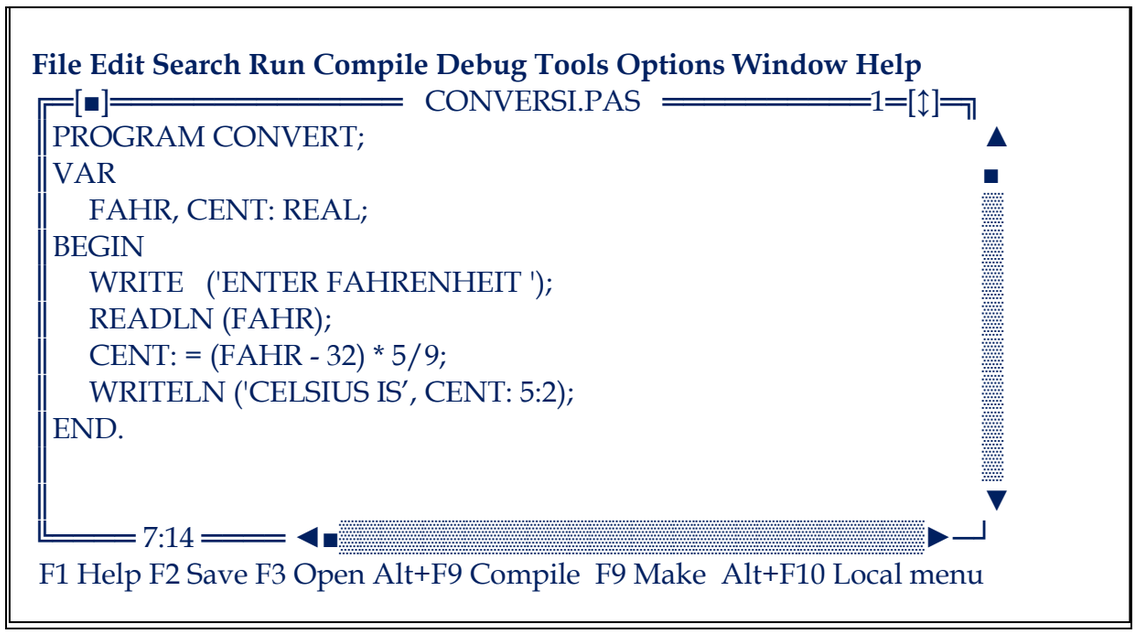
ALGOL (**ALGO**ritmic Language = Lenguaje algorítmico). - Lenguaje compilador de alto nivel que se desarrolló como un lenguaje internacional para comunicarse mediante algoritmos entre personas y también máquinas. Algol – 60 (1960) era sencillo y se utilizó ampliamente en Europa. Algol 68 (1968) era más complicado y se utilizó muy poco, pero fue la inspiración para otro lenguaje de programación, este sí muy famoso, el Pascal.

Ahora veamos el mismo programa de conversión Fahrenheit a centígrados, pero en lenguaje Algol – 68:

```
Fahrenheit
        begin
        real fahr;
        print("Enter Fahrenheit  ");
        read (fahr);
        print("Celsius is ",  (fahr – 32.0) * 5.0 / 9.0
        end
finish
```

PASCAL. - Lenguaje de programación de alto nivel de carácter científico, desarrollado por el suizo Niklaus Wirth en los comienzos de la década de los setenta, al que se le dio el nombre en honor al matemático y filósofo francés **Blaise Pascal** (1623 – 1662). Se destaca por su **programación estructurada,** que lo ha convertido en uno de los lenguajes de programación más utilizados en la formación de futuros programadores.

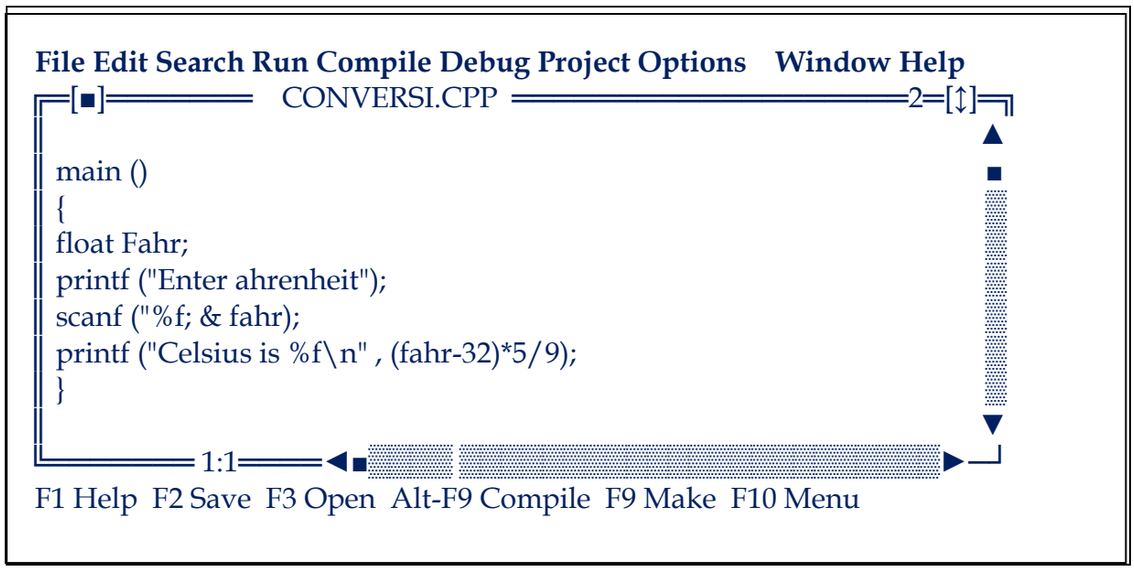
Existe un compilador más poderoso de Pascal denominado **Turbo Pascal**, creado por la empresa **Borland,** el cual tiene mayor versatilidad y le ha incluido una serie de mejoras que van desde el manejo de colores hasta la programación orientada a objetos. El siguiente programa de conversión Fahrenheit a centígrados, ahora en versión Turbo Pascal 7:



C (Científico). - Este lenguaje de programación fue desarrollado por los laboratorios **Bell**, y a pesar de que es un lenguaje de alto nivel, es capaz de manipular la computadora a bajo nivel, tal como lo haría el lenguaje Assembler (ensamblador).

Durante la segunda mitad de la década de los ochenta este lenguaje fue elegido para el desarrollo de software comercial. Por ejemplo, el sistema operativo para redes ***UNIX,*** está codificado íntegramente en lenguaje C. Comparado con otros lenguajes, el C puede parecer complicado. Su apariencia intrincada se debe a la flexibilidad de sus instrucciones. Sin embargo, el C es uno de los lenguajes de programación **más potentes** para la programación científica y matemática.

El lenguaje C se hizo tan famoso que pronto aparecieron un sinnúmero de versiones, cada uno con sus propias ventajas y mejoras. De esta manera, ha surgido el **Turbo C, Borland C, Visual C y C++.** A continuación, el programa de conversión Fahrenheit a centígrados en versión **C++:**



Lenguaje de Programación Comercial. - Son aquellos lenguajes de programación que se utilizan en el desarrollo de sistemas computacionales aplicados al campo de negocios. Utilizando lenguajes de programación comercial se codifican los sistemas con los que se trabaja en los bancos, almacenes, oficinas, etc. Entre los principales lenguajes de programación comercial tenemos:

COBOL. - (**CO**mmon **B**usiness **O**riented **L**anguaje = Lenguaje común orientado a los negocios). Durante mucho tiempo fue el principal lenguaje de programación de alto nivel orientado al campo comercial y a los negocios.

El Cobol requiere una escritura más extensa que otros lenguajes, pero el resultado es una mejor legibilidad. Adoptado formalmente en 1960, se convirtió rápidamente en el más popular lenguaje comercial, que hoy en día, se sigue utilizando.

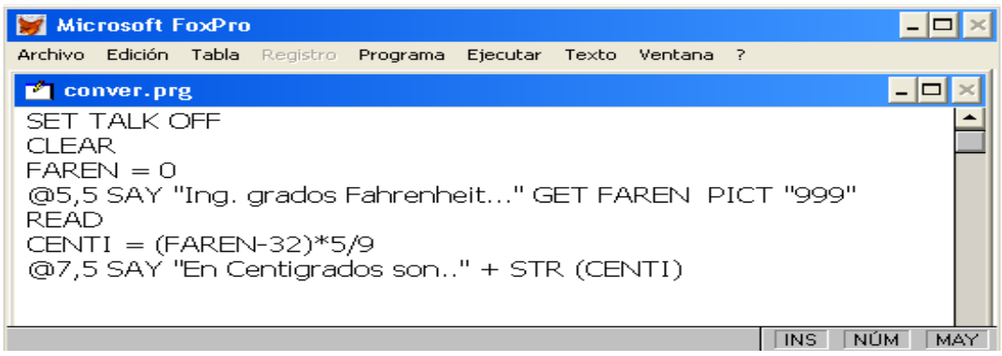
El Cobol se distribuyó en dos versiones: el MS COBOL y el RM COBOL. También se crearon versiones de Cobol para trabajar en computadoras grandes, tales como el COBOL/370 o el COBOL II. En Cobol, sea cual sea el programa, siempre tiene **cuatro divisiones**: IDENTIFICACIÓN, ENVIRONMENT, DATA y PROCEDURE DIVISIÓN. El programa a continuación muestra cómo, en la versión Microsoft COBOL (MS COBOL) se transforman grados Fahrenheit a centígrados:

```
IDENTIFICATION DIVISION.
        PROGRAM – ID. FARENCITI.
        AUTHOR.CAMILO CORONEL
        ENVIRONMENT DIVISION.
        CONFIGURATION SECTION.
        DATA DIVISION.
        WORKING – STORAGE SECTION.
        77 CENTI PIC 999.
        PROCEDURE DIVISION.
INICIO.
        DISPLAY (1, 1) ERASE.
        DISPLAY (5, 5) “Ingrese grados Fahrenheit: “
        ACCEPT (5, 30) FAREN.
CALCULOS.
        COMPUTE CENTI = (FAREN – 32) * 5 / 9.
        DISPLAY (7,5) FAREN “Fahrenheit son: “ CENTI  “Centigrados      ”.
FIN.
        STOP RUN.
```

Luego de codificar un programa en COBOL, este debe ser **compilado,** es decir, traducido a lenguaje de máquina.

F**OXPRO. -** Si el Cobol fue popular en las décadas de los 70’s y 80’s, se podría decir que es el FoxPro, el lenguaje comercial más usado a principio de los **90’s.** Mucho **más simple** que sus competidores, el FoxPro es un lenguaje orientado al manejo de grandes cantidades de información organizada en forma de tablas, denominadas **Bases de Datos.** Así, con el FoxPro, podemos rápidamente acceder a un determinado dato de un gran archivo de información general. El FoxPro maneja además **ventanas, menú, botones,** y un gran número de herramientas que lo hacen muy potente a la hora de codificar programas complejos y largos.

En la actualidad, una gran cantidad de aplicaciones comerciales, tanto en almacenes, farmacias, librerías, y tiendas en general, están codificadas en FoxPro. Nuestro programa de conversión Fahrenheit a centígrados, ahora en versión FoxPro 2.6:



VISUAL FOXPRO. - Versión de FoxPro para ambiente visual, es decir, un ambiente en donde primero se diseñan los controles, los objetos y luego se escriben las instrucciones o código. Visual FoxPro 6.0 es un programa con el que se organiza y administra la información mediante vistas, aplicaciones, tablas, consultas, pantallas e informes. Su lenguaje está enfocado a la programación orientada a objeto y permite, a través de asistentes, crear aplicaciones con un alto grado de flexibilidad.

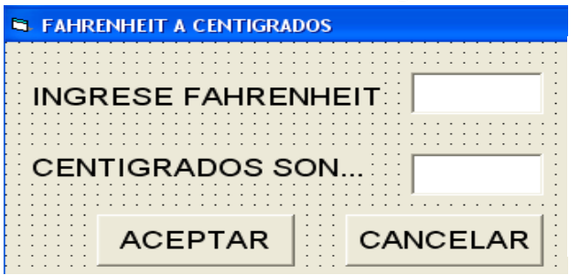
Objecto: Faren **Proc: Valid**
Thisform.centi.value = (Val (Thisform.faren.value) – 32) * 5 / 9

Objecto: Op1 **Proc: Clic**
Thisform.faren.centi.value =" "
Thisform.centi.value =" "
Thisform.faren.Setfocus

Objecto: Op2 **Proc: Clic**
Thisform.Release

VISUAL BASIC. - Aunque originalmente el Visual Basic se diseñó como una versión mejorada del octogenario BASIC de Microsoft, en la actualidad dista mucho de su antecesor. Posiblemente el Visual Basic, sea hoy en día el lenguaje de programación más utilizado para desarrollar aplicaciones comerciales bajo Windows.

Se caracteriza por su programación visual, es decir, bajo un interfaz 100% gráfico. Su fácil manejo populariza rápidamente su aplicación en el mercado de los programadores comerciales. En el mundo de los lenguajes de programación visuales, el programador primero diseña el **GUI** (Interfaz Gráfica del Usuario), es decir, los títulos, mensajes, cajas de ingreso de texto, botones, ventanas y luego escribe el código para cada uno de estos objetos. A continuación, el programa que transforma Fahrenheit a centígrados desarrollado en Visual Basic:




```
Private Sub Command1_Click ()
        If Keyence = 13 Then
           cent = (Faren - 32) * 5 / 9
        End If
        End Sub
```

Lenguaje de Programación General. - Son aquellos que se pueden utilizar para desarrollar una gran variedad de aplicaciones incluyendo aplicaciones pedagógicas o de entretenimiento de las técnicas de programación. Entre los lenguajes generales más representativos tenemos:

LOGO. - Lenguaje de programación de alto nivel que se distingue por su facilidad de uso y sus capacidades gráficas. La sintaxis de Logo es sumamente sencilla para los principiantes.

Logo fue creado por Seymour Papera a mediados de los años setenta. Originalmente se utilizó en computadoras grandes, pero en la actualidad se puede usar en casi cualquier computador personal, y es ideal para que lo **utilicen los niños**, que están aprendiendo a programar. A continuación, el ejemplo que transforma grados Fahrenheit a Centígrados escrito en lenguaje LOGO:

BASIC. - (**B**eginners **A**ll purpose **S**y, bolic **I**nstruction **C**ode) = Código de instrucciones simbólicas de próposito general para principiantes). - Lenguaje de programación desarrollado por **Thomas Kurtz** y **Jhon Kemeny** a mediados de la década de los setenta.

Basic es considerado como uno de los lenguajes de programación **más fáciles de aprender.** Programas simples, pueden ser codificados rápidamente en Basic, al mismo tiempo que se **entrena** al estudiante en la programación de computadoras.

UNIDAD III

Introducción a los Tipos de Programas.

Tipos de Programas.
Programa Fuente.
Programa Objeto.
Pasos para Elaborar un Diagrama.
Algoritmos.
Algoritmos Domésticos.
Algoritmos Lógicos.
Algoritmos Matemáticos.

UNIDAD III

TIPOS DE PROGRAMAS

Existen dos tipos de programas: el Programa Fuente y el Programa Objeto.

Programa Fuente. - Es el conjunto de instrucciones que se encuentran escritos en un **Lenguaje de alto nivel** (Pascal, Cobol, Basic, etc…) o en un lenguaje de bajo nivel (Assembler). El programa fuente es aquel **que entiende el programador,** pero no la computadora.

Programa Objeto. - Es el programa que está escrito en **Lenguaje de máquina,** o sea el código binario. El programa objeto se lo obtiene de la traducción del programa fuente por medio de un **compilador** a un lenguaje que entienda directamente la máquina (código binario). El programa objeto lo **entiende el computador,** pero no el programador.

El software que hace posible la traducción del programa fuente a programa objeto se denomina **COMPILADOR.**

Todos los lenguajes de alto nivel tienen un COMPILADOR**.** Algunos lo tienen **externamente** como es el caso de Cobol, y otros los tienen **internamente** como es el caso del FoxPro o Visual Basic.

Pasos para elaborar un programa

De acuerdo al criterio de cada programador, un programa podría tener una infinidad de pasos. Nosotros los hemos resumido en 7 pasos fundamentales que son:

1. **Entendimiento del problema a resolver.**
2. **Estructuración de un Algoritmo.**
3. **Elaboración de un diagrama de flujo.**
4. **Prueba de Escritorio.**
5. **Pseudocódigo.**
6. **Codificación.**
7. **Depuración.**

1.- Entendimiento del problema a resolver. Aunque parezca demasiado obvio, este primer paso es muy importante. Los programadores principiantes cometen el ***grave error*** de intentar resolver un problema de programación **sin *entenderlo*** al 100%. Por ejemplo, si le piden elaborar un programa que calcule la fórmula de la trasmigración de las almas a través del aura del infinito (¿?). Ud. Solo responderá que no, porque no entiende el problema planteado (ni siquiera con qué se come). Sin embargo, si le piden calcular la hipotenusa de un triángulo rectángulo es muy probable que Ud. Si pueda resolverlo, ya que solo necesita la fórmula C2 = A2 + B2; donde A y B son los catetos mayor y menor.

Moraleja: nunca intente comenzar a escribir un programa sino entiende cabalmente el problema planteado.

2.- Estructuración de un algoritmo. Un algoritmo es un ***conjunto de pasos*** ordenados y lógicos que se llevan a cabo para resolver un problema. Un algoritmo establece el **mecanismo** mediante el cual se va a resolver el problema. Un algoritmo es el "como" resolvemos el problema. En el algoritmo definimos cuáles serán los datos de entrada, los cálculos, condiciones o procesos y los datos de salida del programa.

Por ejemplo, si se nos pide un programa que transforme euros a dólares, necesitamos conocer: cuantos euros vamos a cambiar *(dato de entrada)*, cuál es

la cotización del euro *(calculo)* y finalmente obtener el resultado en dólares *(datos de salida)*. Este es un ejemplo de uno de los tipos de algoritmos conocidos.
Existen tres tipos de algoritmos: MATEMÁTICOS, LÓGICOS y DOMÉSTICOS. Más adelante estudiaremos en detalle a cada uno de ellos.

3.- Elaboración de un diagrama de flujo. El diagrama de flujo o Flujograma como también se lo conoce, es la **representación gráfica** de la solución al problema planteado. El diagrama muestra el flujo de datos desde su entrada, pasando por los cálculos, procesos y condiciones, hasta llegar a la salida u obtención de los resultados. De ahí su nombre Flujograma, el Flujo gráfico de los datos.

4.- Prueba de escritorio. Una vez terminado el Flujograma, es necesario verificar los resultados. A esto se lo conoce con el nombre de prueba de escritorio, es decir, darle **valores ficticios** al diagrama de flujo con el fin de probar la veracidad de sus resultados. En una prueba de escritorio se colocan todas las variables que intervienen en el flujograma y se les dan **valores iniciales** a las variables de entrada. Luego se simulan los cálculos y demás procesos y se verifican los resultados de salida.

5.- Pseudocódigo. El Pseudocódigo es una **versión abreviada,** de fácil entendimiento y en **Lenguaje común y corriente** del programa final. En un

Pseudocódigo se desglosa el programa sin importar el lenguaje de programación que se vaya a utilizar. Un Pseudocódigo consta de palabras en español describiendo las acciones que se deben efectuar para resolver el problema. Se utilizan palabras tales como: lea, ingrese, haga, calcule, repetir, si, mientras, entre otras.

6.- Codificación. La Codificación es la trascripción del flujograma y Pseudocódigo utilizando un Lenguaje de Programación de alto nivel. Cuando un programador codifica un programa, debe ir interpretado el diagrama de flujo y el Pseudocódigo a instrucciones

7.- Depuración. - Una vez que el programa ha sido codificado, siempre será necesario hacerle ciertos ajustes o modificaciones. Este proceso de retro-alimentación se lo denomina Depuración. Es en este paso en donde podemos detectar algún tipo de error no necesariamente de sintaxis, sino de orden de los procesos.

Algoritmos

La forma correcta de elaborar un algoritmo es enumerando uno de los pasos a seguir.

Algoritmos Domésticos. - Son aquellos que se utilizan para resolver los problemas de la **vida cotidiana.** Usualmente no nos sentamos a escribir un algoritmo antes de resolver un problema doméstico. Sólo lo hacemos. Ejemplo: Imagine que usted va viajando en su vehículo tranquilamente vía a Salinas. De pronto, una llanta parece ponchada. Se detiene, examina la llanta y comprueba su sospecha. La llanta está tubo abajo. Escriba el algoritmo para cambiar la llanta y seguir el viaje hacia las hermosas playas ecuatorianas.

1. Sacar la llave de cruz.
2. Aflojar las tuercas de la llanta.
3. Sacar la gata.
4. Colocar la gata debajo del auto.
5. Accionar la gata para que el auto se eleve.
6. Sacar las tuercas.
7. Sacar la llanta ponchada.
8. Bajar la llanta de emergencia.
9. Colocar la llanta de emergencia.
10. Colocar las tuercas.
11. Bajar la gata.
12. Ajustar las tuercas de la llanta.
13. Guardar las herramientas (gata y llave).
14. Guardar la llanta ponchada.
15. Continuar el viaje.
16. Fin.

Este es el algoritmo correcto para cambiar la llanta. Nótese que pudieron ser más de 16 pasos si hubiéramos sido mucho más detallistas, como, por ejemplo, ajustar la **1ª. tuerca, ajustar la 2ª. Tuerca, etc...** Sin embargo, el número de pasos depende del criterio del programador. Recuerde, un buen algoritmo debe ser sobre todas las cosas **lógico y ordenado.** Tenemos a continuación el siguiente algoritmo que sirve para salir al cine con una chica.

1: Revisar la cartelera cinematográfica.
2: Escoger la película que se desea ver.
3: Seleccionar el cine y la hora.
4: Llamar por teléfono a la chica.
5: Invitarla al cine a ver la película escogida.
6: Quedar a la hora que la pasa recogiendo.
7: Bañarse.
8: Cambiarse.
9: Asegurarse de llevar el dinero suficiente.
10: Salir de la casa.
11: Recoger a la chica.
12: Ir al cine.
13: Disfrutar la película.
14: Fin.

Claro, mucha gente no estará de acuerdo con este algoritmo. ¡Algunos están pensando que en el paso 11 (recoger a la chica) faltó saludarla y tal vez en el paso 7 (bañarse) falta sacarse la ropa, ponerse shampoo o que se yo! Insistimos, si bien es cierto que el algoritmo debe detallar los pasos a seguir para resolver un problema, tampoco debe exagerar en sub - problemas (bañarse, cambiarse) que no resuelven el problema principal.

Por otro lado, el problema también se puede complicar. Suponga que la chica no este de acuerdo en la película que Ud. ha preferido ver. Mientras Ud. quiere ver Rocky VI, ella prefiere ver *La Venganza del Peluquero (¿?)*. En este caso el algoritmo se complica. ¡Pero Ud. es el que decide, si acepta ver la película del peluquero en contra de su voluntad o va al cine solo (mejor que mal acompañado)!

El siguiente algoritmo permite servir una taza de café para el desayuno:

1. Hervir suficiente agua.
2. Llenar una taza de agua hervida.
3. Colocar una cucharadita de café.
4. Colocar dos cucharadas de azúcar.
5. Revolver bien.
6. Tomar café.
7. Fin.

Este algoritmo puede variar si a Ud. le gusta el café más cargado o con menos azúcar.

Para el siguiente algoritmo solicitamos ayuda a una señorita. Mientras la hermana mayor se maquilla para ir a una elegante fiesta, su hermana menor la observaba para luego tratar de imitarla. Sin embargo, la joven notó una gran cantidad de pasos. Entonces decidió escribirlos para no olvidarlos:

1. Quitar residuos de maquillaje anterior (con crema).
2. Lavarse la cara.
3. Secarse la cara.
4. Aplicar la base o polvo compacto.
5. Aplicar el blush.
6. Delinear las cejas.
7. Aplicar la sombra.
8. Delinear los párpados.
9. Rizar las pestañas.
10. Aplicar el rime.
11. Delinear los labios.
12. Fin.

Como podrá apreciar, en cada momento de nuestras vidas utilizamos algoritmos para la resolución de problemas domésticos. A continuación, otro tipo de algoritmos.

Algoritmos lógicos. - Son aquellos que para su resolución necesitamos la ayuda de algún **artificio lógico** y de razonamiento pausado y calculado del problema. Por Ej.:

Suponga que tenemos tres copas A, B, C. En la copa A tenemos café; en la B tenemos vino; la copa C está vacía. Queremos intercambiar los contenidos de las copas A y B, es decir café en B y vino en A

A **B** **C**

1. Poner A en C.
2. Poner B en A.
3. Poner C en B.
4. Fin.

¡Muy sencillo…! Veamos otro ejemplo:

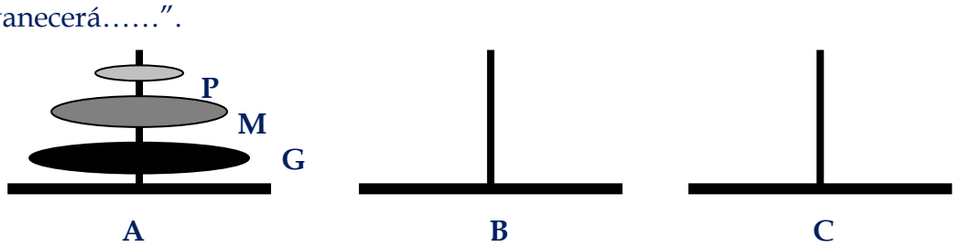
En la cumbre de una montaña se encuentran tres alpinistas. Dos de ellos pesan 60 Kg. cada uno, mientras que el tercero pesa 120 kg. Desean pasar a la cumbre de la otra montaña, y para ello disponen de un transportador manual que solo soporta un peso máximo de 120kg. Escriba un algoritmo que especifique las secuencias de pasos a seguir para que los tres alpinistas pasen a la otra montaña.

1. Pasan los dos delgados.
2. Regresa un delgado.
3. Pasa el gordo.
4. Regresa el otro delgado.
5. Pasan los dos delgados.
5. Fin.

Obviamente el alpinista delgado (60kg) tiene que regresar dos veces debido al que el transportador requiere de alguien que lo conduzca.

Este tipo de Algoritmo es utilizado inclusive en la resolución de problemas de tipo religioso.

En el texto sagrado de la religión indoasiática Brahamanistas, se encuentra el siguiente relato que habla acerca de la creación del mundo:
"… En el gran templo Benarés, bajo el domo que marca el centro del mundo, yace una placa de latón a la que están fijadas tres agujas de diamantes de un codo de alto cada una, y del grosor del cuerpo de una abeja. En torno de estas agujas, Dios, en el acto de la creación colocó 64 discos de oro puro, descansando el mayor sobre la placa de latón y decreciendo progresivamente los demás conforme se asciende por la pila. Es la torre de Brahma. Día y noche sin cesar, los sacerdotes del templo van transfiriendo los discos de una aguja de diamante a otra, de acuerdo con las leyes fijas e inmutables del Brahma, que exigen que el oficiante no deba mover más de un disco a la vez y deba situar tal disco en otra aguja sin que debajo quede el disco más pequeño. Cuando los 64 discos se hallan transferido de la aguja que Dios los colocó en la creación a una de las otras, la torre, el templo y los Brahamanes quedarán reducidos a polvo y, con un inmenso estruendo, el mundo se desvanecerá……".



Dejando de lado la veracidad de este relato religioso, a continuación, un algoritmo que describe la secuencia de pasos para trasladar 3 aros (no 64 como en la historia) marcados como "**G"** (grande), "**M"** (mediano) y "**P"** (pequeño) desde la torre **"A"** a cualquiera de las torres **"B" o "C"**. Las reglas son: se debe de trasladar un sólo aro a la vez, y en ningún momento se debe colocar un aro más grande por encima de otro más pequeño. (Se debe respetar el orden de tamaño.)

1. Pasar el aro P a B.
2. Pasar M a C.
3. Pasar G a B.
4. Pasar M a A.
5. Pasar de G en C.
6. Pasar de M a C.
7. Pasar P en C.
8. Fin.

Un hombre deseaba transportar un lobo, una gallina, y un saco de maíz de una orilla a otra. Dispone de una canoa que sólo resiste su peso (el del hombre) y de otro más (lobo, gallina o maíz), es decir, sólo dos pesos. Elabore un algoritmo que indique las secuencias de pasos a seguir para que el hombre transporte al lobo, gallina y maíz, sin que en ninguna de las dos orillas queden solos el lobo y la gallina (porque se la come) ni la gallina sola con el maíz (porque también se lo come).
Suponiendo que es el hombre el que conduce la canoa:

1. Pasa con la gallina.
2. Pasa el maíz, pero se regresa con la gallina.
3. Deja la gallina y pasa al lobo.
4. Pasa con la gallina.
5. Fin.

Se tiene dos recipientes: uno para capacidad de 5lts. y el otro con capacidad de 4lts. Se desea tener exactamente 3litros en el recipiente de 4lts, sabiendo que disponemos de abundante agua, y los recipientes no tienen marcación alguna.

1. Se llena el recipiente de 4lts.
2. Se vierte este contenido en el de 5lts.
3. Se llena nuevamente el recipiente de 4lts.
4. Se vierte el contenido en el recipiente de 4lts, en el de 5lts hasta que este último se llene, como le faltaba sólo un litro. En el recipiente de 4lts, exactamente quedan 3lts.
5. Fin.

Algoritmos Matemáticos. - Son utilizados en la solución de problemas ***aritméticos*** y que tiene que ver con la aplicación de una fórmula matemática.

Un algoritmo matemático requiere saber cuáles son los datos con que contamos, cuál va a ser la ***fórmula empleada*** y el resultado que deseamos obtener.

El siguiente algoritmo matemático define los pasos para obtener el promedio trimestral de un alumno:

1. Obtener el 1er aporte.
2. Obtener el 2do aporte.
3. Obtener el 3er aporte.
4. Saber la calificación del examen.
5. Sumar todas las notas.
6. Promediar para 4.
5. Fin.

Se desea calcular la hipotenusa de un triángulo rectángulo. Recuerde que la formula es:

$$H=\sqrt{a^2+b^2}$$

El siguiente es el algoritmo:

1. Saber cuánto vale el cateto mayor.
2. ¿Cuánto vale el cateto menor??
3. Elevar el cuadrado del cateto mayor.
4. Elevar el cuadrado del cateto menor.
5. Sumar estos dos resultados.
6. Extraerle la raíz cuadrada.
5. Fin.

El siguiente algoritmo que establece el salario a percibir por un empleado que trabaja por horas:

1. Determinar cuantas Horas ha trabajado el empleado.
2. Saber cuánto gana por hora.
3. Multiplicar las horas trabajadas por el sueldo de horas.
5. Fin.

Tenemos una infinidad de ejemplos de algoritmos matemáticos, pero por su similitud con los pseudocódigos los trataremos más adelante.

UNIDAD IV

Diagramas de Flujo.

Diagramas de Flujo, definición.
Simbología.
Reglas para la Diagramación.
Las Variables y Constantes
Operadores Aritméticos.
Reglas de Prioridad.
Fórmulas.
Flujogramas de Procesos Simples.
Ejercicios de Aplicación.
Función SQRT.

UNIDAD IV
DIAGRAMAS DE FLUJO

Un diagrama de flujo es la resolución ***gráfica*** de un problema de programación. El diagrama de flujo o flujograma consta de una serie de **símbolos** especiales los cuales representan cada uno de los pasos o acciones a seguir para resolver el programa.

Entre las principales ***ventajas*** de los diagramas de flujo tenemos:

- Permite tener una ***visión*** mucho más ***amplia*** y ***objetiva*** del programa a codificar.

- Un mismo diagrama de flujo puede ser luego codificado indistintamente a ***cualquier lenguaje*** **de programación de alto nivel.**

- Es mucho más ***fácil y rápido codificar*** un programa teniendo como base el diagrama de flujo.

- Los diagramas de flujo sirven como una excelente ***documentación*** a la hora de hacer ***modificaciones*** en el programa. Es más ***práctico*** efectuar primero las modificaciones en el diagrama que en la codificación.

- La descripción paso a paso de todas las instrucciones del programa resulta de máxima utilidad para el programador, ya que es el fiel reflejo del programa. Es de gran ayuda para ***estructurar*** su propia documentación de referencia.

Dicho de otra manera, el **diagrama de flujo** es tan importante para el programador como un ***plano*** de un edificio para un constructor. Ud. cree que un Ingeniero Civil podría edificar una casa sin tener previamente un plano de la construcción?

Bueno, un programador no puede (ni debe) escribir la codificación de un programa sin haber diseñado antes el ***diagrama de flujo*****.**

SIMBOLOGÍA

A continuación, los ***principales*** símbolos utilizados en los diagramas de flujo:

Indica el ***inicio*** o ***fin*** de un diagrama. Todo diagrama por extenso o corto que sea, debe ***empezar*** y ***terminar*** con este símbolo.

Especifica que se van a ***ingresar datos*** por medio del teclado. Se utiliza con palabras tales como lea, ingrese, introduzca

Se utiliza para **visualizar** o **presentar** todo tipo de datos por pantalla.

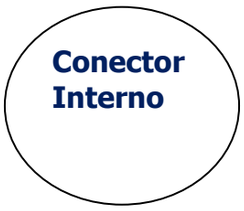
Utilice este símbolo cuando se emplea alguna **fórmula** para resolver un proceso o calculo

Una bifurcación es una **condición** o **pregunta** que puede tener solo dos opciones de respuestas: verdadero o falso.

El símbolo de **ciclo** o **bucle** se usa cuando se desea **repetir** un determinado número de veces una parte del diagrama de flujo

Un procedimiento es un subprograma al que se invoca o llama desde el programa principal

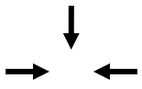
Símbolo usado para representar la **salida** o presentación de datos por **impresora.**



Cuando un diagrama es demasiado largo se utiliza el conector interno para especificar que el diagrama **continúa** en otra parte de la **misma hoja**.

Cuando un diagrama requiere continuar en **otra hoja**, se utiliza el conector externo.

Este es un símbolo que se puede usar indistintamente para representar un **ingreso** o una **salida de datos**.



Líneas de Flujo

Permite **unir** o enlazar los símbolos del diagrama entre sí

Reglas para la diagramación

1.- Los diagramas se dibujan de ***arriba hacia abajo*** y de ***izquierda a derecha*.**

2.- Los símbolos van siempre ***interconectados*** por medio de las líneas de flujo.

3.- Las líneas de flujo deben ser ***rectas*** y no pueden cruzarse.

4.- Cuando un diagrama no alcance en una página se deben ***usar conectores,*** ya sea internos (dentro de la misma hoja) o externos (en otra hoja).

5.- Para visualizar, presentar o imprimir mensajes estos deben estar entre comillas.

Variables
Una variable es un ***casillero de memoria*** en el cual podemos almacenar datos.
Estos datos pueden variar dependiendo de la ejecución del programa.

Toda variable debe tener un nombre. Los nombres de las variables deben cumplir con las siguientes reglas:

- ☑ Deben de empezar siempre con una ***letra***, (Mayúscula o Minúscula) y pueden ser una ***combinación*** de ***letras*** y ***números***. Por ejemplo, nombres correctos de variables son: SUMA, XX, M1, M2, XYZ123, SUELDO, etc…

- ☑ **No** pueden tener ***espacios en blanco,*** ni símbolos especiales intermedios. Para nombres de variables largos, use abreviaturas (sin puntos). Por ejemplo: incorrecto es usar SUELDO POR HORA, correcto sería SUELXHOR.

- ☑ Deben ser nombres ***cortos*** **y** significativos. Evite usar nombres largos y difíciles. Por ejemplo, en lugar de usar la variable PROMEDIO use PROM.

- ☑ Cada nombre de variable debe ser ***único***, es decir, no puede existir otra variable con el mismo nombre.

Operadores Aritméticos

Los operadores aritméticos permiten efectuar ***procesos*** o cálculos ***matemáticos elementales*****.** Los operadores aritméticos son:

OPERADOR	OPERACIÓN	EJEMPLO
+	Suma	5 + 2 = 7
-	Resta	5 – 2 = 3
*	Multiplicación	5 * 2 = 10
/	División	5 / 2 = 2.5
Mod	Residuo de división	5 mod 2 = 1

En el caso de los operadores ***"/"*** y ***Mod,*** son exclusivamente para la división. El operador "/ "es usado para la división decimal, es decir, la división común y corriente. Por ejemplo:

6 / 3 = 2
7 / 2 = 3.5
6 / 4 = 1.5
4 / 3 = 1.33333

El operador ***Mod*** obtiene el residuo, es decir, lo que sobra de una división. Por ejemplo:

14	mod	3 = 2	(porque 14 / 3 es 4 sobrando 2)
6	mod	4 = 2	(divida 6 para 4 y le sobran 2)
9	mod	2 = 1	(sobra 1 al dividir 9 para 2)
10	mod	5 = 0	(divida 10 para 5 y no le sobra nada)

Adicionalmente existe la función ***INT,*** la cual permite ***extraer la parte entera*** de una expresión decimal, es decir ***desecha los decimales*** **(**si los hubiere**).** Por ejemplo:

INT (13 / 3) = 4 (13/3 es 4.33, pero INT solo toma en cuenta la parte entera)
INT (7 / 2) = 3
INT (6 / 4) = 1

Reglas de prioridad

Dentro de una expresión matemática compleja, siempre se efectúan **primero** los **paréntesis** más internos, luego las divisiones y las multiplicaciones y al final se realizan las sumas y restas.

Por ejemplo: 5 + 4 / 2 – 1
Primero se efectúa 4 / 2, o sea 2 y luego 5 + 2 - 1. El resultado es 6.

( )	**Máxima Prioridad**
/, Mod, *	**Prioridad Intermedia**
+, -	**Prioridad Mínima**

Por ejemplo:

5 + 2 * 4 - 3

es 10; primero se efectúa 2*4 y luego se suma y se resta. Sin embargo, los ***paréntesis*** siempre se evalúan ***primeros***.

Por ejemplo:

(5 + 2) * 4 - 3

es 25; primero se hace el paréntesis (5+2) y luego se multiplica por 4 y se resta 3.

Veamos ahora los siguientes ejemplos:

A) 2 + 1 * 5 + 2
Es 9, porque primero es **1 * 5 = 5**; luego **2 + 5 + 2 = 9**

B) (5 Mod 2) + 2 / 2
Primero es **5 Mod 2 = 1** y **2 / 2 = 1**; entonces **1 + 1 = 2**

C) INT (11 / 3) – (7 Mod 2)
INT (11 / 3) es **3 y 7 Mod 2** es **1**; entonces **3 – 1 = 2**

D) 5 + 2+3 / 2
Primero **3 / 2 = 1.5**; más **5 + 2 + 1.5 = 8.5**

Fórmulas

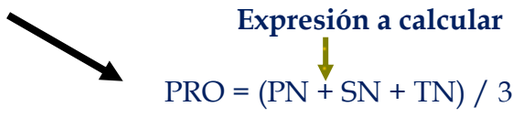
Una fórmula es una expresión que se utiliza para ***obtener*** un resultado. Por ejemplo:

S = A + B

Es la fórmula que representa la suma de dos valores, donde la variable **"A"** representa el primer valor, la variable **"B"** el segundo valor y la **"S"** la suma.

Las fórmulas siempre se evalúan de ***derecha*** a ***izquierda***, es decir, colocando la expresión a calcular al lado derecho del igual y la variable que guarda el resultado del lado izquierdo.

La siguiente fórmula representa el promedio de tres calificaciones de un alumno:



Veamos las siguientes fórmulas:

SALARIO = SXH * HT

PROM = (PP + SP + TT) /3

P = A + B + C

HORA = M * 60

Sin embargo, no todas las fórmulas llevan exclusivamente variables. En el caso de conversiones o transformaciones se utilizan valores ***constantes.***

Una ***constante*** es un valor preestablecido, es decir, que no cambia durante la ejecución del programa.

Ejemplos de valores constantes son:

- El valor de Pi = 3.141592654
- Los Minutos que tiene una hora = 60
- Metros que tiene un Kilómetro = 1000
- Libras que tiene un kilo = 2.2; etc.

- La siguiente fórmula:

M = H * 60

Convierte una determinada cantidad de **Horas (H)** a **Minutos (M)**

- Para transformar Kilos a Libras:

LB = KG * 2.2

Se multiplica lo que deseamos transformar KG por el valor ***constante*** 2.2 y se lo almacena en LB.

- Para convertir metros a Kilómetros:

KM = MT / 1000

Porque por cada 1000 mts. hay 1 KM

Flujogramas de procesos simples

Ahora vamos a realizar nuestro ***primer diagrama*** de flujo. Son diagramas de procesos simples aquellos en los que se ingresan una o más variables, luego se aplica alguna fórmula y finalmente se visualiza o imprime el resultado. Lea detenidamente el enunciado del problema:

Elabore un diagrama de Flujo que lea dos valores. **Calcule y Visualice** su suma.

Bien. Analicemos el enunciado. Primero ya sabemos que todo diagrama debe empezar con el símbolo de **Inicio (fig1)**. Luego observe como se pide que ***lea*** dos valores.
Entonces necesitamos dos variables; por ejemplo, A y B y las colocamos dentro símbolo de **ingreso (fig2)**. Después se pide que se **calcule** su suma. Entonces aplicamos la fórmula S = A+B y la ponemos dentro del símbolo de **proceso (fig3)**.

Al final **visualiza** su suma. Entonces colocamos la variable **S** que representa la suma, dentro del símbolo de **visualizar (fig4)**. Para terminar, colocamos el símbolo de ***fin*** **(fig5)** y unimos todos los símbolos con líneas de flujo y tenemos listo nuestro diagrama.

Figura 1

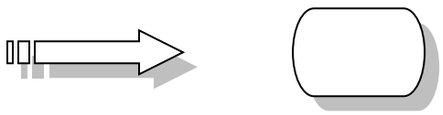


Figura 2

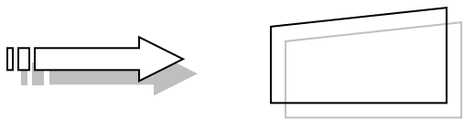


Figura 3

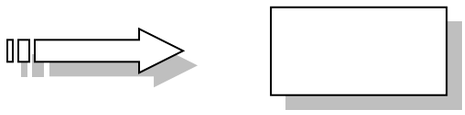


Figura 4

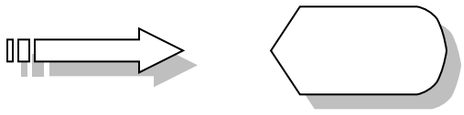
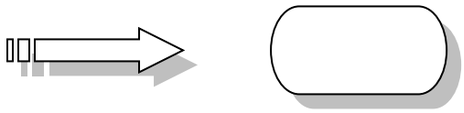


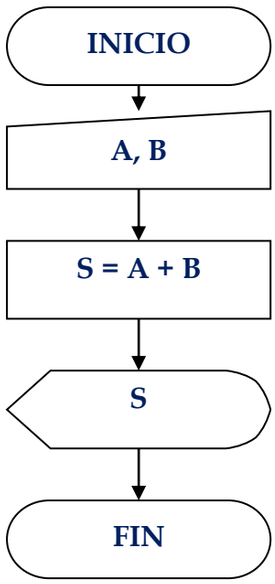
Figura 5



EJERCICIOS DE APLICACIÓN

Ejercicio de Aplicación 01

Elabore un diagrama de Flujo que lea dos valores. **Calcule y visualice** su suma.



Las **variables** usadas van a representar los dos valores ingresados. "**A**" representa al ***primer valor*** y "**B**" representa al ***segundo valor.*** Luego, mediante la formula **S=A+B** calculamos la suma y se almacena en la variable "**S**". Siempre el resultado de todo cálculo debe almacenarse en alguna variable de memoria, Luego presentamos el resultado de la suma **S** mediante el símbolo de visualizar.
Para comprobar que el diagrama de flujo esté bien hecho, se le realiza una prueba de escritorio.

La prueba de escritorio consiste en darle valores ficticios o de prueba a todas las variables de entrada, es decir en este caso **A** y **B** luego se simulan los cálculos y se obtienen los resultados.

PRUEBA DE ESCRITORIO

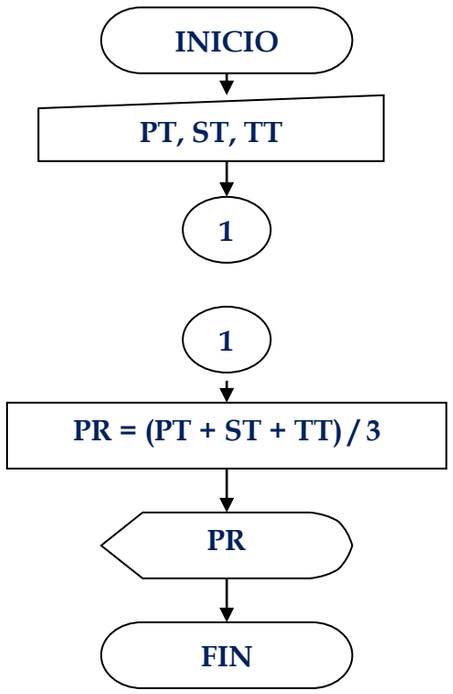
A	B	S
3	5	8
10	2	12

Los valores de 3, 5, 10 y 2 son valores ficticios, usted puede colocar sus propios valores.

o

Ejercicio de Aplicación 02

Elabore un diagrama que ingrese las calificaciones del primer, segundo y tercer trimestre de un estudiante. Calcule y visualice su promedio.



Como son tres notas, usamos 3 variables. **PT** primer trimestre, **ST** segundo trimestre, **TT** tercer trimestre. ***Luego note el uso del símbolo conector interno, el cual indica que el diagrama continuo dentro del conector se coloca un número de orden.***

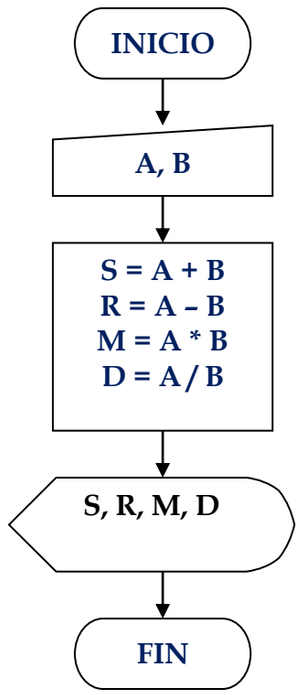
Como es el primer conector, entonces ponemos1. Luego aplicamos la fórmula en donde primero efectuamos la suma de las tres notas y después dividimos para 3.

Finalmente presentamos el promedio obtenido **PR**. La prueba de escritorio es:

PT	ST	TT	PR
13	16	18	15.6
19	20	16	18.3

Ejercicio de Aplicación 03

Lea dos números. Visualice los resultados de sus 4 operaciones fundamentales:



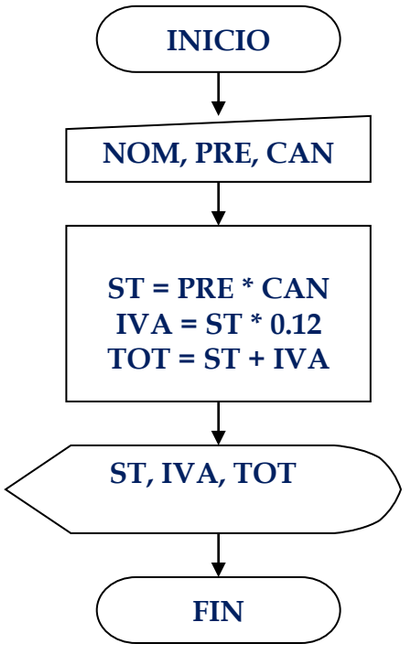
Para los dos números ingresados usaremos las variables **A** y **B**. Luego almacenamos el resultado individual de cada operación en una variable distinta; así **S** para la suma, **R** para la resta, **M** para la multiplicación y **D** para la división.

Finalmente presentamos los resultados, es decir S, **R, M, D.** Veamos la prueba de escritorio:

A	B	S	R	M	D
6	3	9	3	18	2
21	7	28	14	147	3

Ejercicio de Aplicación 04

Ingrese el nombre, el precio y la cantidad de un producto a comprar. Calcule y visualice el subtotal, el 12% IVA y el total a pagar.



CÁLCULO DE PORCENTAJE.
Un porcentaje es la **proporción** de una cantidad con relación a otra que se calcula sobre la centena. De esta manera:

Porcentaje = tanto * monto / 100

Donde **tanto** es el tanto por ciento a aplicar; **monto** la cantidad a la que se le extrae el tanto y el **porcentaje** el resultado obtenido. De esta manera veamos el 5% de 2000:

Porcentaje = 5 * 2000 / 100
Porcentaje = 100

En nuestros diagramas de flujo usaremos variables. Entonces, si monto es ***M*** y queremos abreviar la expresión, podemos dividir el tanto para 100 recorriendo el punto decimal dos lugares a la izquierda.

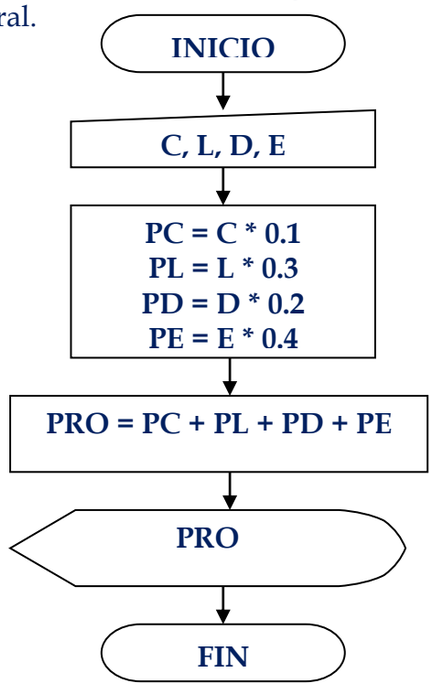
Usaremos las variables **NOM** para el nombre del producto, **PRE** para el precio y **CAN** para la cantidad. El cálculo del subtotal se obtiene multiplicando el precio por la cantidad. El 12% IVA se obtiene multiplicando el subtotal por **0.12;** el total a pagar es la suma del subtotal más el IVA. Luego presentamos todos los resultados obtenidos.

A continuación, la prueba de escritorio:

NOM	PRE	CAN	ST	IVA	TOT
Teclado	15	2	30	3.6	33.6
Mouse	9	5	45	5.4	50.4

Ejercicio de Aplicación 05

En cierto colegio, el promedio trimestral se calcula de la siguiente manera: el 10% de la nota es el cuaderno al día, el 30% las lecciones, el 20% los deberes y el 40% el examen. Si todas las calificaciones son sobre 20, elabore un diagrama que ingrese las notas de cuaderno, lección, deberes y exámenes de un estudiante. Visualice el promedio trimestral.



Ingresemos las variables **C** cuaderno, **L** lecciones, **D** deberes, **E** exámenes. Estas notas son sobre 20 y hay que extraerle el porcentaje para obtener el promedio trimestral. Entonces multiplicamos cada nota por su respectivo porcentaje. **PC** porcentaje de cuaderno, **PL** porcentaje de lección, **PD** porcentaje de deberes y **PE** porcentaje de exámenes. Luego sumamos los resultados obtenidos y presentamos el promedio.
Aquí la prueba de escritorio:

C	L	D	E	**PC**	**PL**	**PD**	**PE**	**PRO**
18	15	12	14	1.8	4.5	2.4	5.6	14.3

Aprecie que si en el **cuaderno** sacó 18 su equivalente porcentual es 1.8 (10%). En **lecciones** obtuvo 15 pero su equivalente es 4.5 (30%); en **deberes** sacó 12, y su equivalente es 2.4 (20%); en **exámenes** sacó 14 y su equivalente es 5.6 (40%). Todo sumado es la nota trimestral del alumno: 14.3 (un poco despreocupado).

UNIDAD V

BIFURCACIONES.

Bifurcaciones, definición.
Tipos de Bifurcaciones.
Bifurcaciones Simples.
Ejercicios de Aplicación.
Bifurcaciones Dobles.
Ejercicios de Aplicación.
Bifurcaciones Múltiples.
Ejercicios de Aplicación.
Bifurcaciones Compuestas
Operador Lógico AND.
Ejercicios de Aplicación.
Operador Lógico OR.
Ejercicios de Aplicación.

UNIDAD V

Bifurcaciones, definición

Una Bifurcación es una pregunta con tan solo dos respuestas posibles: **Verdadero** o **Falso**. Las Bifurcaciones se construyen a partir de las condiciones. Una **condición** es una expresión lógica de forma:

Variable OPERADOR ***RELACIONAL*** Variable/valor

Los Operadores Relaciónales son:

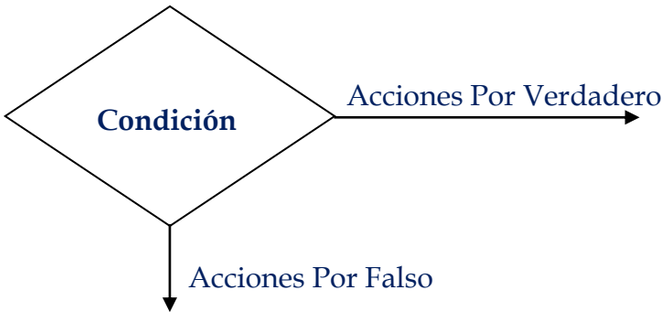
OPERADORES RELACIONALES	SIGNIFICADO	EJEMPLO
>	Mayor que	5 > 2
<	Menor que	3 < 6
=	Igual	1 = 1
>=	Mayor o igual	4 >= 3
<=	Menor o igual	5 <= 10
< >	Diferente a	2 < > 7

De esta manera, utilizamos los operadores lógicos y podemos construir las siguientes condiciones, suponiendo que: **A = 1;** **B = 2** **y** **C = 0;**

a) **A > B**
b) **C < A**
c) **C = 0**
d) **A >= 1**
e) **B <= 0**
f) **A < > C**

El **literal A)** es falso, porque **"A "**que vale 1 no es mayor que **"B"** (que vale 2).
El **Literal B)** es verdadero porque **"C"** vale 0 y A vale 1.
El **Literal C)** es verdadero porque "C" = 0.
El **Literal D)** es verdadero porque **"A"** que vale 1 si es mayor o igual que 1.
El **Literal E) es** falso porque **"B"** no es menor o igual que 0.
Por último, el **Literal F)** es verdadero porque **"A"** si es diferente que **"C"**.

En un diagrama de flujo las condiciones se representan mediante el símbolo de la Bifurcación, indicando claramente cuáles serán las dos alternativas de Verdadero o Falso.

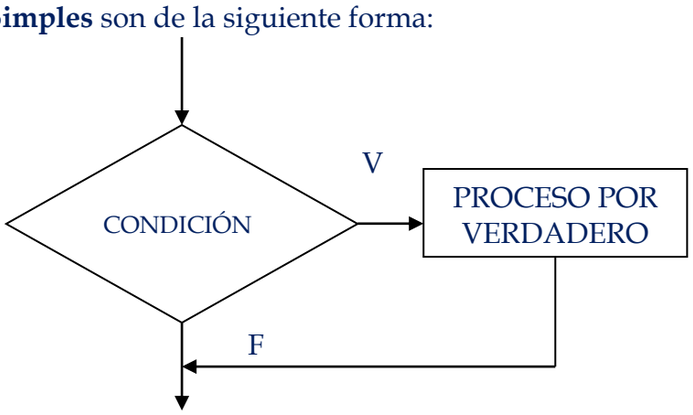


Los diferentes tipos de Bifurcaciones son:
Bifurcaciones Simples.
Bifurcaciones Dobles.
Bifurcaciones Múltiples.
Bifurcaciones Compuestas.

Bifurcaciones Simples

Toda Bifurcación tiene dos alternativas: verdadero y falso. Sin embargo, las bifurcaciones simples sólo realizan acciones cuando la Condición es Verdadera, cuando la Condición es Falsa no se realiza ninguna acción, es decir, el diagrama continúa su flujo normal.

Las **Bifurcaciones Simples** son de la siguiente forma:

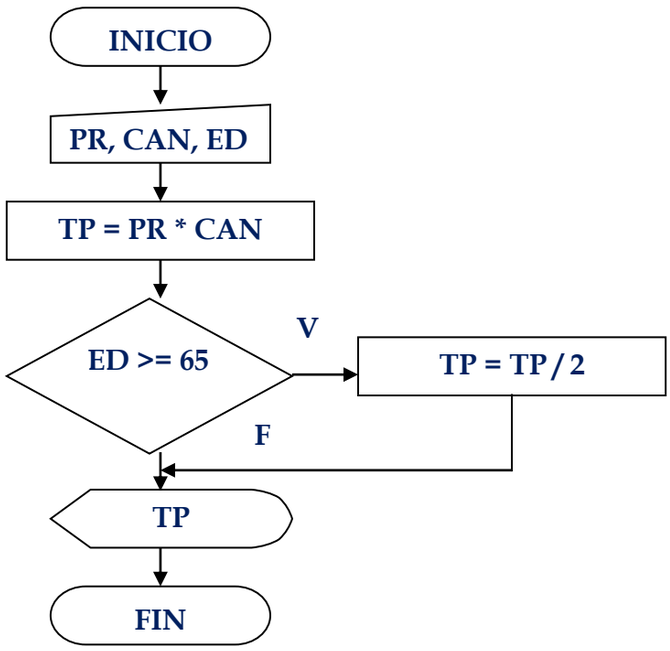


Nota: cuando la condición es ***verdadera***, se realiza un proceso, sin embargo, cuando la condición es ***falsa***, no se realiza nada.

EJERCICIOS DE APLICACIÓN

Ejercicio de Aplicación 01

Elabore un diagrama que lea el precio individual y el número de entradas al cine a comprar. Adicionalmente ingrese la edad de la persona. Calcule y visualice el total a pagar. Considere que si la persona es de la 3ra. edad (65 años o más) debe pagar sólo la mitad de todo.



Se ingresa el precio y la cantidad de entradas a comprar. También se ingresa la edad. Se calcula el total a pagar multiplicando el precio por la cantidad. Luego, es necesario descontarle la mitad si es una persona de la 3ra. Edad. Entonces si **ED >=65** (años) a la misma variable **TOT** la dividimos 2, de esta manera se le rebaja la mitad. **Vea** como después de efectuarse este proceso, el flujo se dirige hacia la alternativa de falso. **Es decir, esta Bifurcación realiza un proceso por verdadero y por falso no hace nada**. Continua el diagrama, al final se visualiza el total a pagar.

Para la prueba de escritorio se deben colocar valores que prueben las dos alternativas. Por ejemplo:

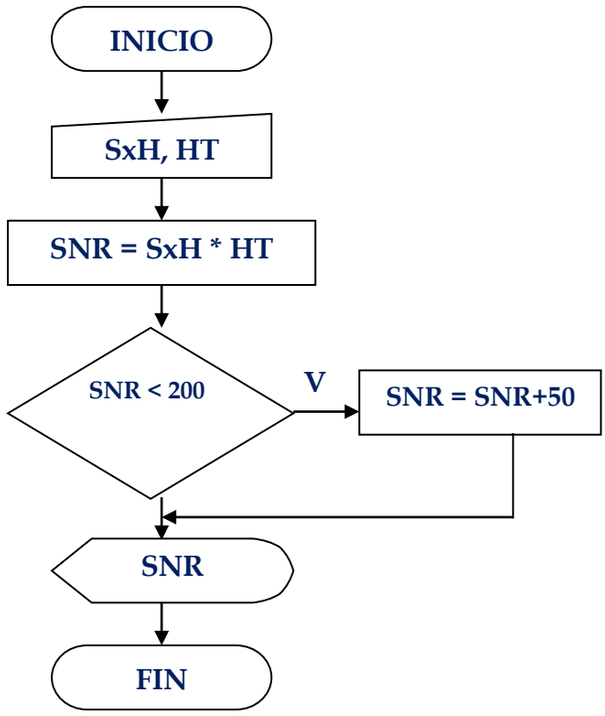
PR	CAN	ED	TP
6	2	26	12
6	3	68	9

Para el primer caso, se compran 2 entradas a $6.00 cada una**. Sin embargo, la edad del comprador es de 26 años.** Entonces el resultado de la bifurcación es falso. Se presenta el total directamente. **Para el segundo caso, se compran 3 entradas a $6.00 cada una, pero el comprador tiene 68 años**. Entonces se le divide el total para 2. Luego se visualiza el total a pagar.

Ejercicio de Aplicación 02

Ingrese el sueldo por hora y las horas trabajadas por un empleado. Sólo si el salario neto a recibir por el empleado es menor a $200.00 páguesele por concepto de transporte $50.00 adicional. Visualice el salario a recibir.

Luego de ingresar el sueldo por hora y las horas trabajadas se efectúa el cálculo del salario neto. Si dicho salario es **< 200** se le suma al salario 50. Luego se visualiza presenta **SNR**.
Cuando SxH es 1.5 y HT vale 40, el salario neto a recibir es 60; entonces la condición es verdadera. **Por eso se le suma 50 y el resultado final es 110**.
Cuando el sueldo por hora es 10 y las horas trabajadas son 40, el salario neto es 400. Por lo tanto, no recibe bono de transporte porque no es menor que 200

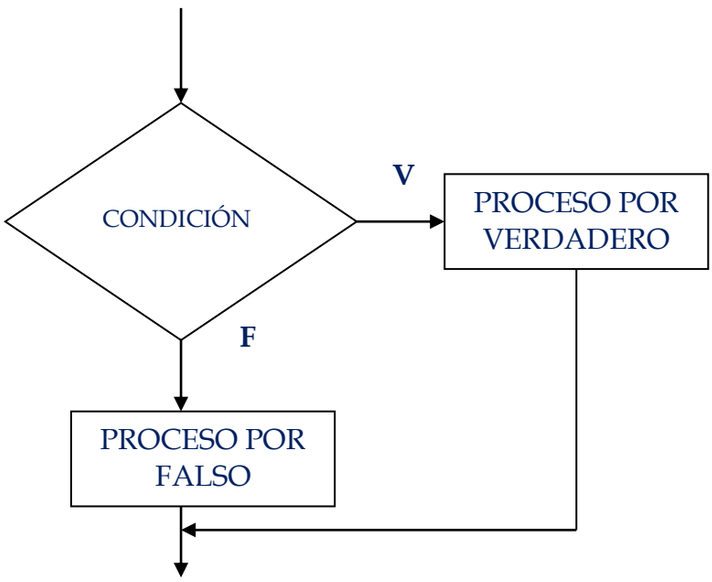


SxH	HT	TRANS	SNR
1.5	40	50	110
10	40	0	400

Entonces la condición **SNR < 200** es falsa. No se efectúa ningún proceso. Al final, ya sea por verdadero o por falso indistintamente siempre se visualiza el salario neto a recibir **SNR**

BIFURCACIONES DOBLES

Una bifurcación es doble cuando tanto por verdadero como por falso se realiza la ejecución de alguna acción o proceso. Las Bifurcaciones dobles son de la siguiente forma:

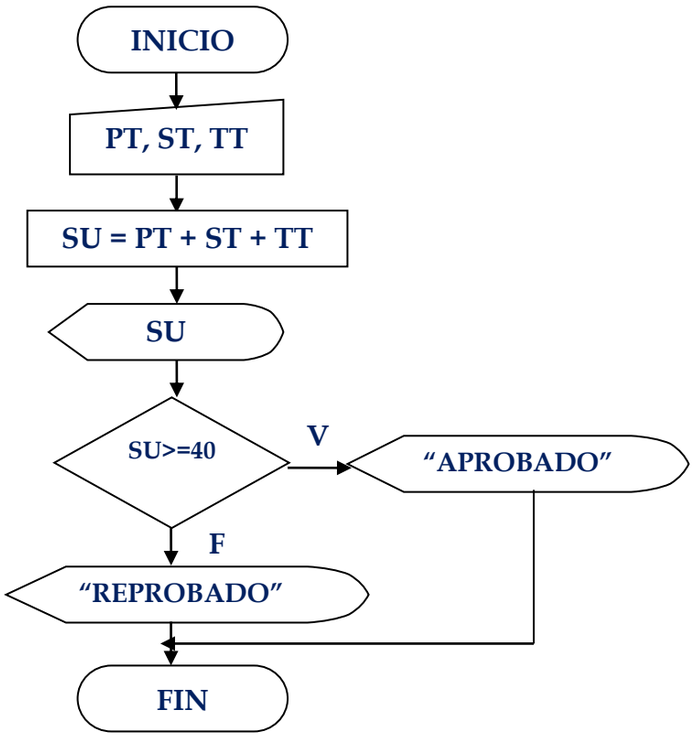


De esta manera, se dice que una bifurcación es doble cuando se realizan acciones tanto por la alternativa de verdadero como por la alternativa de falso.

EJERCICIOS DE APLICACIÓN

Ejercicio de Aplicación 03

Ingrese las notas de los 3 trimestres de un alumno. Visualice su suma. Si dicha suma es mayor o igual a 40 visualice el mensaje "Aprobado". Si la suma es menor a 40, visualice el mensaje "Reprobado".



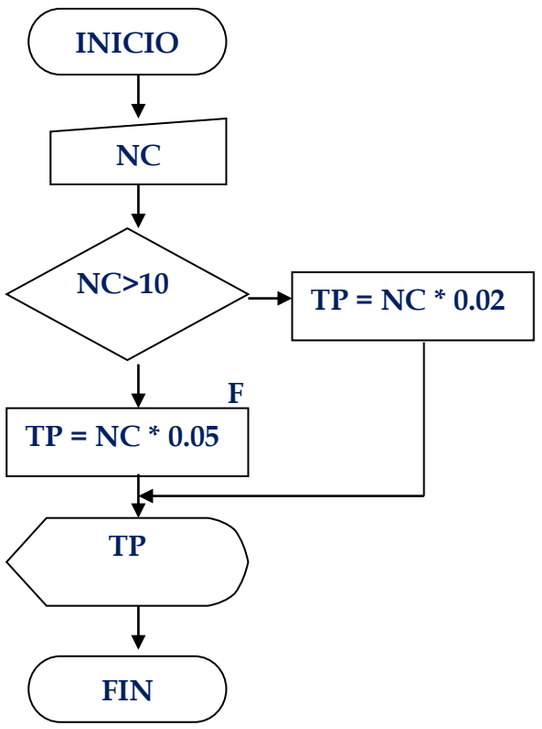
Para este diagrama leemos las variables equivalentes a los tres trimestres: **PT, ST, TT**.
Luego las sumamos y presentamos su suma Si la suma es mayor o igual a **SU>=40** se presenta el mensaje "APROBADO"; caso contrario, si la condición es falsa, es decir, la suma no es mayor o igual a 40, entonces se presenta el mensaje "REPROBADO"

PT	ST	TT	SU	MENSAJE
15	12	16	43	APROBADO
11	10	13	34	REPROBADO

El enunciado del ejemplo dice claramente "*… **si dicha suma es mayor o igual a 40 visualice el mensaje APROBADO; si la suma es menor a 40, el mensaje REPROBADO".*** Nótese cómo la condición **SU >= 40**, satisface el texto del enunciado. Si la suma es mayor o igual a 40 se presenta el **mensaje APROBADO**. Luego dice, si la suma es menor a 40. Alguien puede pensar que hace falta otra condición: **SU<40**.
Está condición no es necesaria porque su **SU>=40 es falsa**, entonces **SU** es menor que **40**. Está sobreentendido.

Ejercicio de Aplicación 04

Unas copias se sacan a $ 0.05 cts. cada una si son 10 o menos. Si se sacan más de 10, entonces se cobra $ 0.02 cts. por cada fotocopia. Elabore un diagrama que lea el número de fotocopias a sacar y visualice el total a pagar.



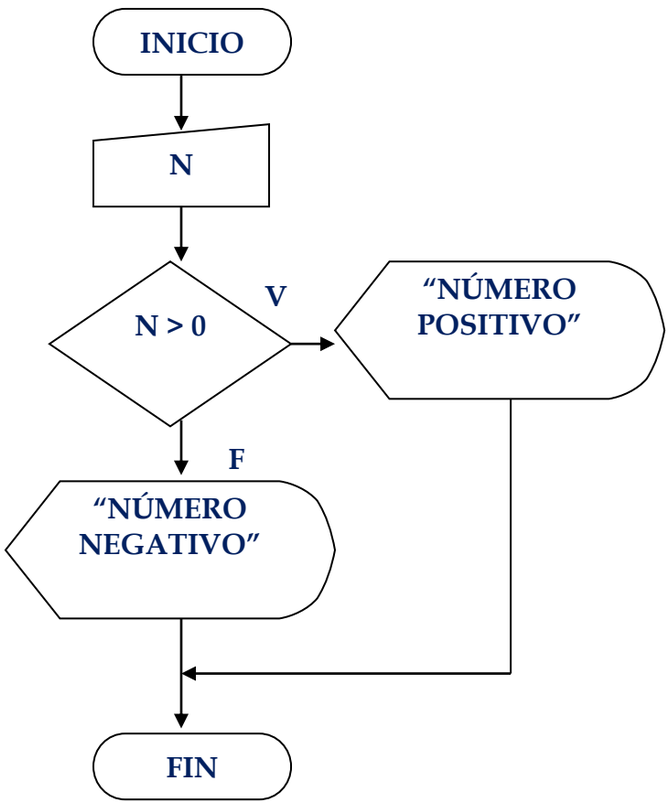
NC	TP
8	0.40 cts.
20	0.40 cts.

Si NC es 8, la condición es falsa. Si NC es 20 la condición es verdadera.

Se ingresa la variable **NC** número de copias. Antes de realizar el cálculo debemos efectuar la bifurcación, porque no sabemos cuánto van a costar las copias; **si 0.05 cts.** o **0.02 cts.** cada una. Si la condición **NC>=10** es verdadero, entonces se multiplica el número de copias **NC por 0.02**. Si la condición es falsa, se multiplica el número de copias por **0.05**. Al final se presenta el total a pagar.

Ejercicio de Aplicación 05

Se ingresa por teclado un número. Visualice el mensaje "Número Positivo" o "Número Negativo" según corresponda.

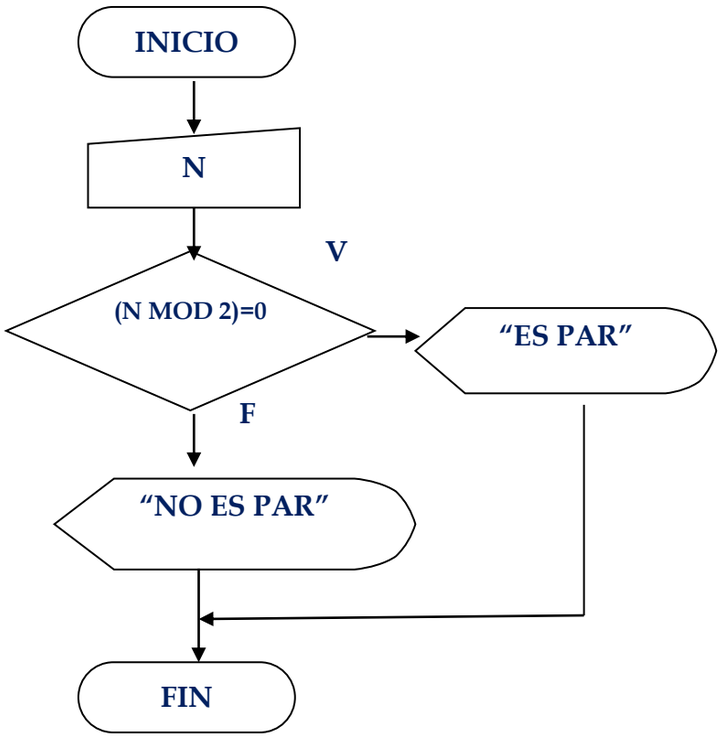


N	MENSAJE
5	POSITIVO
-3	*NEGATIVO*

Todo número mayor que cero es positivo. Si el número es **menor que cero, es negativo**. Entonces se ingresa el número el **N** y se aplica la condición **N>0**; si dicha condición es verdadera el número es **positivo**; caso contrario es **negativo**.

Ejercicio de Aplicación 06

Efectué un diagrama que lea un número. Visualice si es "PAR" o "IMPAR".

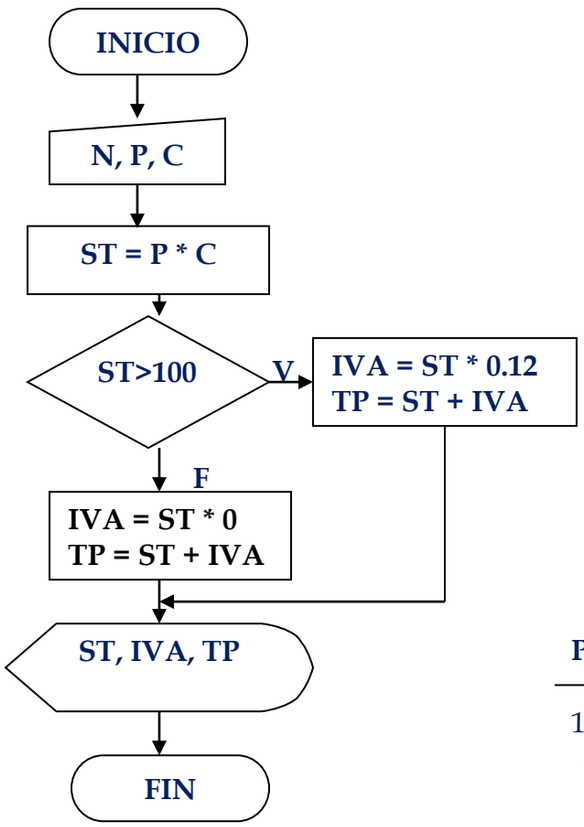


N	MENSAJE
12	ES PAR
33	*ES IMPAR*

Al dividir un número para 2 siempre sobra 1 o 0. Si sobra **0 entonces** el número es exactamente *divisible para 2*, es **decir es par**. Si sobra **1, entonces el número es impar.**

Ejercicio de Aplicación 07

Realice un diagrama que ingrese el nombre de un producto, precio y la cantidad. Si el subtotal es mayor a $100,00, calcule el 12% del IVA, caso contrario IVA cero. Visualice el subtotal, Iva y Total a Pagar.



Se ingresa el nombre de un producto, el precio y la cantidad. Luego realizamos el calculo para obtener el subtotal **ST=P*C.** A continuación, realizamos la bifurcación porque no sabemos si tenemos que pagar **IVA o no**. Si la condición es mayor a 100 **ST>100** por verdadera, se multiplica **ST * 0.12** para el **IVA**; y el **TP= ST+IVA.** Si la condición es falsa, se multiplica **ST * 0** para obtener el IVA y **TP= ST+IVA.** Al final visualizamos en ST, IVA y TP

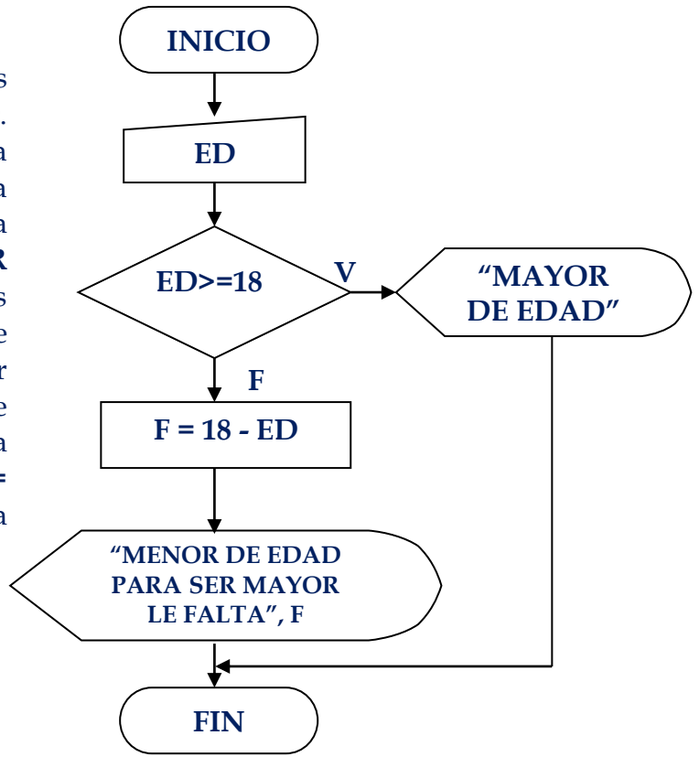
PRE	CAN	ST	IVA	TP
120	5	600	72	672
40	2	80	0	80

Cuando, por ejemplo, el precio es 120 y la cantidad es 5, el subtotal es 600 y la condición **ST>100** es verdadera. Entonces, en ese caso, se aplica el 12% IVA.

Ejercicio de Aplicación 09

Elabore un diagrama que ingrese la edad de una persona. Visualice el mensaje "Mayor de Edad" o "Menor de Edad"; según corresponda. En el caso de ser menor de edad, visualice cuantos años le faltan para ser mayor de edad.

En el Ecuador, una persona es mayor de edad al cumplir 18 años. Así ingresamos la edad en la variable **ED**. Efectuamos la condición **ED>=18**; si es verdadera se presenta el mensaje **"MAYOR DE EDAD"**; si es falsa, entonces primero se efectúa el cálculo de cuantos años le faltan para ser mayor de edad. Esto se lo hace restando 18 **menos** edad de la persona. **Si tiene 15 años, 18 – 15 = 3 (años),** que es lo que le falta para ser mayor de edad.



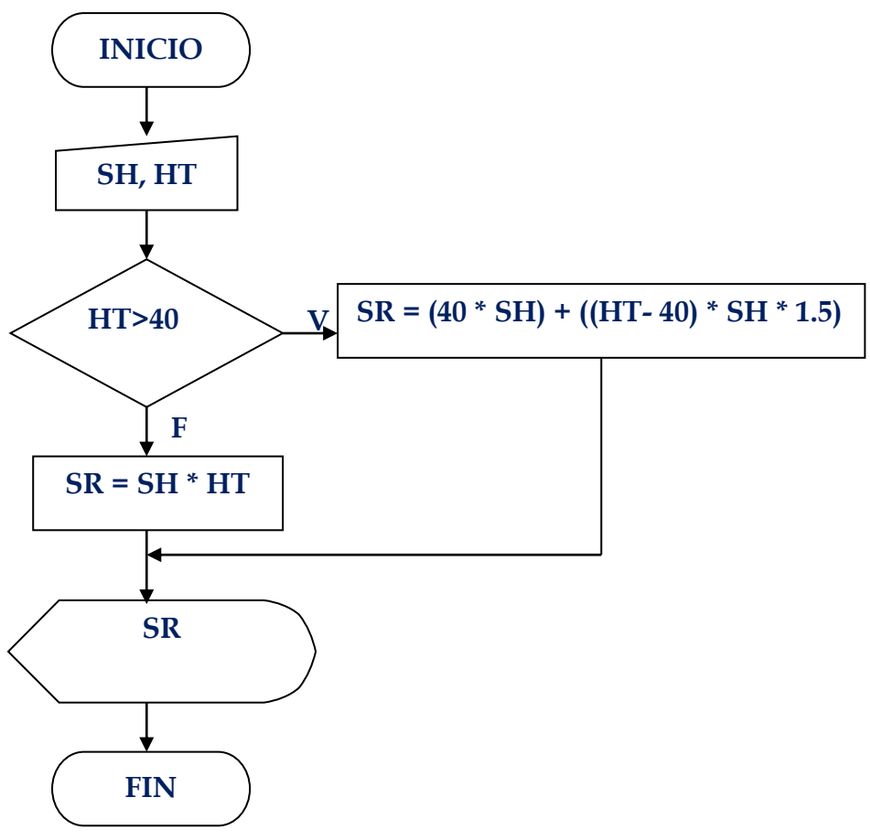
Nótese que para **presentar el mensaje se usaron comillas**. Sin embargo, la variable **F** debe ir fuera de las comillas, separada por una coma. Para presentar los mensajes y variables en un solo símbolo, debe poner, los mensajes entre comillas y las variables con su propio nombre separadas del mensaje por una coma.

ED	F	MENSAJE
19		MAYOR DE EDAD
15	3	MENOR DE EDAD PARA SER MAYOR LE FALTAN 3 AÑOS

Ejercicio de Aplicación 10

Lea el sueldo por hora y las horas que trabaja un empleado. El empleado gana su sueldo por hora normal hasta las 40 horas trabajadas. Por cada hora adicional a las 40, el empleado gana el 50% más del sueldo por hora normal. Efectué los cálculos necesarios y visualicé el total a pagar.

Por ejemplo, si un empleado trabaja 45 horas, con un sueldo de $2.00 por cada hora trabajada; por las primeras 40 horas el empleado gana ***40 * 2.00 = 80***. Por cada hora adicional a las 40, tiene 5 horas de sobretiempo, multiplicando por el ***50%*** más de su sueldo por hora normal, es decir, si gana 2.00 por hora normal y por cada hora de sobretiempo ganará 3.00, así ***5 * 3.00 = 15***, que sumados a los 80 por horas de trabajo normales da un total a recibir de ***$95.00cts***



Explicaremos el cálculo para poder obtener el salario a recibir si las horas trabajadas son mayores a 40. Primero sabemos que si se ha trabajado más de 40 horas, por estas primeras 40 horas se pagará su salario normal, es decir, ***(40 * SH).*** Luego, para saber las horas sobretiempo, restamos **HT** de **40: (HT – 40).** Este número de horas se multiplica por el salario normal aumentado en un 50%. Esto ha sido abreviado multiplicando el sueldo por hora por 1.5, que es la expresión después de facturar:

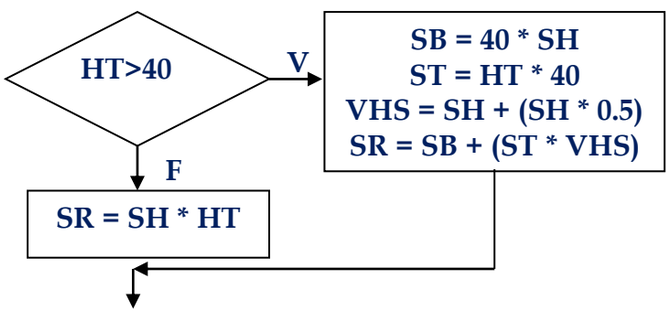
SH * (SH * 50 / 100)

SH * (SH * 0.5)

SH * (1 + 0.5)

SH * 1.5

Ahora escribimos un segmento del diagrama para detallar de una mejor manera la fórmula en mención:



SB = Sueldo Base
ST = Horas de Sobretiempo
VHS = Valor * hora sobretiempo

En la variable SB almacenamos el sueldo base si las horas trabajadas son mayores a 40, es decir, su sueldo normal. Luego, la variable ST almacena las horas sobretiempo trabajadas. En la variable VHS se almacena el valor a pagar por hora sobretiempo. Observe como se suma SH más el 50% de SH, es decir, **SH + (SH*0.5).** Finalmente, se suman el sueldo base SB y la multiplicación de las horas sobretiempo por el valor de horas sobretiempo. Sin embargo, todas estas formulas se pueden abreviar en una sola:

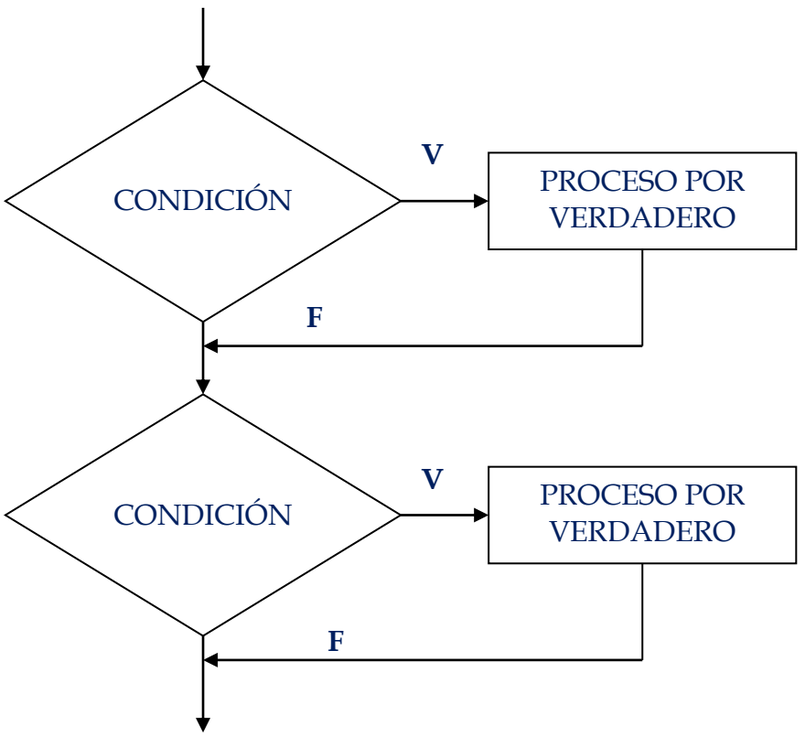
SR = (40 * SH) + (HT – 40) * SH * 1.5

Que es la que usamos en nuestro diagrama original. Siempre será un mejor diagrama, aquel que contenga de forma más concisa y concreta las formulas a calcular.

BIFURCACIONES MÚLTIPLES

Se dice que una Bifurcación es múltiple cuando colocamos varias Bifurcaciones, una a continuación de otra.

Las **Bifurcaciones Múltiples** son de la siguiente forma:



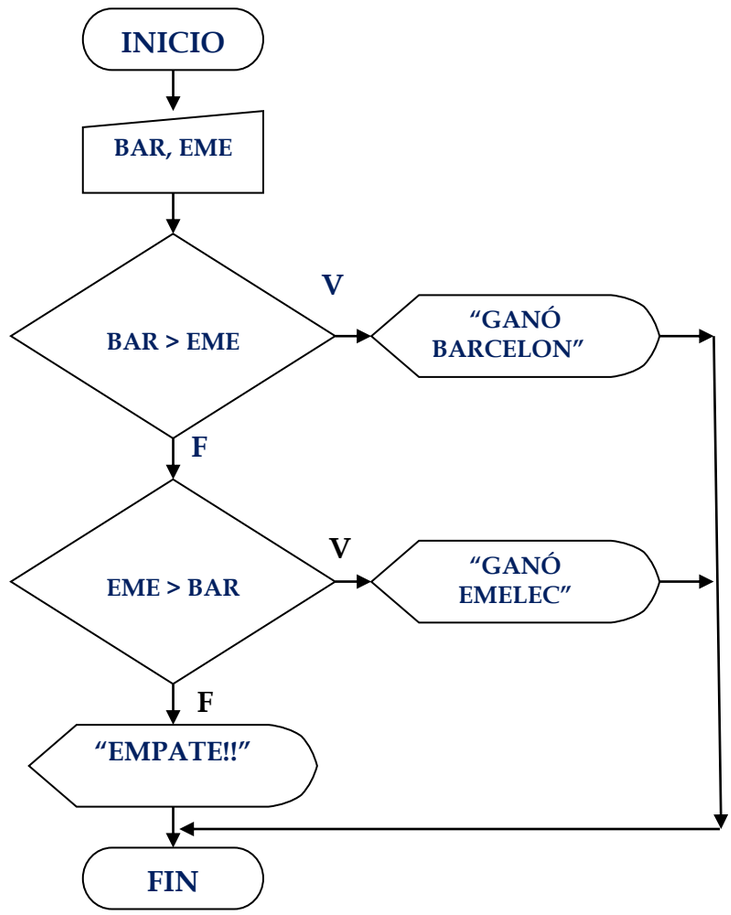
Donde podemos colocar una determinada cantidad de Bifurcaciones, siempre una a continuación de otra.

EJERCICIOS DE APLICACIÓN

Ejercicio de Aplicación 11

Realice un diagrama que lea el marcador de un clásico del astillero, es decir, cuanto gol marco el equipo de BARCELONA y cuantos anoto el equipo del EMELEC. Visualice el mensaje con el nombre del equipo ganador o si es que hubo empate.

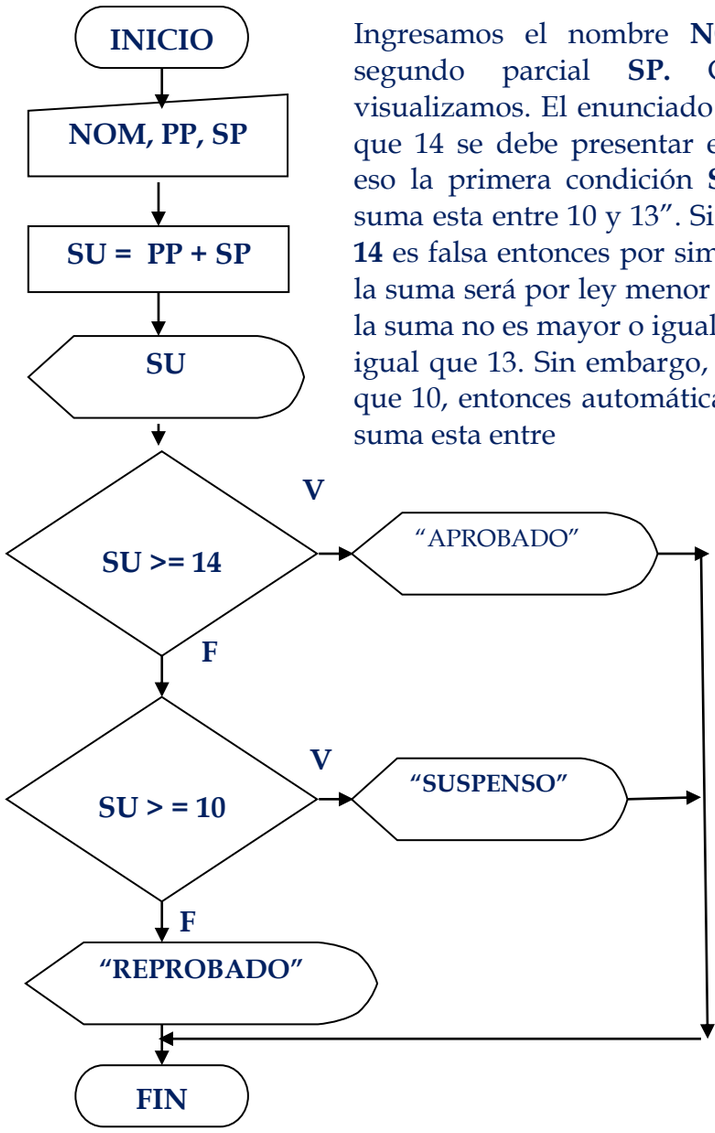
Para este diagrama ingresamos los goles tanto de BARCELONA como de EMELEC. Si la condición **BAR > EME** es verdadera, significa que BARCELONA consiguió un mayor número de goles que EMELEC. Entonces gana BARCELONA. Si la condición es falsa, preguntamos si **EME > BAR**. Si es verdadera gana EMELEC. Sin embargo, si esta nueva condición también es falsa, entonces significa que hubo un empate.
Analice. Si BAR > EME es falsa y EME > BAR también es falso, entonces BAR y EME son iguales.



BAR	EME	MENSAJE
2	1	*GANÓ BARCELONA*
1	3	*GANÓ EMELEC*
0	0	*EMPATE*

Ejercicio de Aplicación 12

Efectué un diagrama que lea el nombre; y las notas del primer y segundo parcial de un estudiante de la universidad: Calcule y visualice su suma. Si dicha suma es mayor o igual a 14, visualice el mensaje "APROBADO"; si la suma esta entre 10 y 13 visualice el mensaje" SUSPENSO"; y si la suma es menor o igual a 9 el mensaje "REPROBADO".

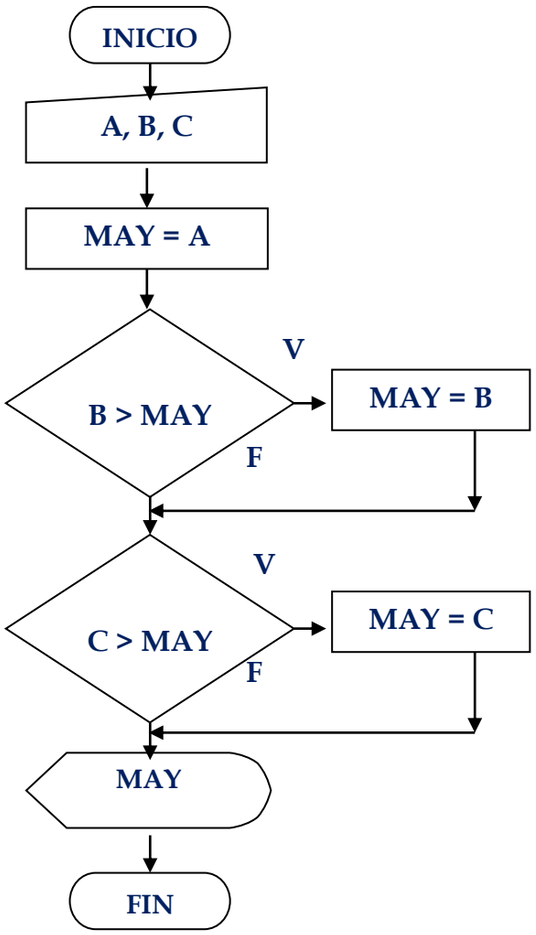


Ingresamos el nombre **NOM**, primer parcial **PP** y segundo parcial **SP.** Calculamos su suma y visualizamos. El enunciado indica que si la suma es **>=** que 14 se debe presentar el mensaje "Aprobado". Por eso la primera condición **SU >= 14**. Luego dice Si la suma esta entre 10 y 13". Si la primera condición **SU >= 14** es falsa entonces por simple lógica se presupone que la suma será por ley menor o igual que 13. Insistimos, si la suma no es mayor o igual que 14, entonces es menos o igual que 13. Sin embargo, si la suma es mayor o igual que 10, entonces automáticamente sabemos que dicha suma esta entre 10 y 13. De esta manera la segunda condición SU < = 10, por verdadero presenta "Suspenso". Finalmente, si la suma no es mayor o igual que 14, ni mayor o igual que 10 indudablemente es menor o igual que 9. Entonces se presenta "Reprobado."

Ejercicio de Aplicación 13

Realice un diagrama de flujo que lea tres números. Visualice al mayor de ellos.

Para facilitar la búsqueda del número mayor entre tres, en este diagrama usamos un pequeño artificio. Ingresamos los tres valores y luego asumimos que el mayor de todos es el primero. De esta manera, almacenamos en la variable **MAY** el valor de la variable **A**. Luego preguntamos si B es mayor que **MAY** (o sea A). Si la condición es verdadera, entonces el nuevo mayor es **B**. Así **MAY = B**. Ahora preguntamos si **C** es mayor a MAY. Ya en ese momento, **MAY** contiene el mayor valor entre **A** y **B**. Si **C > MAY** es verdadero, entonces almacenamos **C** en **MAY**.



Este es un algoritmo muy usado para encontrar el mayor de varios números. Se asume que el primer valor es el mayor, y luego se hacen sucesivas comparaciones con los demás valores. En el momento que uno de estos valores sea menor, se intercambian los valores. Veamos la prueba:

A	B	C	MAY
3	5	9	~~3~~
			~~5~~
			9

Ejercicio de Aplicación 14

Lea el nombre y las cuatro notas trimestrales de un alumno. Calcule y visualice su suma y promedio y su equivalente en letras:

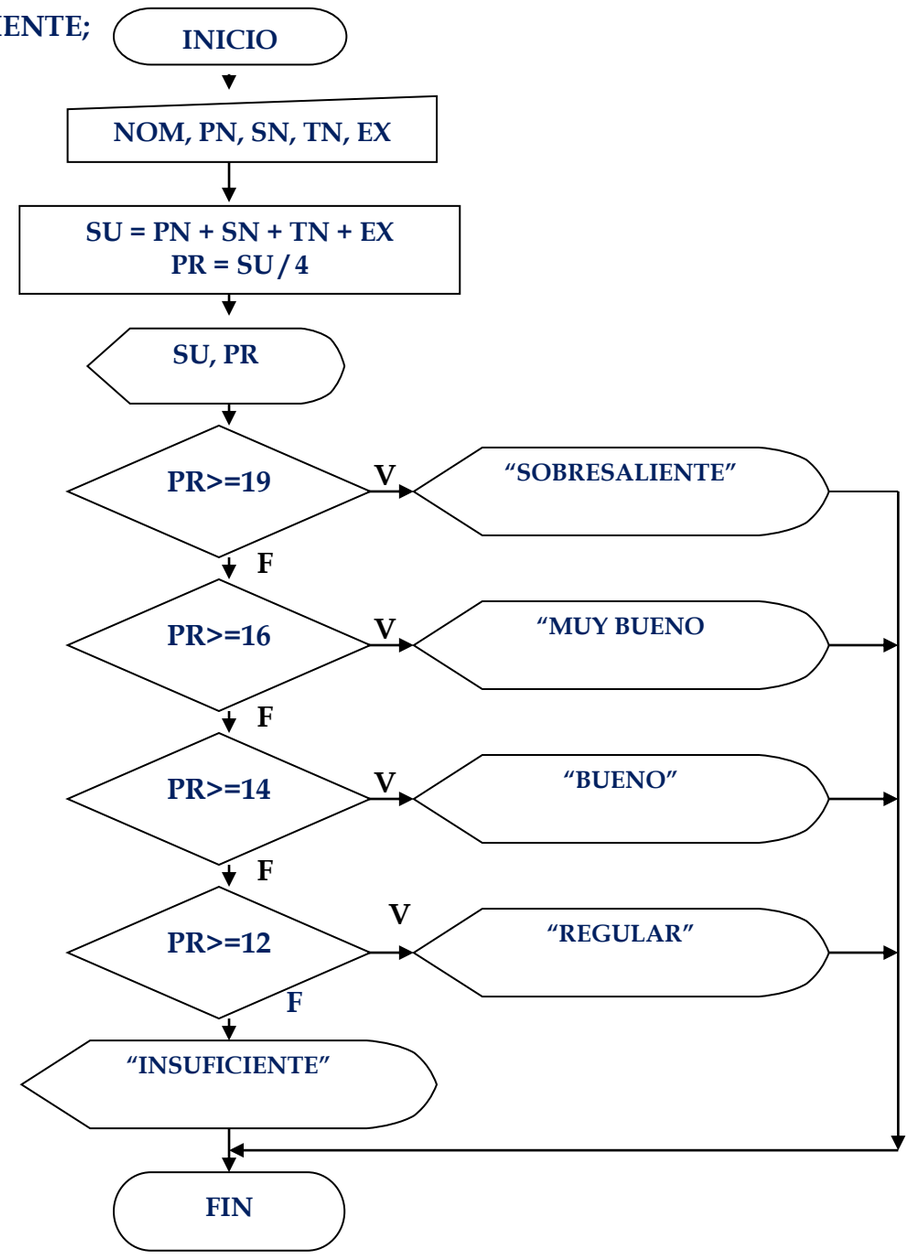
20 – 19 SOBRESALIENTE;

18 – 16 MUY BUENO;

15 – 14 BUENO;

13 – 12 REGULAR;

11 – 00 INSUFICIENTE;



Leemos el nombre del estudiante y las notas de sus tres mensuales y examen. Calculamos suma, promedio y luego las visualizamos. La primera condición **PR>=19** abarca notas de 19 y 20, es decir, "Sobresaliente".

Luego si **PR** no es mayor o igual a 19, entonces se supone que es menor a 19, o sea **18, 17, 16……** Sin embargo, si **PR** es mayor o igual que 16, entonces abarca a **16, 17 y 18**; es decir "Muy Bueno".

Si **PR** no es mayor o igual a 16, entonces es menor que 16. Si **PR** es mayor o igual que 14 (14 – 15), entonces se visualiza "Bueno".

Si **PR >=14** es falso, entonces PR es menor que 14, es decir, 13, 12…. Así **PR>=12** incluye al 12 y 13; o sea "Regular".

Finalmente, si PR no fue mayor o igual que 19, ni mayor o igual que 16, mayor o igual que 14, mayor o igual que 12; entonces definitivamente es una nota "Insuficiente" (menor o igual que 9).

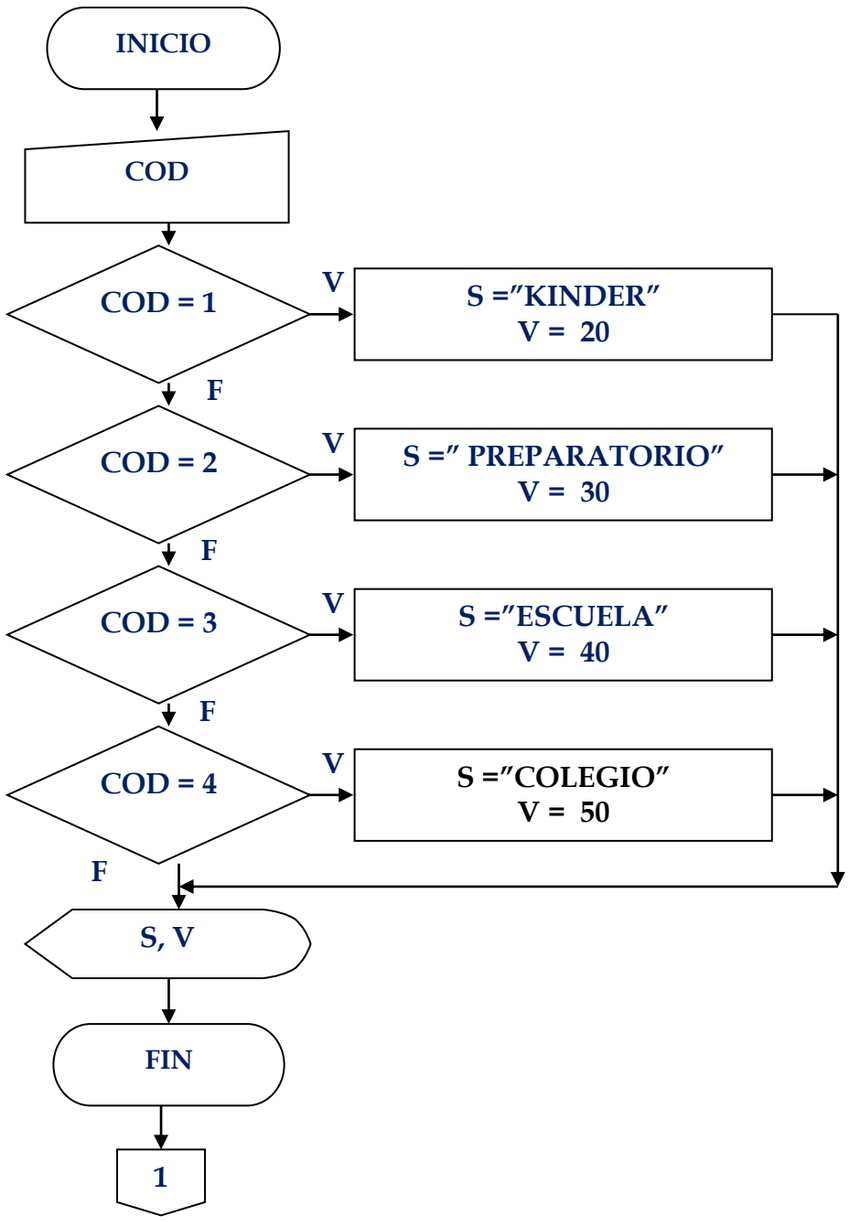
A continuación, la prueba de escritorio:

NOM	PN	SN	TN	EX	SU	PR	MEN
Bryan	18	16	15	19	68	17	MB
Eva	10	06	11	13	40	10	INSUF
Joel	14	16	15	15	60	15	BUENO

Ejercicio de Aplicación 15

Cierta Unidad Educativa tiene 4 secciones: Kinder, Preparatorio, Escuela, Colegio. Se requiere un diagrama de flujo que permita ingresar el código y visualice automáticamente su sección y el valor de la mensualidad de acuerdo a la siguiente tabla:

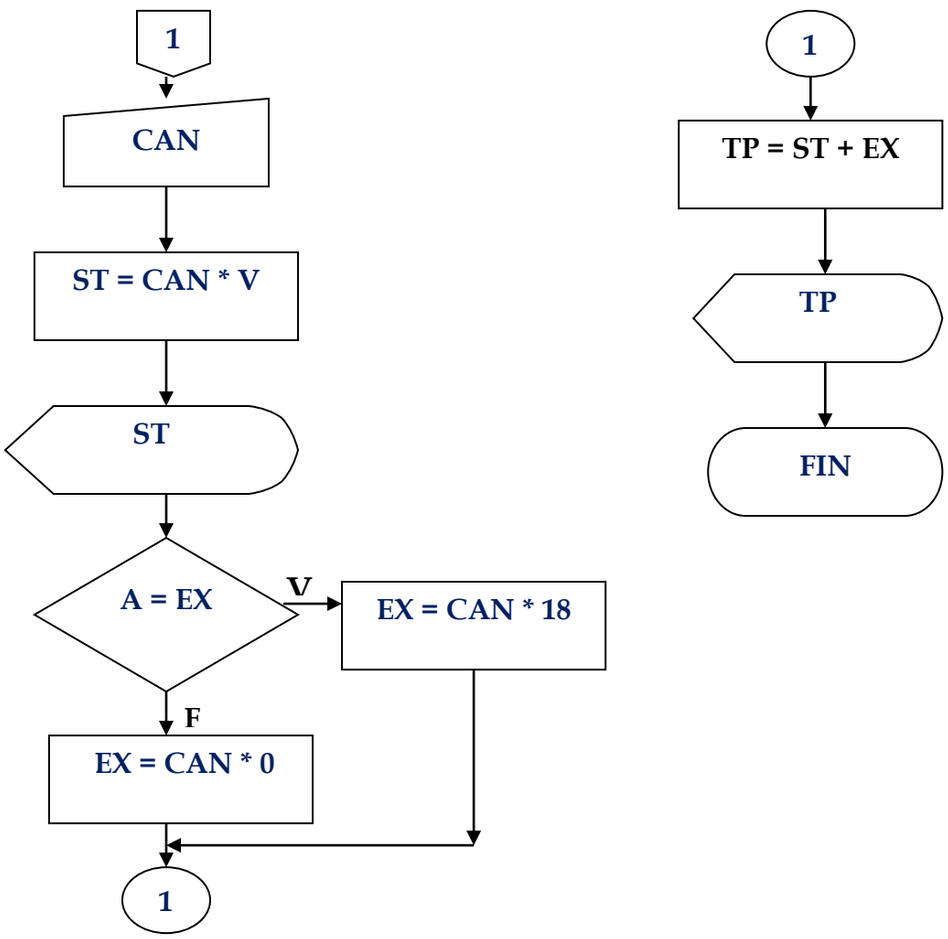
CÓDIGO	SECCIÓN	MENSUALIDAD
1	Kinder	$ 20.00
2	preparatoria	$ 30.00
3	escuela	$ 40.00
4	colegio	$ 50.00



Luego ingresamos el número de mensualidades a pagar. Visualice el subtotal. Pregunte si el alumno tiene expreso. Si la respuesta es afirmativa, agregue $18.00 por cada mes de pago. Visualice el total a pagar.

Este diagrama es un tanto extenso porque es necesario comparar la variable ingresada **COD** para los 4 valores posibles. Así, si COD es igual a 1, entonces almacenamos el mensaje "Kinder" en la variable **S.** Nótese el uso de las comillas para almacenar un mensaje (todo mensaje debe ir entre comillas). También almacenamos el valor mensual a cobrar, en este caso $20. Luego preguntamos si **COD** es igual a 2, 3 o 4. En cada condición asignamos el correspondiente mensaje y valor.

Después, al final de todas las condiciones **COD**, visualizamos las variables **S** y **V**, que para ese momento deben tener almacenada la sección y mensualidad correspondiente al código ingresado. Usamos un conector de página para continuar el diagrama. En esta segunda parte, leemos la cantidad de meses **CAN** por el valor de la mensualidad, que es la variable **V.** Visualizamos el subtotal. Utilizamos la variable EX para preguntar si el alumno tiene expreso. La respuesta debe ser S si o N no. En caso de se S se debe de agregar **$18.00** por concepto de expreso, caso contrario no pagaría expreso.



BIFURCACIONES COMPUESTAS

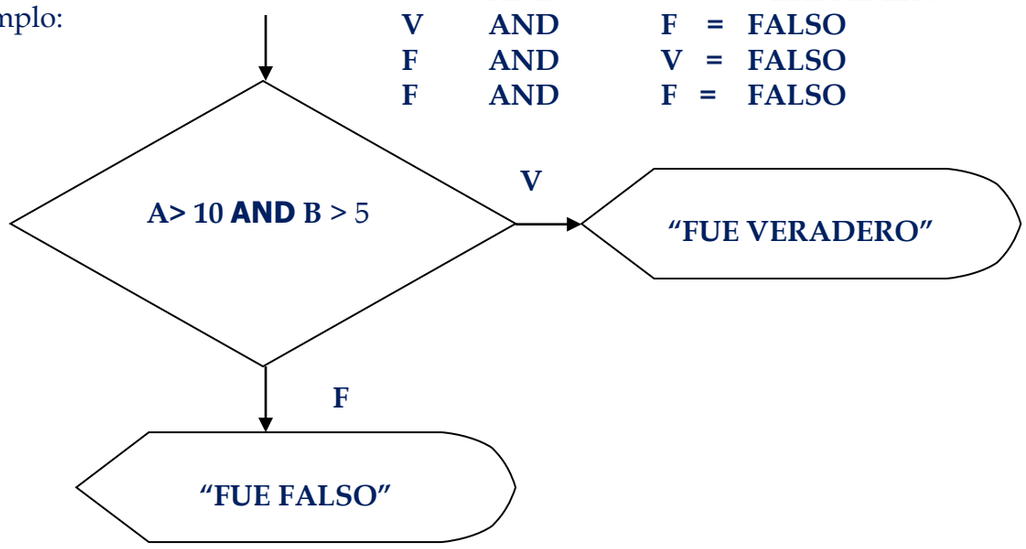
Una **Bifurcación Compuesta** es aquella que se construye a partir de los operadores lógicos **AND** y **OR**.

OPERADOR LÓGICO	Regla
AND	Resulta .T. si todas las condiciones son verdaderas. Obtiene falso cuando una condición es también falsa.

A continuación, la tabla comparativa

	Operador	Resultado
V	**AND**	**V = VERDADERO**
V	**AND**	**F = FALSO**
F	**AND**	**V = FALSO**
F	**AND**	**F = FALSO**

Por ejemplo:

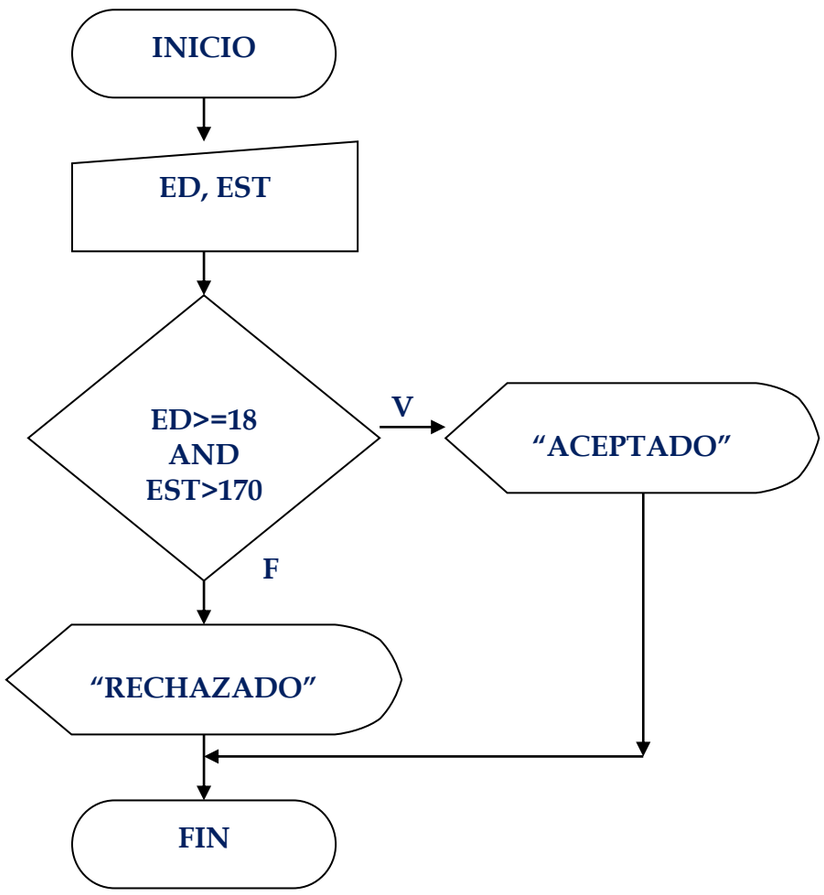


Visualiza "FUE VERADERO" cuando **A** sea mayor que **10** y **B** sea mayor a **5**. En todos los demás casos visualiza "FUE FALSO". Es decir, se requiere que ambas condiciones sean .T. para que la bifurcación sea también verdadera. Si una de las dos condiciones es .F., la bifurcación también es falsa.

EJERCICIOS DE APLICACIÓN

Ejercicio de Aplicación 16

El siguiente diagrama visualiza "ACEPTADO" si la edad es mayor o igual a 18 años y la estatura es superior a 170 cms. Caso contrario visualiza "RECHAZADO"



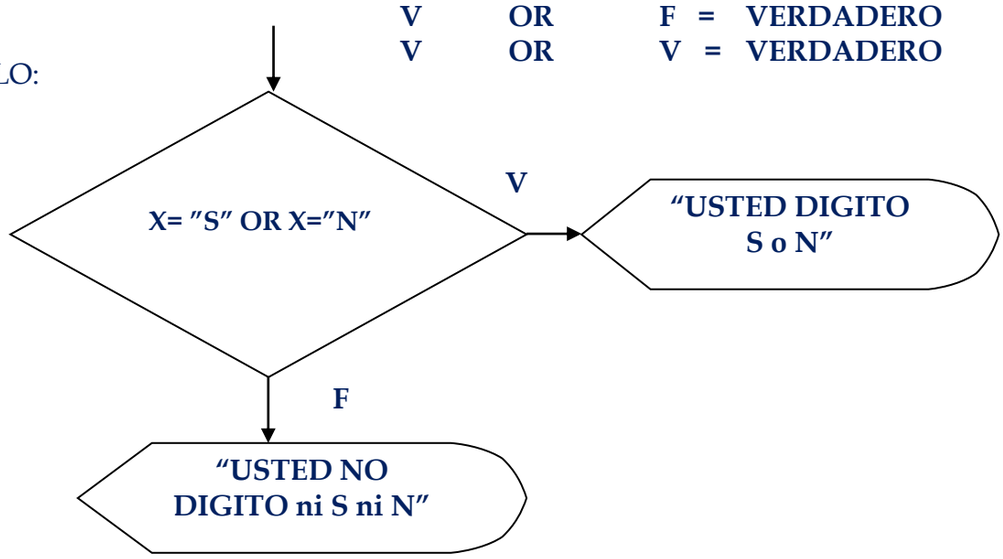
La bifurcación sólo será verdadera cuando **ED>=18** sea **.T.** y **EST>170** sea **.T.** visualizará "ACEPTADO". Si alguna de las dos condiciones es falsa (o las dos), se visualizará "RECHAZADO"

ED	EST	OPERADOR	MENSAJE
25	180	VERADERO	ACEPTADO
20	165	FALSO	RECHAZADO

OPERADOR LÓGICO	Regla
OR	Resulta .T. si al menos una de las condiciones es verdadera. Obtiene falso cuando todas las condiciones son falsas también.

Operador		Resultado
F	**OR**	**F = FALSO**
F	**OR**	**V = VERDADERO**
V	**OR**	**F = VERDADERO**
V	**OR**	**V = VERDADERO**

EJEMPLO:

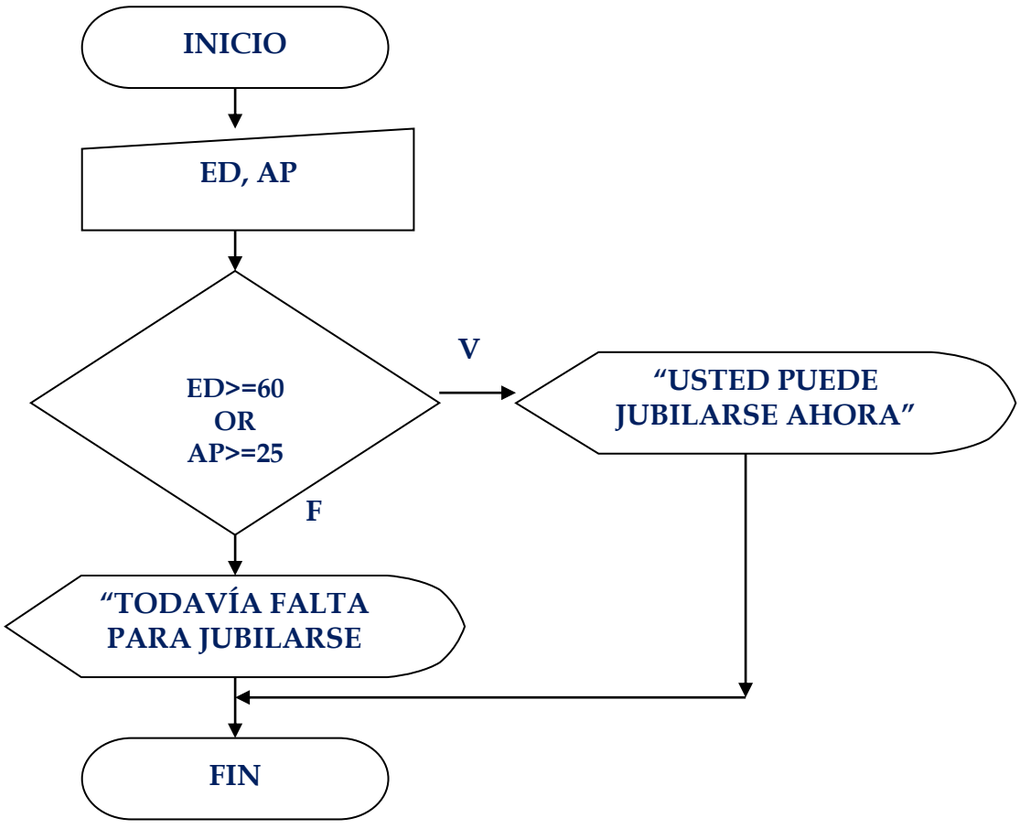


Será .T. cuando **X** sea **“S”** o cuando sea **“N”.** Pero si X no es ni **“S”** ni **“N”,** entonces la bifurcación es falsa. De esta manera, con el operador **Lógico OR** basta que una condición sea verdadera para que la bifurcación también sea verdadera. Sólo cuando ambas condiciones son falsas, la bifurcación también es falsa.

EJERCICIOS DE APLICACIÓN

Ejercicio de Aplicación 17

El siguiente diagrama visualiza el mensaje "USTED PUEDE JUBILARSE AHORA" cuando el trabajador tiene más de 60 años de edad o más de 25 años de aportación. En caso contrario se visualiza "TODAVÍA LE FALTA PARA JUBILARSE".



Sólo basta que **ED** sea mayor o igual a **60** o **AP** sea mayor o igual a **25** para que la bifurcación sea verdadera. Cuando ni la edad sea mayor o igual a **60**, ni los años de aportación mayores o igual a **25**, la bifurcación será falsa.

ED	AP	OPERADOR	MENSAJE
65	18	VERADERO	Usted puede Jubilarse Ahora
20	16	FALSO	Todavía falta para Jubilarse

UNIDAD VI

CICLOS.

Ciclos, definición.

Tipos de Ciclos.

Ciclos Desde…. Hasta

Ejercicios de Aplicación

Ciclos Repetir…. Hasta

Ejercicios de Aplicación

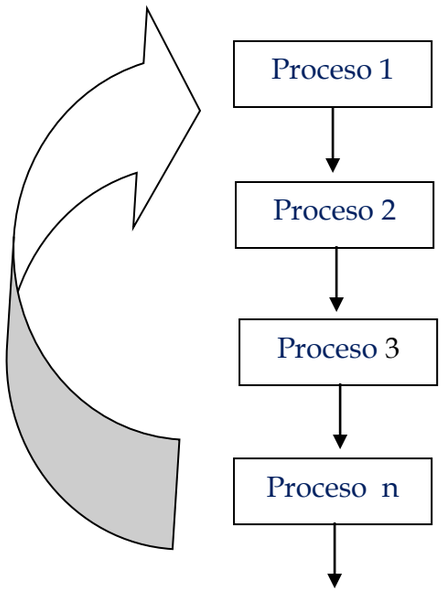
Ciclos Hacer…. Mientras

Ejercicios de Aplicación

UNIDAD VI

CICLOS

Definición. - Se conoce como ***CICLOS*** a todo grupo de instrucciones que son usados para repetir un programa o para repetir un proceso hasta que una condición se cumpla. En ocasiones, por ejemplo, deseamos que cierto grupo de procesos se ejecuten más de una vez. Entonces hacemos que se repitan controlando la cantidad de repeticiones. Esto es **un ciclo o bucle**. Ejemplo.



Existen tres tipos de ciclos:

- Ciclos Desde… Hasta
- Ciclos Repetir…. Hasta
- Ciclos Hacer…. Mientras

CICLOS DESDE…… HASTA

Un **ciclo o bucle**, es un grupo de instrucciones que se repiten un determinado número de veces. Los ciclos son usados para repetir un programa o para repetir un proceso hasta que una condición se cumpla. Ejemplo:

Desde Variable = ***VALOR INICIAL, Hasta VALOR FINAL*****, PASO**

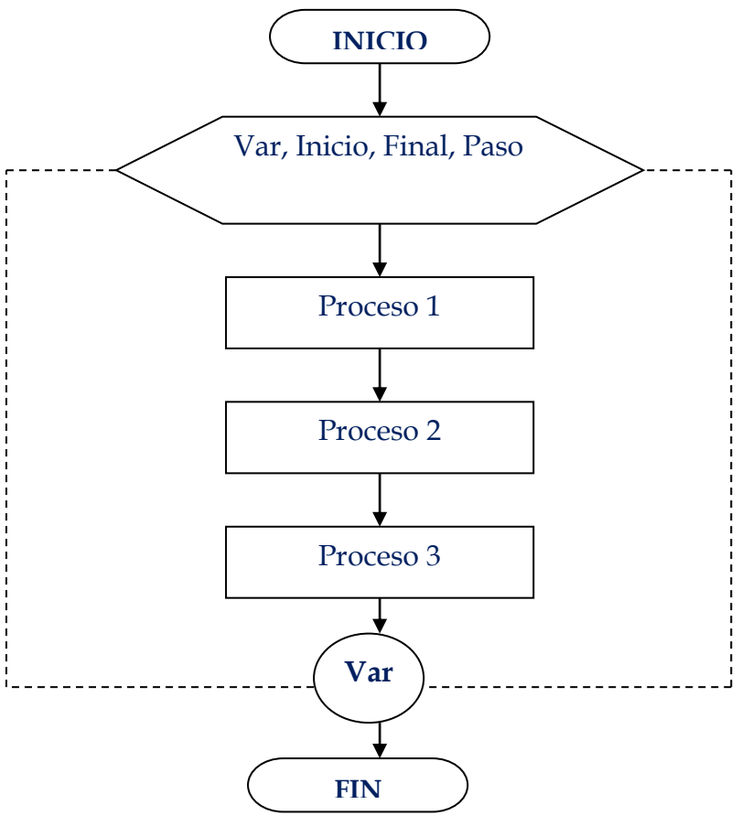
Donde **valor inicial**, es el primer valor que tomará la variable especificada. Esta variable incrementa su valor de **uno en uno** hasta el **valor final.** Ejemplo:

N = 1 TO 10 hará que **N** tome valores desde **1 hasta 10**. Si usted quiere que el contador varíe en pasos diferentes de uno, incluya la cláusula **PASO**. De esta manera: **DESDE VAR = INICIAL, HASTA FINAL PASO**

Donde **PASO** especifica de cuánto en cuánto varía el valor del contador. Ejemplo:

C = 2, 20, PASO 2

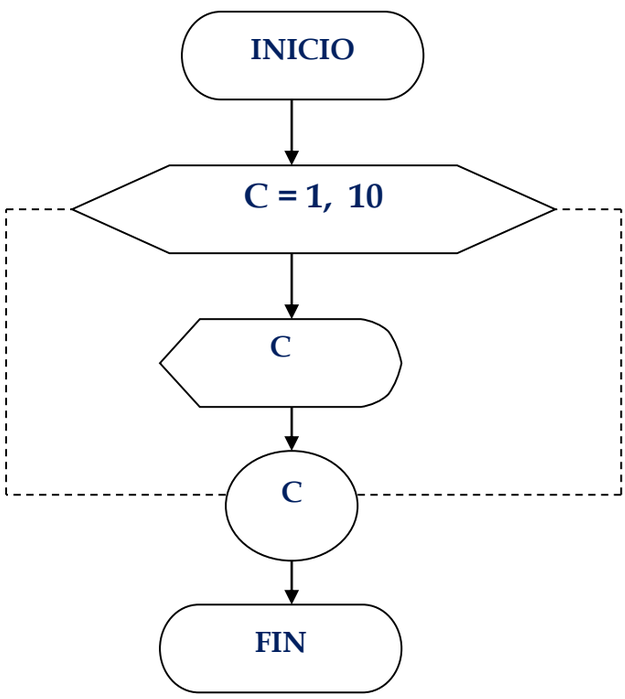
Hace variar C de 2 en 2 hasta el 20. Si desea puede incluir como paso un valor negativo para series descendentes



EJERCICIOS DE APLICACIÓN

Ejercicio de Aplicación 01

A continuación, un diagrama de flujo que visualiza los 10 primeros números consecutivos.



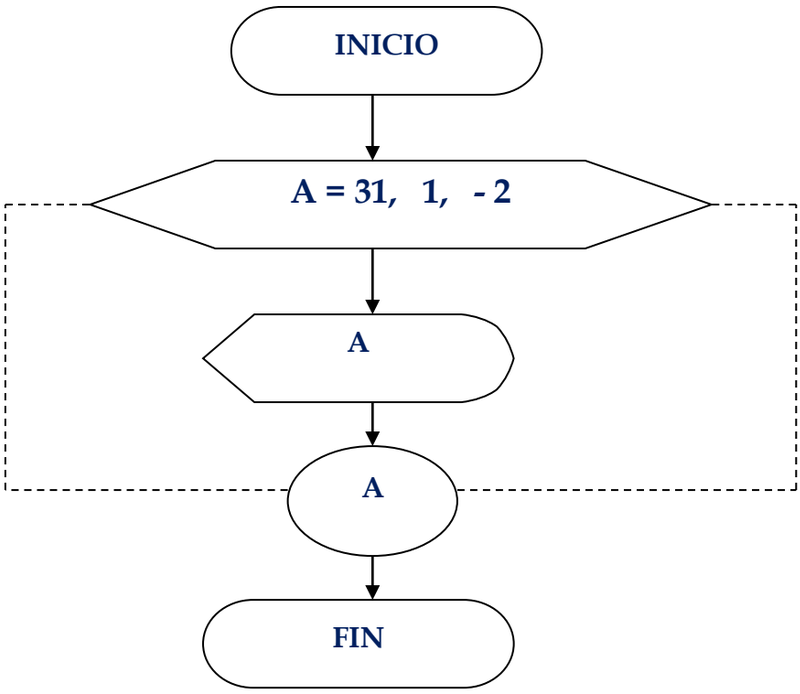
Hace variar C desde 1 hasta el 10 de uno en uno. El PASO se omite, se sobreentiende que el incremento será en 1.

Prueba de Escritorio:

C = 1
C = 2
C = 3
C = 4
C = 5
C = 6
C = 7
C = 8
C = 9
C = 10

Ejercicio de Aplicación 02

El siguiente diagrama de flujo visualiza los números impares en orden descendente del el 31 hasta el 1.

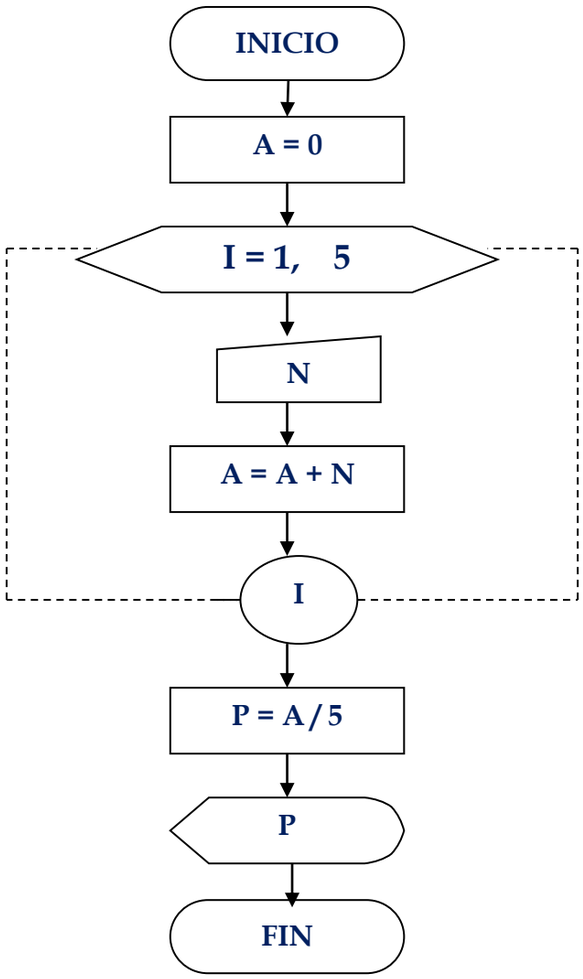


Desciende **A** desde **31** hasta **1** de 2 en 2. Se visualiza el orden descendente porque el paso es negativo y el **valor inicial** es mayor al **valor final**.
La prueba de Escritorio es:

A = 31
A = 29
A = 27
A = 25
A = 25
A = 23
A = 21
A = 19
A = 17
A = 15
A = 13
A = 11
A = 9
A = 7
A = 5
A = 3
A = 1

Ejercicio de Aplicación 03

A continuación, un diagrama de flujo que ingresa 5 notas y visualiza su promedio.



Para este ejercicio es necesario acumular, por que al final del mismo debemos visualizar el promedio de las notas.

Entonces el acumulador es **A = 0.**

Empezamos el **ciclo** con **I = 1, 5**, es decir, desde el **uno** hasta el **cinco**.

Luego ingresamos la variable **N** de nota.

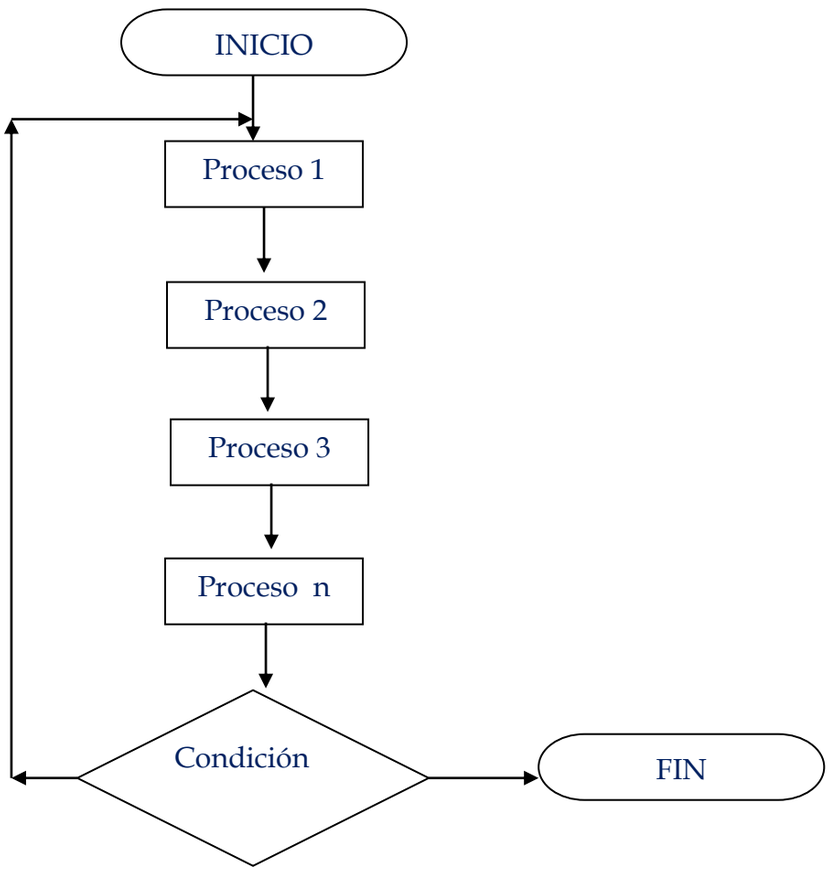
Realizamos un proceso en donde el **acumulador** es igual a **acumulador** más **nota. A = A + N**

Cuando el ciclo termine realizamos un proceso **P = A / 5**, y visualizamos el **promedio** de las notas.

Su prueba de escritorio es:

I	**N**	**A**	
1	18	18	El **uno** es del ciclo, su nota es **18**; acumulador es A = A + N(18) => A =18.
2	20	38	Ciclo **DOS**, nota **20**, Acumulador es A = A(18) + N(20) => A = 38
3	12	50	Ciclo **TRES**, nota **12**, Acumulador es A = A(38) + N(12) => A = 50
4	10	60	Ciclo **CUATRO**, nota **10**, Acumulador es A = A(50) + N(10) => A = 60
5	15	75	Ciclo **CINCO**, nota **15**, Acumulador es A = A(60) + N(15) => A = 75
			Entonces el ciclo termina y realizamos otro proceso **P = A / 5**, es decir, **promedio = acumulador (75) / cinco** y obtenemos el promedio de **15**

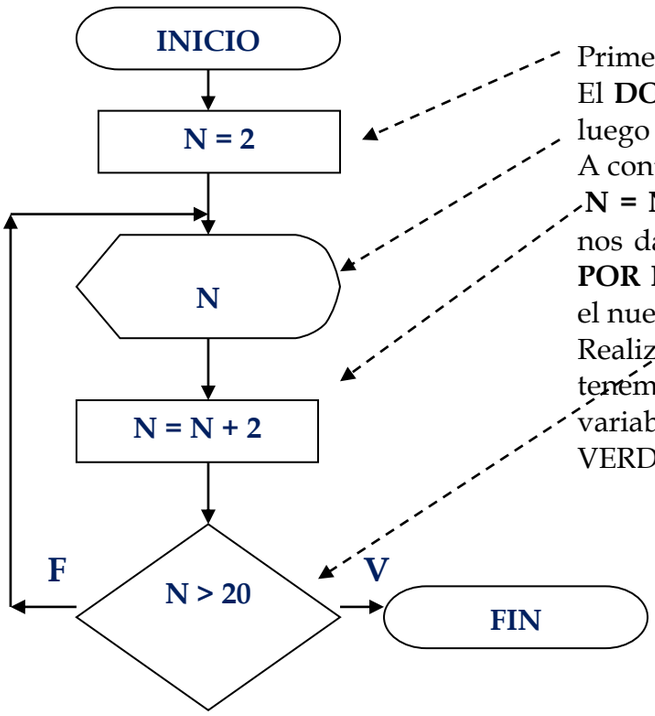
Este tipo de ciclo se repite ***hasta*** que una **condición sea verdadera**. Como estudiamos en el capítulo anterior, las condiciones pueden ser Verdaderas o Falsas. Cuando ponemos una condición en un ciclo Repetir…Hasta, todas las instrucciones o procesos del ciclo se repetirán mientras la condición sea falsa; en el caso de que sea verdadera, el ciclo termina. Ejemplo



EJERCICIOS DE APLICACIÓN

Ejercicio de Aplicación 04

El siguiente diagrama visualiza los 10 primeros números pares.



Primero inicializamos una variable **N = 2.** El **DOS** porque es el primer número par, luego visualiza la variable **N**.
A continuación realizamos un proceso **N = N + 2**; donde **N** es igual a **2 + 2** que nos da **4** y PREGUNTAMOS **SI N(4) > 20 POR FALSO** vuelve a ingresar y presenta el nuevo valor **4**.
Realizamos el proceso **N=N(4)+2** y tenemos **6**, así sucesivamente hasta que la variable **N** sea mayor a **20** y esto nos da VERDADERO y finalizamos

PRUEBA DE ESCRITORIO

VARIABLE(N)	CONDICIÓN (N>20)	IMPRIME O VISUALIZA
2	falso	2
4	falso	4
6	falso	6
8	falso	8
10	falso	10
12	falso	12
14	falso	14
16	falso	16
18	falso	18
20	falso	20
22	verdadero (finaliza el ciclo)	

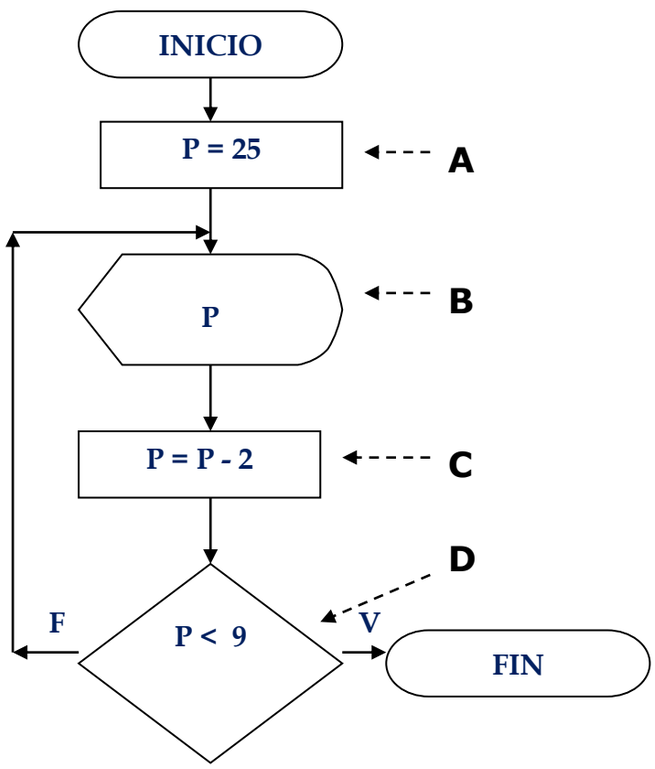
Tal como lo ve, el ejercicio solo presenta o imprime los 10 primeros números pares.

En este ejercicio no es necesario **INGRESAR UNA VARIABLE POR TECLADO** porque solo vamos a presentar un valor inicial de **2** y un valor final de **20**. Por que el **DOS; porque** es el primer número par y sus **10** primeros números pares es el **20. Ejemplo (2 * 10 = 20).**

Ejercicio de Aplicación 05

El siguiente flujograma visualiza en orden descendente los números impares del 25 al 9.

Empezamos inicializando una variable **P = 25;** El **25** que representa al valor inicial del ejercicio y luego visualizamos la variable **P** que almacena **25**.
Realizamos un proceso **P = P - 2**; donde **P** es igual a **25 - 2** que nos da **23** y preguntamos **SI P(23) < 9. POR FALSO** vuelve a ingresar y visualizamos el nuevo valor **23**.
Realizamos el proceso **P = P(23) - 2** y tenemos **21**.
Y volvemos a preguntar **SI P(21)< 9;** así sucesivamente hasta que la variable **P** sea menor a 9 y esto nos da VERDADERO y finalizamos el ciclo.



a) Primero inicializamos una variable **P = 25**

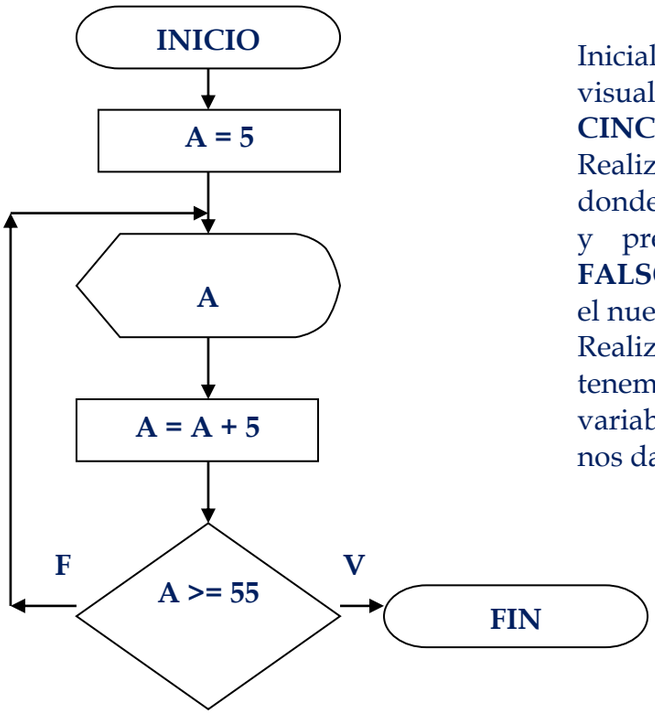
b) Luego visualiza la variable **P** que en este caso vale **25**

c) A continuación, realizamos un proceso **P = P - 2** y vamos a obtener un nuevo valor **P = P (25) - 2**, que es el **23**

d) Preguntamos **SI P (23) < 9;** por falso vuelve a ingresar y visualiza el nuevo número **23** y realizamos el proceso **P = P (23) - 2**, que nos da **21** y OTRA VEZ preguntamos **SI P (21) < 9.** Así sucesivamente hasta que la variable **P** sea menor a **9 y el ciclo terminará.**

Ejercicio de Aplicación 06

A continuación, un diagrama que visualiza los 10 primeros múltiplos de 5.



Inicializamos la variable **A = 5**, luego visualizamos la variable **A**. que vale **CINCO**.
Realizamos un proceso **A = A + 5**; en donde **A** es igual **A(5) + 5** y obtenemos **10** y preguntamos **SI A(10) >=55 POR FALSO** vuelve a ingresar y visualizamos el nuevo valor **10.**
Realizamos el proceso **A = A(10)+5** y tenemos **15**, así sucesivamente hasta que la variable **A** sea mayor o igual a **55** y esto nos da VERDADERO y finalizamos.

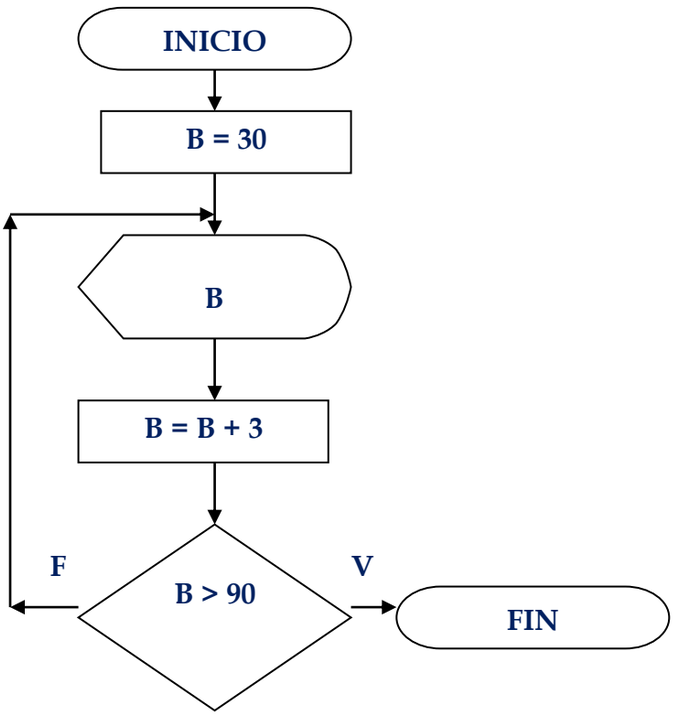
PRUEBA DE ESCRITORIO

VARIABLE(A)	CONDICIÓN (A>=5)	IMPRIME O VISUALIZA
5	falso	5
10	falso	10
15	falso	15
20	falso	20
25	falso	25
30	falso	30
35	falso	35
40	falso	40
45	falso	45
50	falso	50
55	verdadero (finaliza el ciclo)	

Imprime los 10 primeros múltiplos de 5. HASTA que **5 * 10 = 50** y por **verdadero** finaliza el ciclo.

Ejercicio de Aplicación 07

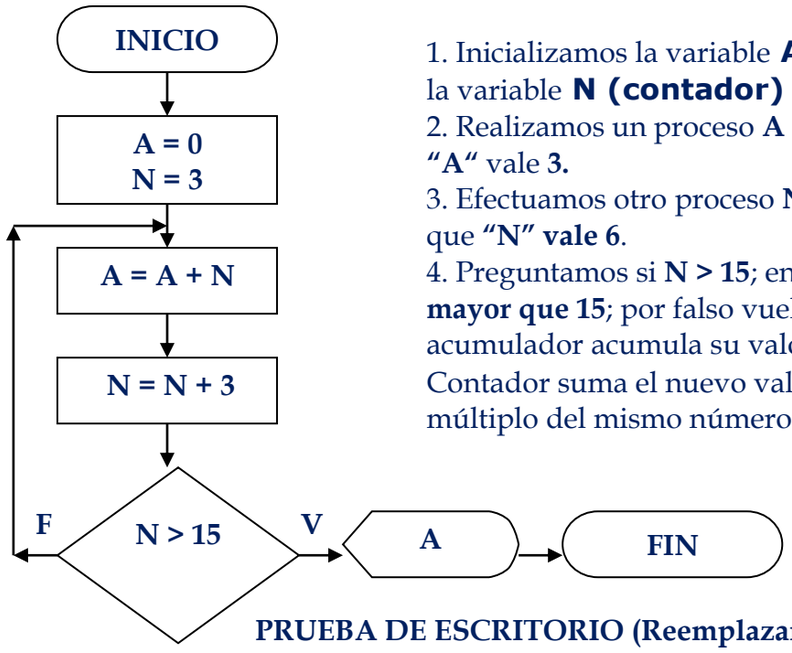
El siguiente flujograma visualiza todos los múltiplos de 3 entre 30 y 90



Analicemos el ejercicio, los múltiplos de 3 son: 3, 6, 9, 12,….n.
Pero como son entre 30 y 90 comenzamos: 30, 33, 36, 39, 42, 45…….90
Empezamos dando un valor inicial a nuestra variable **B = 30**.
Visualizamos la variable **B**, pero recuerde que vale **30.**
Procedemos a realizar un proceso **B = B +3**; donde **B** es igual a **B(30) + 3** y tenemos **33.**
Preguntamos **SI B(33) > 90 POR FALSO** vuelve a ingresar y visualiza el nuevo valor **33.**
Realizamos el proceso otra vez y tenemos **B = B(33) + 3** que nos da **36**.
Para que el ciclo termine la variable **B** debe ser mayor a **90 y si es mayor es VERDADERO y el ciclo termina.**

CONTADORES:
Un **contador** es una **variable** que **incrementa** su **valor** en una **cantidad constante**.
Por Ejemplo: C = C + 1; es la forma más simple de contar.
Así mismo el incremento puede ser en cualquier valor constante.
Este tipo de instrucción es usada con los ciclos para incrementar una **SERIE NUMÉRICA,** en este capítulo la usaremos para contabilizar el número de veces que se repite un proceso.
Además, hay que señalar que toda **variable contadora** debe ser **inicializada** antes de **entrar** al ciclo. Esto es para que empiece su valor con cero y pueda incrementarlo dentro del ciclo sin ningún problema.

Ejercicio de Aplicación 08
El siguiente diagrama de flujo visualiza la sumatoria de los 5 primeros múltiplos de 3.



1. Inicializamos la variable **A (acumulador)** y la variable **N (contador)**
2. Realizamos un proceso **A = A(0) + N(3);** donde **"A"** vale **3.**
3. Efectuamos otro proceso **N = N(3)+N** tenemos que **"N" vale 6**.
4. Preguntamos si **N > 15**; en este caso **N(6) es mayor que 15**; por falso vuele a ingresar y el acumulador acumula su valor; mientras que el Contador suma el nuevo valor **+ 3** por ser múltiplo del mismo número.

PRUEBA DE ESCRITORIO (Reemplazamos los valores)

A = A (0) + N (3); ► A = 0 + 3; A =►**3**
A = A **(3)** + N (6) =►**9**
A = A **(9)** + N (9) =►**18**
A = A **(18)** + N (12) =►**30**
A = A **(30)** + N (15) =►**45**

N = N + 3; ► N = 3 + 3; N =► 6
N = N(6) + 3; N =9
N = N(9) + 3; N = 12
N = N(12) + 3; N = 15
N = N(15) + 3; N = 18; Preguntamos si **N > 15** como es verdadero visualizamos la variable **"A"** que vale **45**

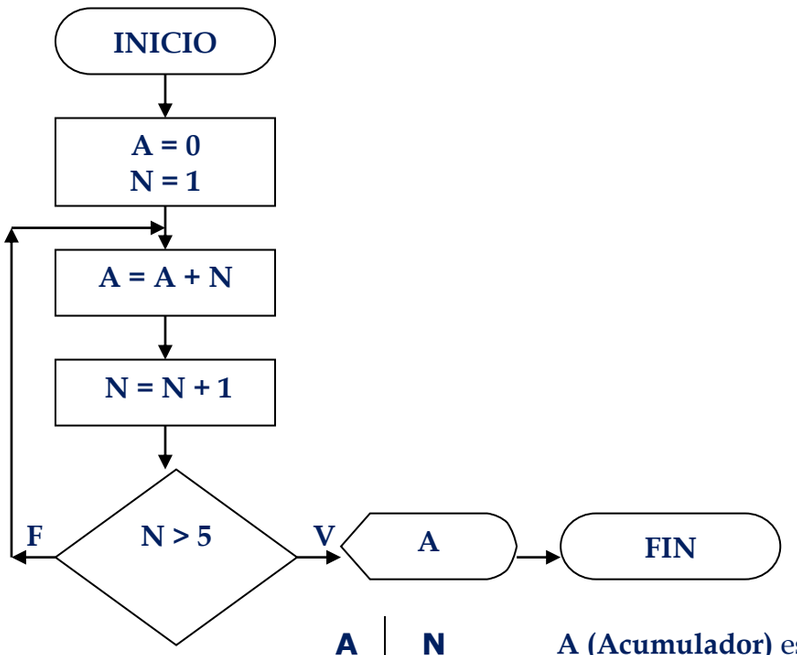
ACUMULADORES:
De función similar a la de los contadores, éstos se **encargan** de **acumular** un **valor** en una **cantidad variable**. La forma más simple de acumular es:
A = A + X
Donde **X** es una cantidad variable. Al igual que los contadores deben ser inicializados al principio del diagrama, es decir, antes de entrar al ciclo.
Un acumulador es una variable que incrementa o acumula un determinado valor. Por ejemplo: si deseamos saber a cuánto equivale la sumatoria de los 5 primeros números:

1 + 2 + 3 + 4 + 5 = 15

El acumulador sería: 1+2 = **3;** 3+3=**6;** 6+4=**10;** 10+5=**15**

Generalmente los acumuladores se inicializan con cero (0). Ejemplo:

Ejercicio de Aplicación 09
A continuación, un diagrama de flujo visualiza la sumatoria de los 5 primeros números consecutivos.



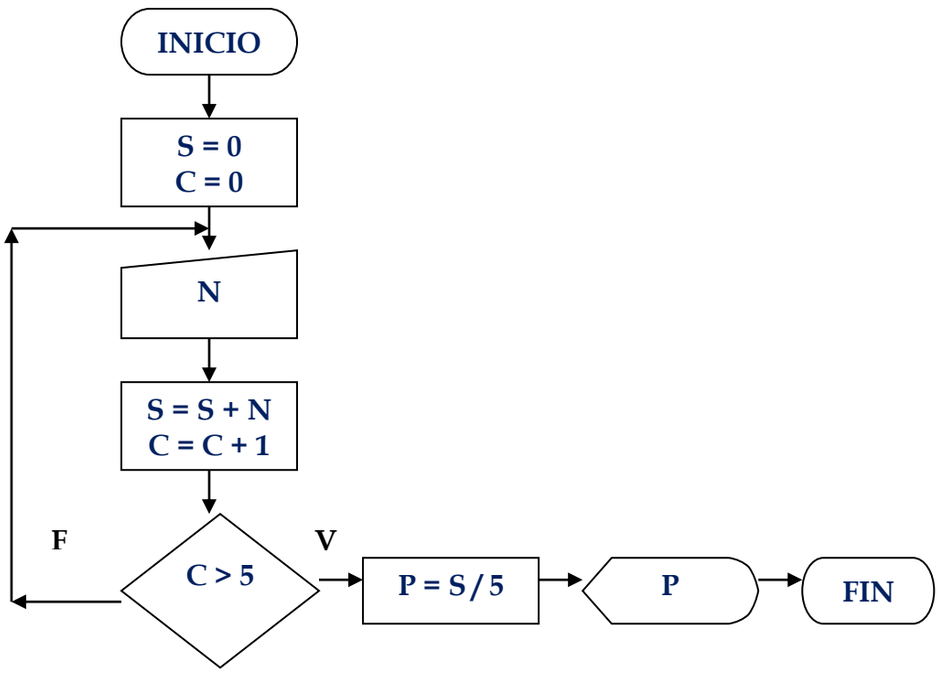
A	N
0	1
1	2
3	3
6	4
10	5
15	6

A (Acumulador) es el que se encarga de acumular.

Cuando **N (Contador)** sea mayor que **5**, es decir, **"SEIS"**, será verdadero y visualizará la variable **"A"** cuyo valor acumulado es **15**

Ejercicio de Aplicación 10

El siguiente diagrama de flujo ingresa 5 calificaciones. Luego presenta su promedio.



S (acumulador)	C (contador)	N (nota)	P (promedio)
0	1	16	
16	2	18	
34	3	12	
46	4	14	
50	5	15	**15**
75	6		

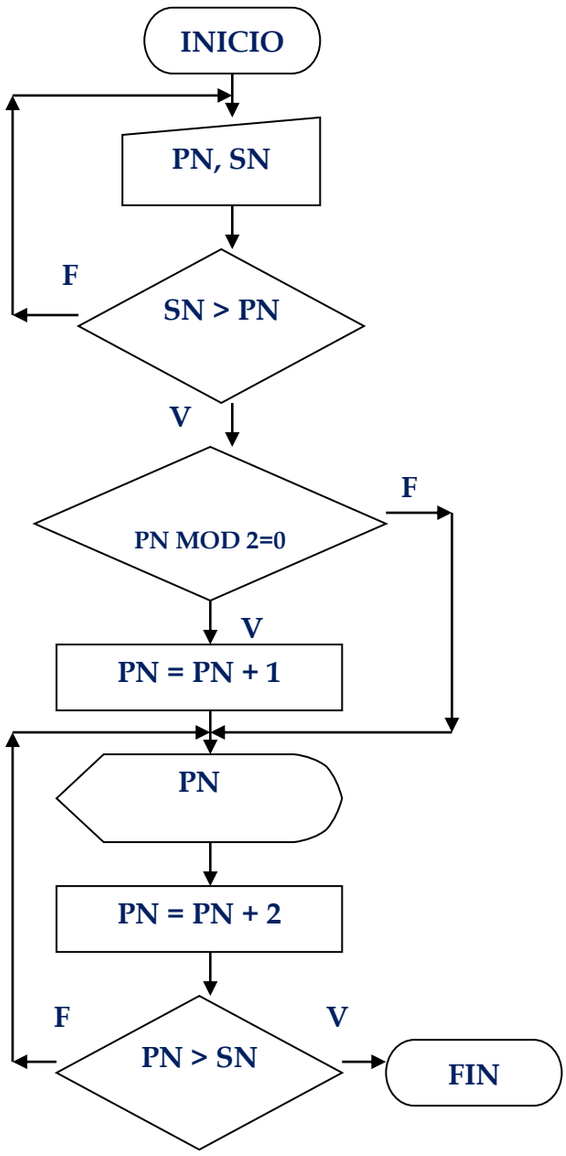
Como **C(6)** es mayor que **5**; realizamos un proceso, en la que obtenemos el promedio **P=S/5; P =75/5 ►15**

Ejercicio de Aplicación 11

Ingrese dos números por teclado. Valide que el segundo número sea mayor que el primer número. Luego visualice los NÚMEROS IMPARES entre dichos limites.

Iniciamos ingresando por teclado los dos números **PN, SN**.
Validamos los dos números y preguntamos si **SN** es mayor que **PN;** por **FALSO** vuelven a ingresar.
Por **VERDADERO** a **PN** le sacamos el **MOD (**El operador ***MOD*** obtiene el residuo, es decir, lo que sobra de una división); EJEMPLO: **PN(12) MOD 2=0 (12/2=6 y como residuo tenemos CERO)** esto nos da VERDADERO y realizamos un proceso **PN = PN(12) +1** esto es **13,** visualizamos la variable **PN;**
Realizamos otro proceso **PN = PN(12) +2 igual a 14**; preguntamos si **PN > SN**, por FALSO vuelve a presentar la variable **PN(14)**; continuamos **PN = PN(14)+2** es **16;** volvemos a preguntar **PN(16)>SN;** si es VERDADERO el Ciclo termina.

Otro EJEMPLO: **PN(13) MOD 2=0** es **FALSO** (13/2=6 y el residuo 1) VISUALIZAMOS LA VARIABLE **PN(13)**; realizamos un proceso **PN = PN(13) + 2**; éste nos da un nuevo resultado **15** y preguntamos **SI PN(15)>SN** por falso presentamos **PN(15)** , otra vez el proceso y preguntamos.
En caso que **PN** sea mayor que **SN** el ciclo terminará



Que el primer número ingresado sea par; **entonces se necesita la conversión a número impar.** Luego visualice los NÚMEROS IMPARES entre dichos limites.

PRIMER CASO:

PRUEBA DE ESCRITORIO:

PN (4) SN (10)

PN > SN VERDADERO

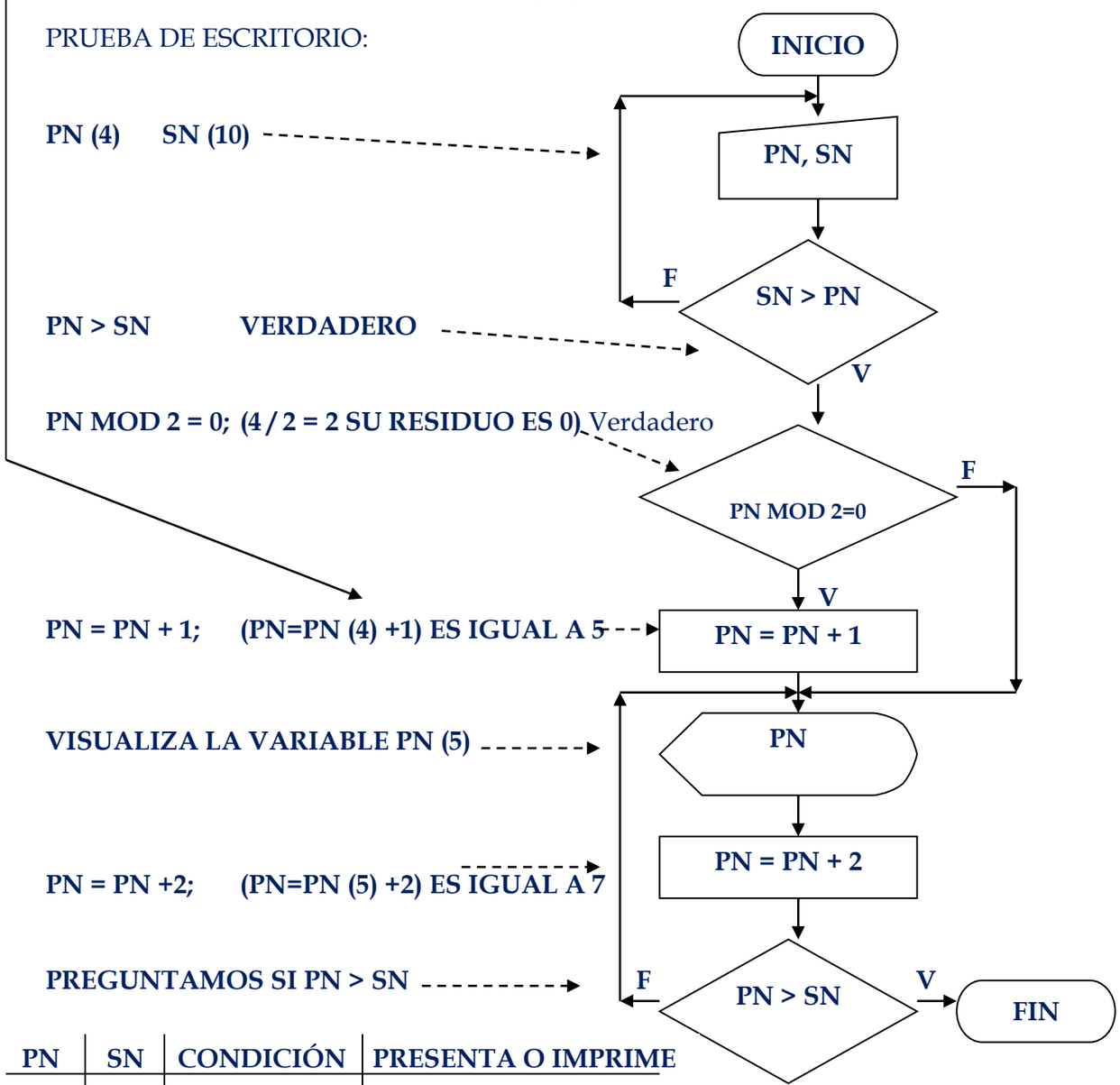
PN MOD 2 = 0; (4 / 2 = 2 SU RESIDUO ES 0) Verdadero

PN = PN + 1; (PN=PN (4) +1) ES IGUAL A 5

VISUALIZA LA VARIABLE PN (5)

PN = PN +2; (PN=PN (5) +2) ES IGUAL A 7

PREGUNTAMOS SI PN > SN



PN	SN	CONDICIÓN	PRESENTA O IMPRIME
4	10	verdadero	
5		falso	5
7		falso	7
9		falso	9
11		verdadero	

Que el primer número ingresado sea impar; **no es necesaria la conversión a número impar.** Luego visualice los IMPARES entre dichos límites.

SEGUNDO CASO:

PRUEBA DE ESCRITORIO:

PN (5) SN (11)

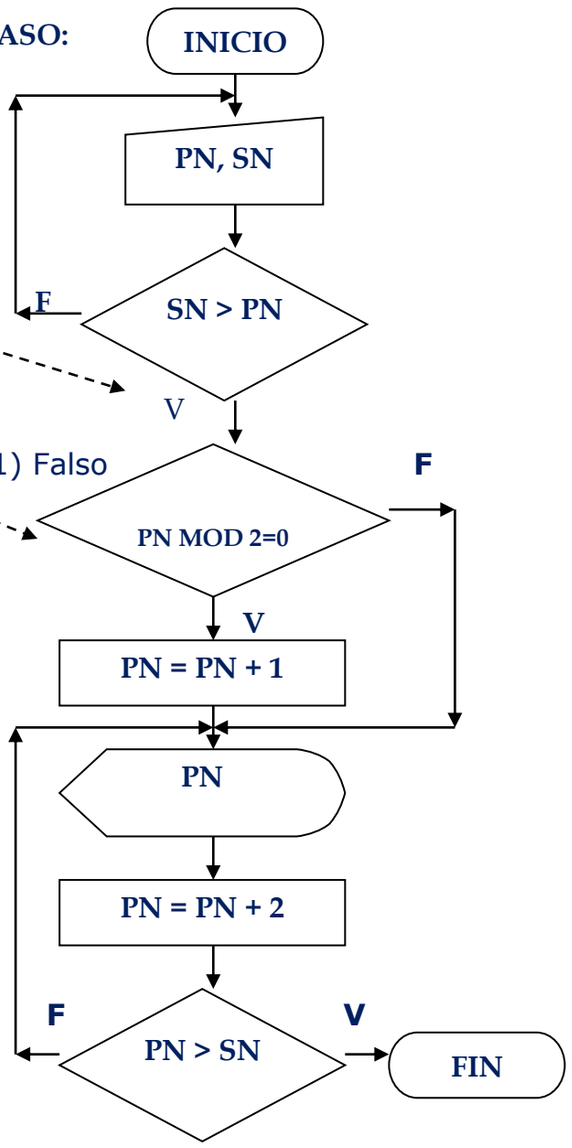
PN > SN VERDADERO

PN MOD 2 = 0; (5/2 =2 su residuo es 1) Falso

VISUALIZA LA VARIABLE PN (5)

PN = PN +2; (PN=PN (5) +2) ES IGUAL A 7

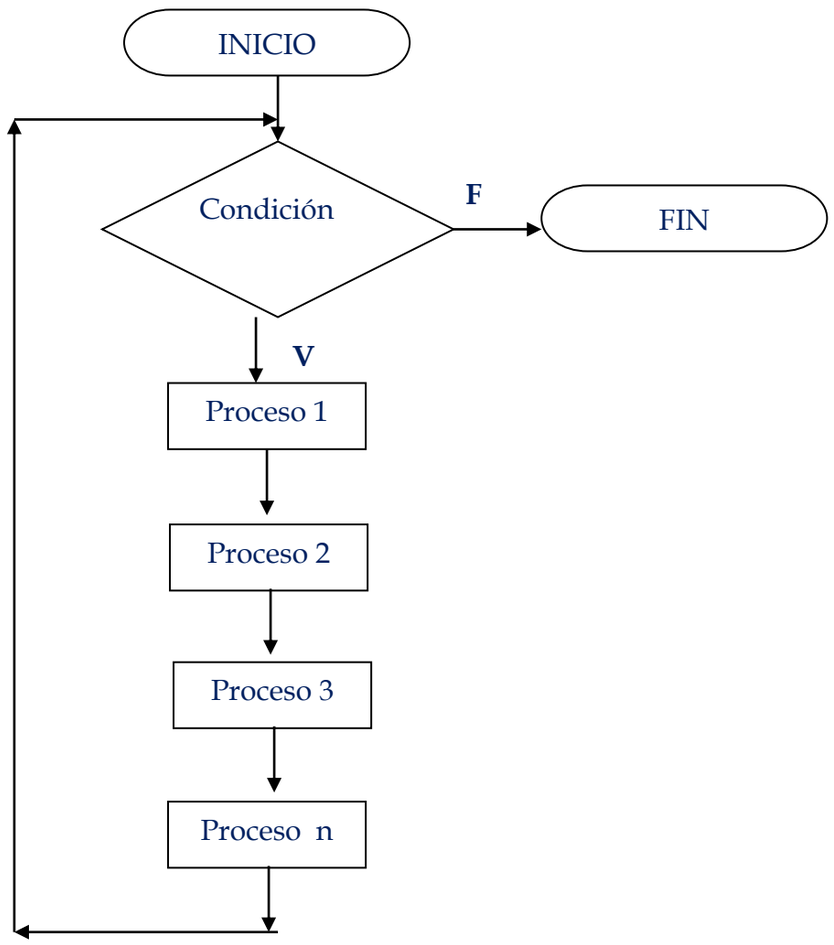
PREGUNTAMOS SI PN > SN



PN	SN	CONDICIÓN	VISUALIZA O IMPRIME
5	11	verdadero	
5		falso	5
7		falso	7
9		falso	9
11		falso	11
13		verdadero	

CICLOS HACER…… MIENTRAS

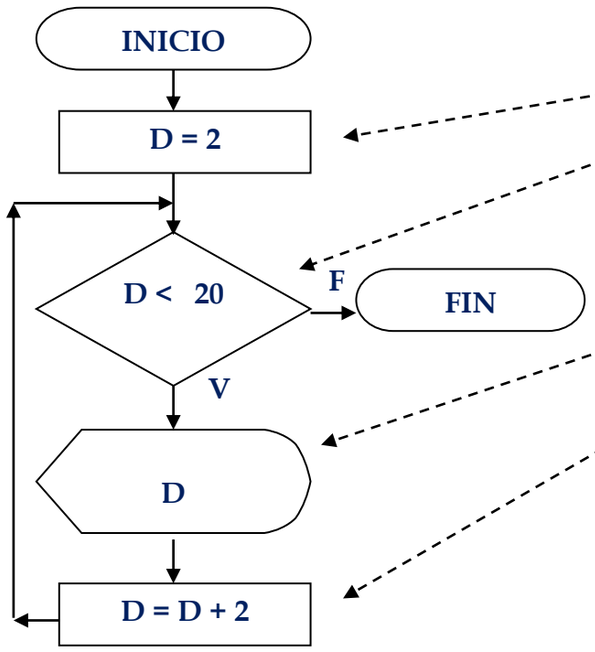
Este tipo de ciclo se repite ***MIENTRAS*** una condición sea verdadera. Note la diferencia entre ***repetir hasta*** y **repetir mientras**. Esto significa que el ciclo se repite tantas veces cuando la condición sea verdadera, contrastando con los ciclos Repetir... hasta, que se repiten tantas veces cuando la condición sea falsa. A continuación, un cuadro comparativo:



EJERCICIOS DE APLICACIÓN

Ejercicio de Aplicación 12

El siguiente diagrama visualiza los 10 primeros números pares.



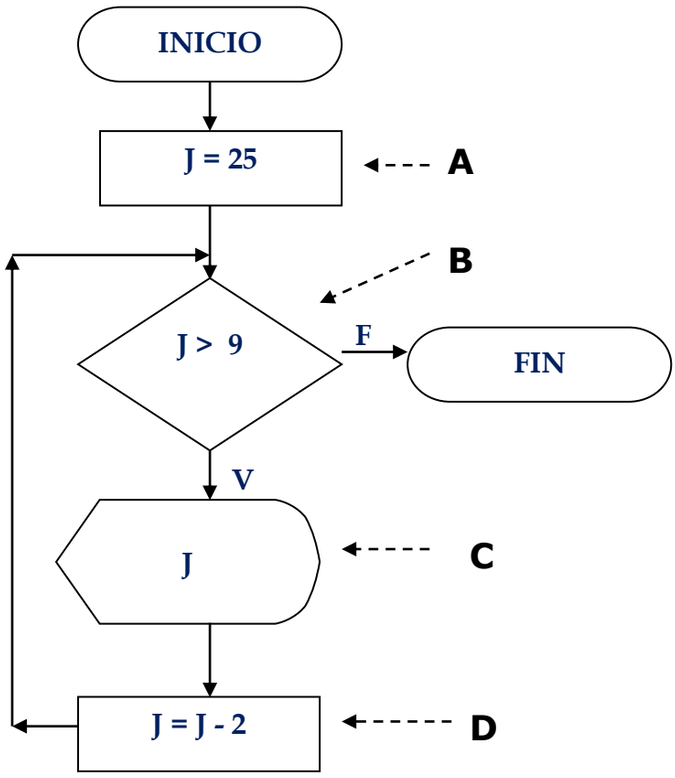
Proporcionamos un valor inicial a nuestra variable **D = 2.**
Preguntamos si **D< 20**; es decir, **SI 2 ES MENOR QUE 20**, por verdadero, visualizamos la VARIABLE **D.**
Realizamos un proceso **D = D + 2** y obtenemos un nuevo valor **D = D(2) + 2** es igual a **4.**
A continuación volvemos a preguntar si **D(4) ES MENOR QUE 20**, como es **VERDADERO**, visualizamos la **Variable** pero con un nuevo valor **D(4)**, así sucesivamente hasta que la variable **D** sea **MAYOR a 20 y** por **FALSO** finalizamos.

PRUEBA DE ESCRITORIO

VARIABLE (D)	CONDICIÓN (D < 20)	IMPRIME O VISUALIZA
2	verdadero	2
4	verdadero	4
6	verdadero	6
8	verdadero	8
10	verdadero	10
12	verdadero	12
14	verdadero	14
16	verdadero	16
18	verdadero	18
20	verdadero	20
22	FALSO (finaliza el ciclo)	

Ejercicio de Aplicación 13

El siguiente flujograma visualiza en orden descendente los números impares del 25 al 9.



A) Primero inicializamos una variable **J = 25**

B) Preguntamos **SI J (25) > =9;** por VERDADERO

C) Visualizamos la **VARIABLE J** con su respectivo valor **25.**

D) Realizamos el proceso **J = J (25) - 2**, que nos da **23;** OTRA VEZ preguntamos **SI J (23) > = 9.** Así sucesivamente hasta que la variable **J** sea mayor o igual a **9 y el ciclo terminará.**

Descubre tu próxima lectura

Si quieres formar parte de nuestra comunidad,
regístrate en **https://www.grupocompas.org/suscribirse**
y recibirás recomendaciones y capacitación

compAs

Grupo de capacitación e investigación pedagógica

@grupocompas.ec
compasacademico@icloud.com

compAs
Grupo de capacitación e investigación pedagógica

@grupocompas.ec
compasacademico@icloud.com

ISBN: 978-9942-33-187-8

@grupocompas.ec
compasacademico@icloud.com

compAs
Grupo de capacitación e investigación pedagógica