Unidad didáctica 3071

Contenidos

Capítulo II.

Sistemas numéricos, operadores y precedencia

Objetivo general

Adquirir conocimientos sobre las características de los sistemas numéricos, los métodos de conversión entre sistemas y el orden de precedencia de resolución de expresiones metemáticas.

Objetivos específicos

Conocer los elementos de los sistemas numéricos: decimal, binario, octal y hexadecimal.

Implementar los métodos de divisiones sucesivas y suma de productos para la conversión de valores entre sistemas numéricos.

Objetivos específicos

Conocer las características de los operadores de asignación, aritméticos, relacionales y lógicos.

Aplicar las reglas de precedencia en la resolución de expresiones matemáticas.

Introducción

En este capítulo se muestran las características de cuatro sistemas numéricos y los métodos de conversión entre los mismos. Además, se explican las características y funcionamiento de diversos operadores y finalmente, se analizan las reglas de precedencia matemática en la resolución de expresiones.

La primera parte del capítulo se enfoca en los sistemas numéricos: decimal, binario, octal y hexadecimal; abarcando tanto los elementos de los componen como sus bases. La segunda sección del capítulo está relacionada directamente con los sistemas numéricos, ya que explica los métodos de conversión, primero el de divisiones sucesivas para convertir un número decimal a su equivalente en cualquiera de los otros tres sistemas y segundo el método de suma de productos para efectuar la conversión de un sistema numérico cualquiera al decimal.

En su tercera parte, el capítulo se enfoca en los operadores de asignación, aritméticos, relacionales (comparativos) y lógicos, para cada uno de ellos se exponen sus características y se analiza su funcionamiento. La cuarta parte analiza las reglas de precedencia para resolver expresiones matemáticas tomando en cuenta los operadores y otros elementos como por ejemplo signos de negatividad y paréntesis.

Finalmente, se brindan explicaciones detalladas de expresiones desarrolladas, así como ejercicios de autoevaluación con sus respectivas respuestas.

Guía de lectura

Este segundo capítulo se centra en el análisis de sistemas numéricos y de operadores, así como la implementación de reglas de precedencia en expresiones matemáticas.

Se inicia con una descripción de cuatro sistemas numéricos, se abarca desde su estructura hasta la conversión entre sistemas.

Posteriormente se entra al tema de operadores, en este punto se explica el funcionamiento y características de operadores unarios, aritméticos, relacionales y lógicos.

Finalmente, se analizan las reglas de precedencia y se resuelven expresiones matemáticas utilizando dichas reglas. Se recomienda poner especial atención a los ejemplos desarrollados y al final del capítulo es ideal que realice los ejercicios de autoevaluación.

Para el estudio adecuado de este capítulo se recomienda seguir el orden de los temas como se presentan, de esta forma, el lector contará con los conocimientos básicos y consecutivos de los contenidos que irá estudiando.

1.1 Definición

Un sistema numérico es un conjunto de símbolos que representan diferentes valores y se emplean para realizar cálculos matemáticos y expresar cantidades. Los sistemas que se estudiarán a continuación emplean el método posicional para otorgar valor a un símbolo, es decir, dependiendo de la posición que ocupe el símbolo tendrá mayor o menor valor.

Todo sistema numérico posee una base, este dato está relacionado directamente con la cantidad de elementos que posea el sistema. Por ejemplo, si un sistema numérico tiene 2 elementos, su base será 2, si posee 8 su base será 8 y si se compone de 10 elementos su base será 10. La representación de la base se hace por medio de un número más pequeño (en formato subíndice) que el número que se desea representar. Veamos los siguientes casos, para representar los números 25 en base 8, 2 en base 10 o 11 en base 2; lo hacemos como se muestra en la figura 2.1:

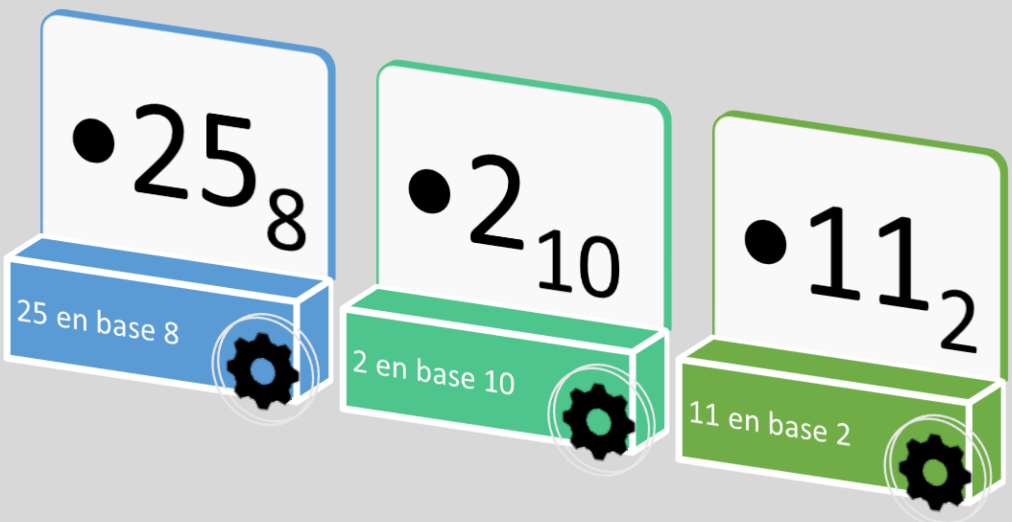


Figura 2.1. Representación de números en base 8, base 10 y base 2
Fuente: Aguilera y Bejarano, 2018

Recordemos que una representación numérica contará con su indicador de cantidad y su indicador de base. Estos dos elementos se identificarán fácilmente por su diferencia de tamaño.

Figura 2.2. Posición del elemento de cantidad y del elemento base en número en base 8
Fuente: Aguilera y Bejarano, 2018

El sistema numérico con el que seguramente estas más familiarizado es el **decimal**, el cual nos enseñan desde los primeros pasos de nuestra vida, sin embargo, hay muchos sistemas numéricos más, como por ejemplo el binario, el octal y el hexadecimal.

En general, un sistema numérico puede ser determinado por el simple hecho de juntar símbolos y asignarles un valor a cada uno de ellos. Lo relevante es tener el conocimiento para comprender el valor que representa una cantidad expresada por los símbolos de ese sistema y para esto debemos hacer una comparación entre lo expresado en el sistema numérico "nuevo" y su valor equivalente en el sistema numérico decimal, ya que es el que conocemos mejor.

1.1.1 Sistema numérico decimal

Este sistema está compuesto de diez dígitos, por ende, su base es 10, y es fundamental en el aprendizaje de diversas ciencias. Tenemos contacto con el mismo desde los primeros años de vida. Los dígitos del sistema decimal son los siguientes: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9; cada uno de ellos representa una cantidad específica y combinándolos se puede representar cualquier cantidad numérica que deseemos.

Para representar un número en el sistema decimal simplemente lo escribimos con o sin su base. Por acuerdo, se establece que si un número no indica su base se asume que está representado en el sistema decimal, por ejemplo (figura 2.3):

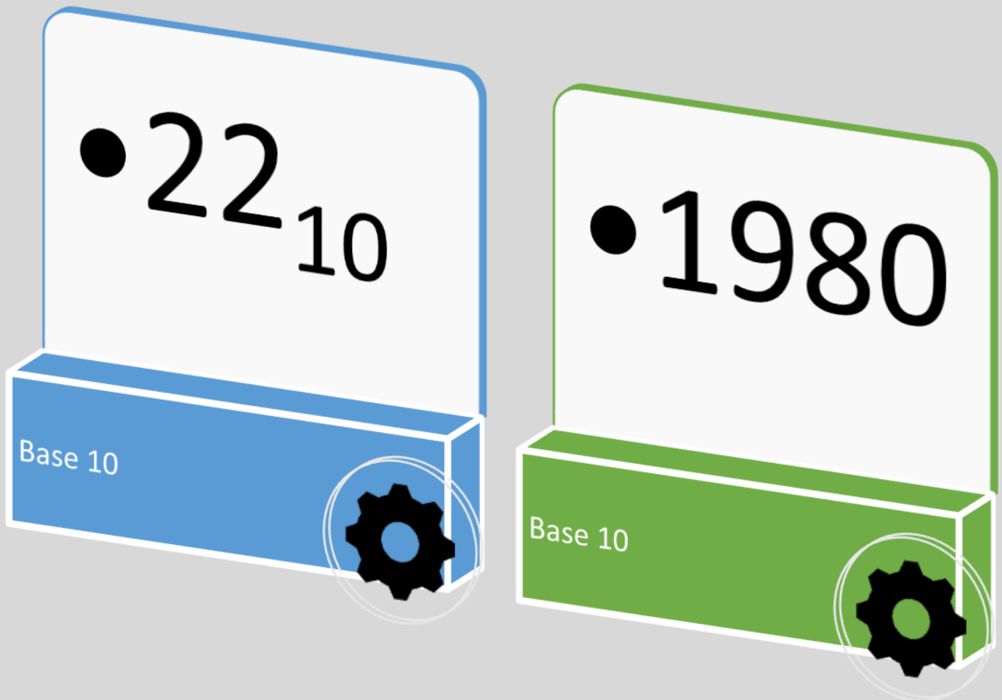


Figura 2.3. Representación de un número decimal con indicación de su base y un número decimal sin indicación de su base
Fuente: Aguilera y Bejarano, 2018

La forma de interpretar un valor en el sistema decimal es analizando la posición de cada elemento de un número. Partiendo de derecha a izquierda se ubican primero las unidades, las cuales están determinadas por la exponenciación 10^0 (1); luego, se ubican las decenas, las cuales emplean la potencia 10^1 (10); un espacio más a la izquierda están las centenas, que utilizan la potencia 10^2 (100) y así sucesivamente. El resultado de cada una de estas potencias se multiplica por el valor ubicado en su misma posición. Finalmente, los productos obtenidos se suman para formar el número original. A este método se le conoce como **suma de productos**.

Por ejemplo, si tomamos el número **25 980** el análisis de su descomposición tendría los siguientes pasos:

1. Sobre cada elemento del número, y de derecha a izquierda, colocamos los nombres de los valores relativos a la posición.

Decenas de millar	Unidades de millar	Centenas	Decenas	Unidades	← Valores relativos
2	5	9	8	0	← Número

2. Debajo de cada elemento del número, y de derecha a izquierda, se colocan las potencias y sus resultados.

10^4	10^3	10^2	10^1	10^0	← Potencias
10 000	1 000	100	10	1	← Resultados

3. Una vez que tenemos las potencias desarrolladas, multiplicamos cada elemento del número por el resultado de la potencia.

Decenas de millar	Unidades de millar	Centenas	Decenas	Unidades	
2	5	9	8	0	
10^4	10^3	10^2	10^1	10^0	Se multiplican
10 000	1 000	100	10	1	
20 000	5 000	900	80	0	← Resultados

4. El último paso es sumar los resultados obtenidos de las multiplicaciones, el número resultante es el mismo que teníamos al inicio (25 980).

Decenas de millar	Unidades de millar	Centenas	Decenas	Unidades	
2	5	9	8	0	
10^4	10^3	10^2	10^1	10^0	
10 000	1 000	100	10	1	
20 000	5 000	900	80	0	Se suman

$$20\,000 + 5\,000 + 900 + 80 + 0 = \boxed{25\,980}$$

Resultado

1.1.2 Sistema numérico binario

La base del sistema numérico binario es 2 y está compuesto de los dígitos 0 y 1. Un número expresado en el sistema binario se compone de una secuencia de unos y ceros y se emplea el número dos en el subíndice para indicar que se refiere a una cantidad en binario y no un número en decimal, por ejemplo:

Figura 2.4. Representación de números en el sistema binario
Fuente: Aguilera y Bejarano, 2018

En la figura 2.4, el número representa una cantidad, pero para reconocerla de forma más sencilla tendríamos que convertirla a un sistema que nos sea familiar, es decir, el sistema decimal; la conversión entre sistemas numéricos la veremos a profundidad más adelante.

1.1.3 Sistema numérico octal

La base del sistema numérico octal es 8 y está compuesto de los dígitos 0, 1, 2, 3, 4, 5, 6, 7. Un número expresado en el sistema octal emplea el número ocho en subíndice para indicar que se refiere a una cantidad en dicho sistema. Tres ejemplos de estos números se presenta en la figura 2.5:

Figura 2.5. Representación de números en el sistema octal
Fuente: Aguilera y Bejarano, 2018

De igual forma que con el sistema binario, para reconocer el número del ejemplo anterior de forma más sencilla tendríamos que convertirlo al sistema decimal.

1.1.4 Sistema numérico hexadecimal

La base del sistema numérico hexadecimal es 16 y está compuesto de los dígitos 0, 1, 2, 3, 4, 5, 6, 7,8,9; y de las letras A, B, C, D, E, F. Un número expresado en el sistema hexadecimal emplea el número 16 en subíndice para indicar que se refiere a una cantidad en dicho sistema, como se ejemplifica en la figura 2.6:

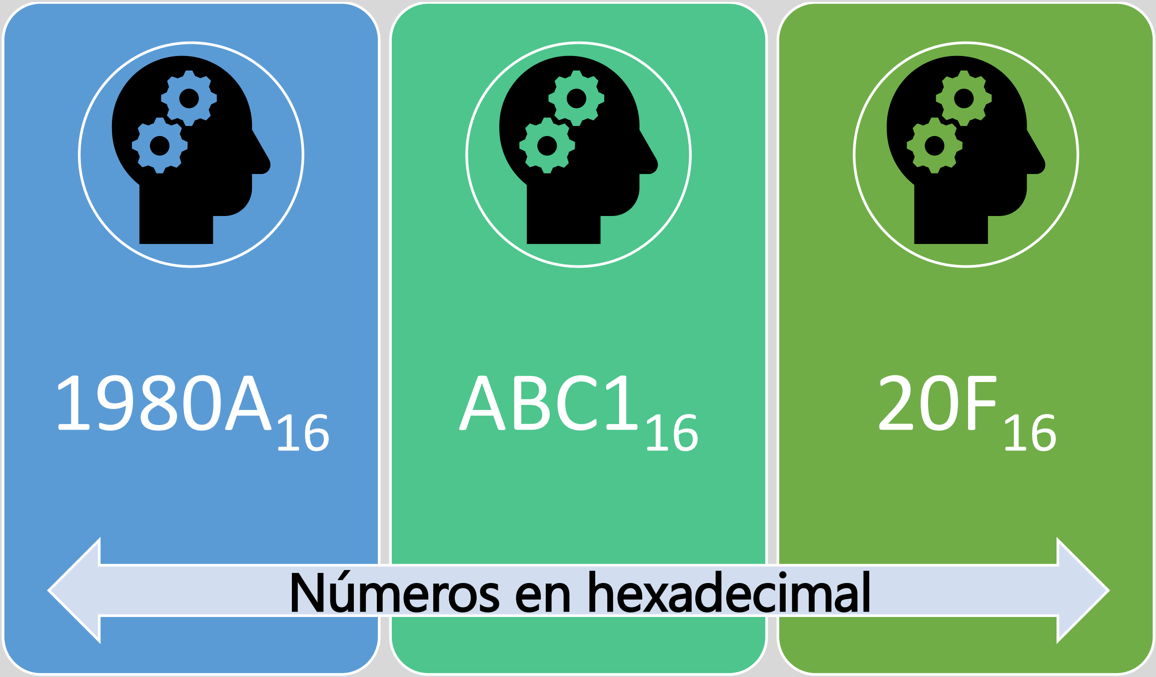


Figura 2.6. Representación de números en el sistema hexadecimal
Fuente: Aguilera y Bejarano, 2018

De igual forma que con los sistemas binario y octal, para reconocer el número del ejemplo anterior de forma más sencilla tendríamos que convertirlo al sistema decimal.

En la figura 2.7 se presenta un resumen de los sistemas numéricos vistos hasta el momento y sus componentes.

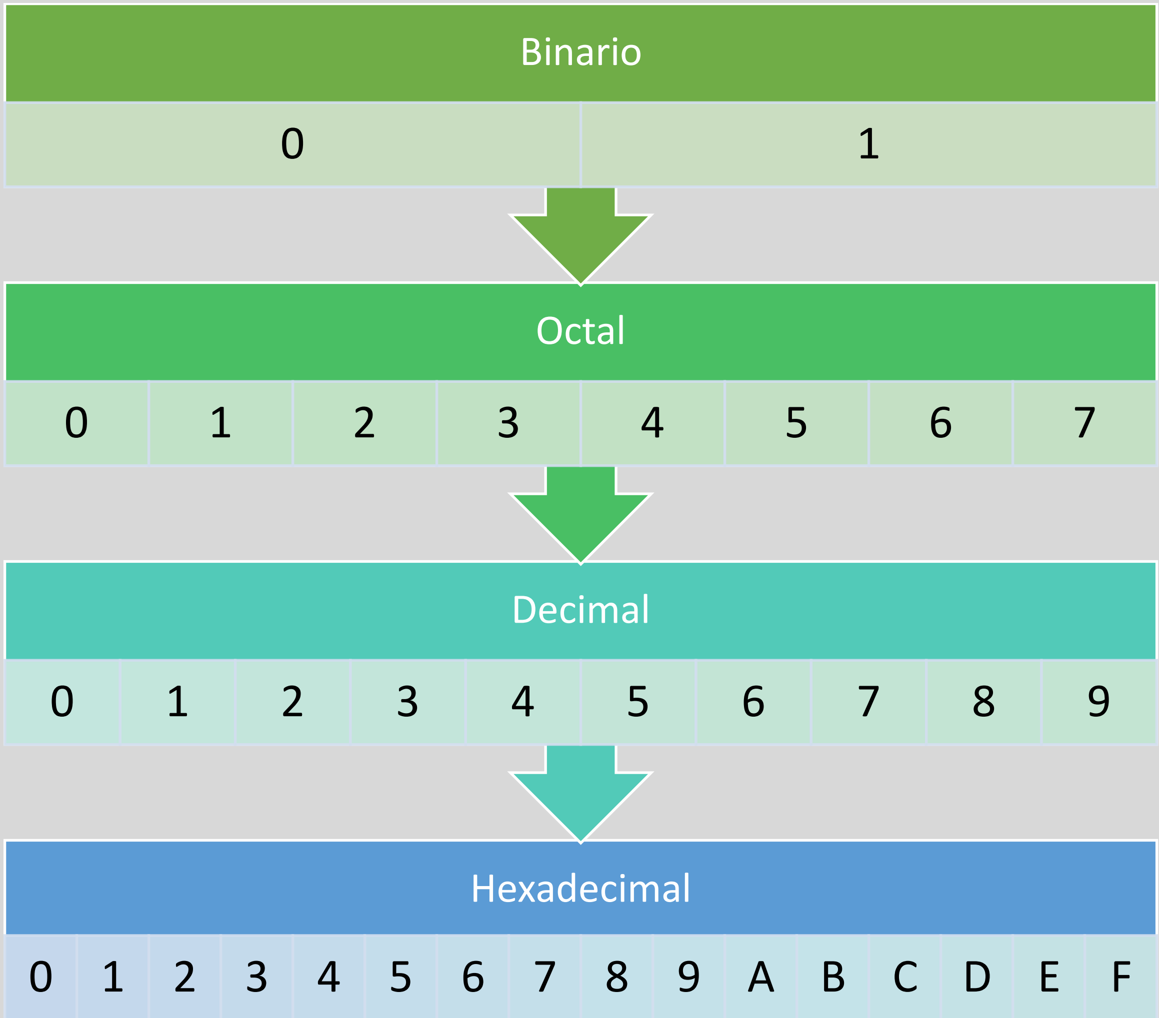


Figura 2.7. Componentes de los sistemas numéricos binario, octal, decimal y hexadecimal
Fuente: Aguilera y Bejarano, 2018

1.2 Conversión entre sistemas numéricos (Sistema N a Decimal)

El método que se utilizó para descomponer un número en el sistema decimal es el mismo que se debe emplear para una cantidad expresada en los otros sistemas, con la diferencia de la base. Por ende, en el sistema binario la base es 2, en el octal la base es 8 y en el hexadecimal la base es 16. Este es el primer paso para convertir un número desde cualquier sistema numérico al sistema decimal y se denomina suma de productos.

1.2.1 Conversión de binario a decimal

Para conocer el equivalente de un número binario en el sistema decimal, se empleará el método de **suma de productos**, el cual empleamos para descomponer un número en el sistema decimal y que se explicó al inicio de este capítulo, con la diferencia de que la base no es 10 sino que es 2.

Para el ejemplo tomaremos el número binario **111001**$_2$ y realizaremos los siguientes pasos:

1. Organizar el número (**111001**$_2$) de acuerdo con los pesos y ubicación de sus elementos.

Centenas de millar	Decenas de millar	Unidades de millar	Centenas	Decenas	Unidades
1	1	1	0	0	1

Valores relativos

Número

2. Debajo de cada elemento del número, y de derecha a izquierda, se colocan las potencias de base 2 y sus resultados respectivos.

2^5	2^4	2^3	2^2	2^1	2^0
32	16	8	4	2	1

Potencias

Resultados

3. Una vez que se tienen las potencias desarrolladas, se multiplica cada elemento del número por el resultado de la potencia, este paso es bastante sencillo ya que la multiplicación solo puede ser por 0 o 1.

Centenas de millar	Decenas de millar	Unidades de millar	Centenas	Decenas	Unidades
1	1	1	0	0	1

2^5	2^4	2^3	2^2	2^1	2^0
32	16	8	4	2	1
32	16	8	0	0	1

Se multiplican

Resultados

4. Por último, se suman los resultados obtenidos de las multiplicaciones, el número resultante es el equivalente en el sistema decimal del número en binario (**111001**$_2$).

Centenas de millar	Decenas de millar	Unidades de millar	Centenas	Decenas	Unidades
1	1	1	0	0	1

2^5	2^4	2^3	2^2	2^1	2^0
32	16	8	0	0	1
32	16	8	0	0	1
32 + 16 + 8 + 0+ 0+ 1 = 57					

Se suman

Resultado

Es decir, el número en binario **111001**$_2$ es el **57** en decimal. Es importante recalcar que este método se utiliza de forma muy similar para los otros sistemas numéricos (octal y hexadecimal), lo único que cambia es la base, pero la mecánica y cantidad de pasos es la misma.

1.2.2 Conversión octal a decimal

Para conocer el equivalente de un número octal en el sistema decimal, se empleará el método de **suma de productos**, ese es el mismo método empleado para la conversión de binario a decimal, pero en este caso se utilizará la base 8 en vez de la base 2.

Para el ejemplo tomaremos el número octal **253710_8** y seguiremos los pasos presentados a continuación:

1. Organizar el número (**253710_8**) de acuerdo con los pesos y ubicación de sus elementos.

Centenas de millar	Decenas de millar	Unidades de millar	Centenas	Decenas	Unidades
2	5	3	7	1	0

Valores relativos

Número

2. Debajo de cada elemento del número, y de derecha a izquierda, se colocan las potencias de base 8 y sus resultados respectivos.

8^5	8^4	8^3	8^2	8^1	8^0
32768	4096	512	64	8	1

Potencias

Resultados

3. Una vez que se tienen las potencias desarrolladas, se multiplica cada elemento del número por el resultado de la potencia.

Centenas de millar	Decenas de millar	Unidades de millar	Centenas	Decenas	Unidades
2	5	3	7	1	0

8^5	8^4	8^3	8^2	8^1	8^0
32768	4096	512	64	8	1
655	204	153	448	8	0

Se multiplican

Resultados

4. Por último, se suman los resultados obtenidos de las multiplicaciones, el número resultante es el equivalente en el sistema decimal del número en octal (**253710_8**).

Centenas de millar	Decenas de millar	Unidades de millar	Centenas	Decenas	Unidades
2	5	3	7	1	0

8^5	8^4	8^3	8^2	8^1	8^0
32 768	4 096	512	64	8	1
65 536	20 480	1 536	448	8	0
65 536 + 20 480 + 1 536 + 448 + 8 + 0 = 88 008					

Se suman

Resultado

Es decir, el número en octal **253710_8** es el **88 008** en decimal.

1.2.3 Conversión hexadecimal a decimal

Para conocer el equivalente de un número hexadecimal en el sistema decimal, se empleará el método de **suma de productos**, utilizado para las conversiones anteriores, con la diferencia de que en este caso emplearemos la base 16.

Para el ejemplo tomaremos el número hexadecimal **$5A3C71_{16}$** y seguiremos los pasos presentados a continuación:

1. Organizar el número (**$5A3C71_{16}$**) de acuerdo con los pesos y ubicación de sus elementos.

Decenas de millar	Unidades de millar	Centenas	Decenas	Unidades
A	3	C	7	1

Valores relativos

Número

2. Debajo de cada elemento del número, y de derecha a izquierda, se colocan las potencias de base 16 y sus resultados respectivos.

16^4	16^3	16^2	16^1	16^0
65536	4096	256	16	1

Potencias

Resultados

Una vez que se tienen las potencias desarrolladas, se multiplica cada elemento del número por el resultado de la potencia.

Compete aclarar que para los elementos que son una letra se utiliza el número equivalente en el sistema decimal de acuerdo a su posición en el sistema hexadecimal, por ejemplo:

- ✓ La **A** equivale a **10.**
- ✓ La **B** equivale a**11.**
- ✓ La **C** equivale a**12.**
- ✓ La **D** equivale a**13.**
- ✓ La **E** equivale a**14.**
- ✓ La **F** equivale a**15.**

Decenas de millar	Unidades de millar	Centenas	Decenas	Unidades
10	3	12	7	1
16^4	16^3	16^2	16^1	16^0
65536	4096	256	16	1
65	12	30	11	1

Se multiplican

Resultados

3. Por último, se suman los resultados obtenidos de las multiplicaciones, el número resultante es el equivalente en el sistema decimal del número en hexadecimal (**A3C71**$_{16}$).

Decenas de millar	Unidades de millar	Centenas	Decenas	Unidades
10	3	12	7	1

16^4	16^3	16^2	16^1	16^0
65 536	4 096	256	16	1
655 360	12 288	3 072	112	1
655 360 + 12 288 + 3 072 + 112 + 1 = 670 833				

Se suman

Resultado

Es decir, el número en hexadecimal **A3C71**$_{16}$ es el **670 833** en decimal.

1.3 Conversión entre sistemas numéricos (Decimal a Sistema N)

Para convertir un número decimal a cualquier sistema numérico se empleará el método de divisiones sucesivas. Este método consiste en tomar la cantidad en decimal y dividirla, de forma entera, entre la base del sistema al que se convertirá, el cociente resultante de esa operación se vuelve a dividir y así sucesivamente hasta que el cociente sea menor al divisor (base del sistema).

El orden del proceso se representa en el diagrama de la figura 2.8:

Dividendo	**÷**	**Divisor (base)**
-(Cociente*Divisor)		**Cociente**
Residuo1		

Cociente	**÷**	**Divisor (base)**
-(Cociente*Divisor)		**Cociente**
Residuo2		

Figura 2.8. Diagrama del proceso de conversión de sistema numérico decimal a un sistema numérico N
Fuente: Aguilera y Bejarano, 2018

El nuevo número se vería de esta forma: **Cociente Residuo2 Residuo1** y se forma tomando el cociente de la última división y los residuos de las demás divisiones, es decir, de atrás hacia adelante. Por ende, el primer dígito del nuevo número será el último cociente, el segundo dígito será el último residuo, el tercer dígito será el penúltimo residuo, así sucesivamente.

1.3.1 Conversión decimal a binario

Para convertir un número de decimal a binario dividimos ese número entre 2, hasta que el cociente sea menor a 2. Por ejemplo, el número 10 en decimal es el 1010_2 en binario. Cada 1 y 0 del número binario se formó de los residuos de dividir sucesivamente el 10 entre 2, como se explica en la figura 2.9:

Figura 2.9. Proceso de conversión de un número decimal a binario
Fuente: Aguilera y Bejarano, 2018

Notemos que en la primera división el residuo es 0 y el cociente 5. Debido a que 5 es mayor al divisor (2), hacemos otra división, como se muestra en la División 2, donde el residuo es 1 y el cociente 2. Ese nuevo cociente (2) es igual al divisor, por lo que se realiza una tercera división de 2 entre 2. En esta ocasión el residuo es 0 y el cociente 1. Como el cociente obtenido en la última división (División 3) es menor al divisor (1 es menor a 2) terminamos el proceso de divisiones y nos disponemos a formar el número en binario. Tomamos el último cociente (1) y luego todos los residuos del último al primero (010), de esta forma el número en binario será: 1010_2.

A continuación, podemos ver el proceso para el número 42 y representado de forma diferente, pero con el mismo método (divisiones sucesivas):

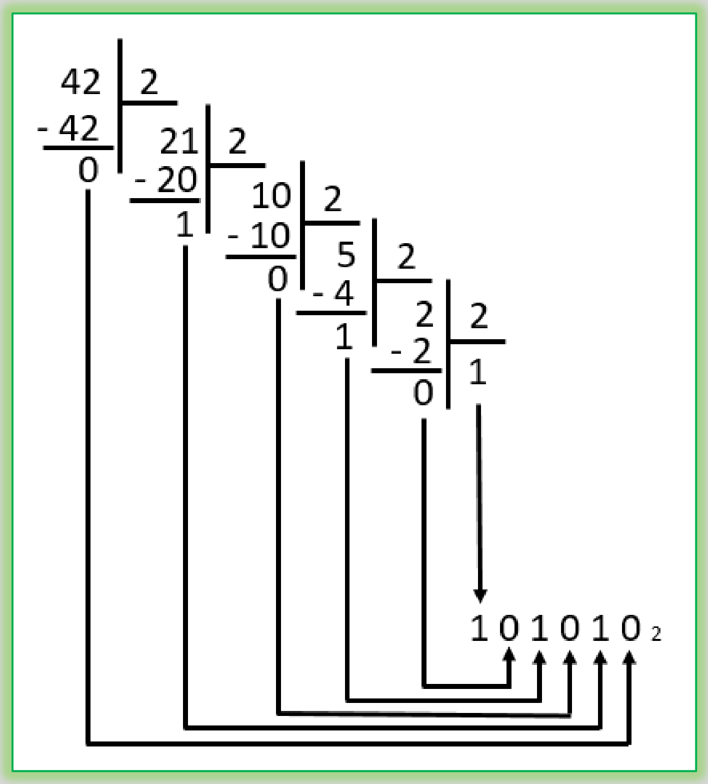


Figura 2.10. Proceso de conversión del número 10 en decimal a su representación en binario
Fuente: Aguilera y Bejarano, 2018

1.3.2 Conversión decimal a octal

La conversión de un número en decimal a uno del sistema octal se hace empleando el mismo método que para la conversión a binario, la diferencia es que en vez de dividir entre 2 se divide entre 8, por ejemplo, el número 80 en decimal es el 120_8 en el sistema octal, como podemos observar en el desarrollo de la figura 2.11:

Figura 2.11. Proceso de conversión del número 80 en decimal a su representación en octal
Fuente: Aguilera y Bejarano, 2018

Al analizar el caso anterior se puede observar que en la División 1 el residuo es 0 y el cociente 10, como este es mayor al divisor (8) se hace la División 2; en esta el residuo es 2 y el cociente es 1, este último cociente es menor al divisor por lo que no se hacen más divisiones y se forma el número en octal partiendo de este cociente, es decir, 120, recordemos que siempre debemos indicar la base del nuevo número, por lo que quedaría así: 120_8.

Veamos otro ejemplo, pero con el número 250 en decimal (figura 2.12):

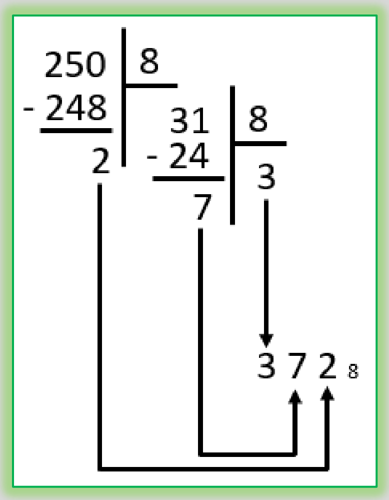


Figura 2.12. Proceso de conversión del número 250 en decimal a su representación en octal
Fuente: Aguilera y Bejarano, 2018

1.3.3 Conversión decimal a hexadecimal

Al igual que en los dos casos anteriores, la conversión de un número en decimal a uno del sistema hexadecimal se hace empleando el método de divisiones sucesivas, pero dividiendo entre 16, ya esta es la base del sistema hexadecimal, por ejemplo, el número 500 en decimal es el $1F4_{16}$ en hexadecimal, observemos el siguiente desarrollo:

Figura 2.13. Proceso de conversión del número 50 en decimal a su representación en hexadecimal
Fuente: Aguilera y Bejarano, 2018

En el análisis del ejemplo anterior podemos observar que el primer residuo es 4 y el cociente es 31. Este se puede seguir dividiendo entre 16, por lo que se efectúa la División 2; en esta, se obtiene un residuo de 15 y un cociente de 1. Es en este punto donde se termina el proceso, ya que 1 es menor a 16.

El nuevo número inicia con el último cociente (1) y los dos residuos (15 y 4). Recordemos que en el sistema hexadecimal los números que van del 10 al 15 se representan con la letra respectiva (A, B, C, D, E, F), por ende, en nuestro ejemplo el residuo 15 pasa a ser la letra F. Con esto obtenemos el número $1F4_{16}$.

Veamos otro ejemplo:

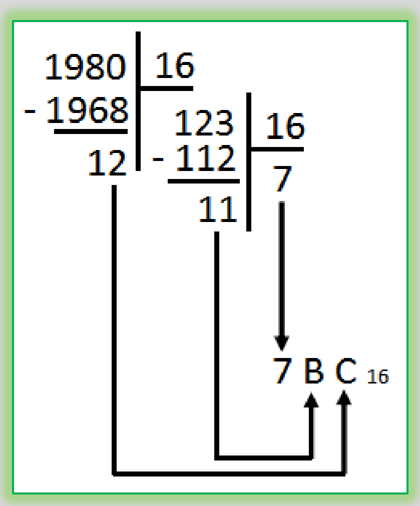


Figura 2.14. Proceso de conversión del número 1980 en decimal a su representación en hexadecimal
Fuente: Aguilera y Bejarano, 2018

1.3.4 Conversión de binario a octal

Para la conversión de un número en binario al sistema octal seguiremos un procedimiento muy sencillo de agrupamiento y sustitución. El método consiste en tomar el número en binario de derecha a izquierda y hacer grupos de tres elementos, si al final nos hacen falta elementos para completar el trío podemos agregar ceros a la izquierda. Una vez agrupados los dígitos convertiremos cada trío a su equivalente en el sistema **decimal**. El número formado por cada grupo será el nuevo número en octal.

Veamos un ejemplo en la figura 2.15, donde convertiremos el número 1011101010_2 a octal:

Pasos	Números			
1. Agrupamiento en tríos (3)	001	011	101	010
2. Equivalente en el sistema decimal	1	3	5	2
3. Nuevo número en octal	1352_8			

Figura 2.15. Proceso de conversión del número 1011101010_2 en binario a su representación en octal
Fuente: Aguilera y Bejarano, 2018

1.3.5 Conversión de binario a hexadecimal

Para la conversión de un número en binario al sistema hexadecimal seguiremos el mismo procedimiento que con el sistema octal, con la única diferencia de que los agrupamientos no son en tríos sino en cuartetos. De igual forma, se toma el número en binario de derecha a izquierda y hacer grupos de cuatro elementos, si al final nos hacen falta elementos para completar el cuarteto agregaremos ceros a la izquierda. Una vez agrupados los dígitos, convertiremos cada uno a su equivalente en el sistema **decimal**, el número formado por cada grupo será el nuevo número en hexadecimal. Recordemos que si el número formado por el cuarteto va del 10 al 15 se sustituirá por una letra al expresarlo en hexadecimal.

Veamos un ejemplo, convertiremos el número 11111101100010_2 a hexadecimal:

Pasos	Números			
1. Agrupamiento en cuartetos (4)	0011	1111	0110	0010
2. Equivalente en el sistema decimal	3	15 (F)	6	2

3. Nuevo número en hexadecimal	$3F62_{16}$

Figura 2.16. Proceso de conversión del número 11111101100010_2 en binario a su representación en hexadecimal
Fuente: Aguilera y Bejarano, 2018

1.3.6 Conversión de octal a binario

Para convertir un número expresado en octal al sistema binario también usaremos la creación de grupos, en este caso, serán tríos. Lo que debemos hacer es tomar el número en octal, de derecha a izquierda, y expresar en formato de tres elementos su equivalente en el sistema binario. Por ejemplo, si el número en octal es el 25_8 el equivalente en binario serían los tríos del 5 (101) y del 2 (010); así; el número en binario será 010101_2.

Analicemos un ejemplo para el número 4215_8:

Pasos	Números			
1. Elementos del número en octal	4	2	1	5
2. Equivalente en el sistema binario en tríos	100	010	001	101
3. Nuevo número en binario	100010001101_2			

Figura 2.17. Proceso de conversión del número 25_8 en octal a su representación en binario
Fuente: Aguilera y Bejarano, 2018

1.3.7 Conversión de hexadecimal a binario

Para convertir un número expresado en hexadecimal al sistema binario, utilizaremos el mismo método que para la conversión a octal, pero con la diferencia de que los grupos creados son cuartetos. Lo que debemos hacer es tomar el número en hexadecimal, de derecha a izquierda, y expresar en formato de cuatro elementos su equivalente en el sistema binario. Por ejemplo, si el número en hexadecimal es $A67_{16}$ el equivalente en binario serían los cuartetos del 7 (0111); del 6 (0110); y del A, que es el equivalente a 10_{16} (diez) y que en binario se representa como 1010; de esta forma el número en binario será el 101001100111_2.

Analicemos un ejemplo para el número $D2C8_{16}$ (figura 2.19):

Pasos	Números			
1. Elementos del número en hexadecimal	D (13)	2	C (12)	8

2. Equivalente en el sistema binario en cuartetos	1101	0010	1100	1000
3. Nuevo número en binario	1101001011001000_2			

Figura 2.19. Proceso de conversión del número $D2C8_{16}$ en hexadecimal a su representación en binario
Fuente: Aguilera y Bejarano, 2018

1.3.8 Ejercicios

Convierta el número dado al sistema expresado entre paréntesis.

1. 9 (Binario)
2. 12 (Octal)
3. 1974 (Hexadecimal)
4. 1011001_2 (Decimal)
5. 312_8 (Decimal)
6. $3A_{16}$ (Decimal)
7. 101010_2 (Octal)
8. $F10A_{16}$ (Binario)
9. 175_8 (Binario)
10. 11001001010_2 (Hexadecimal)

1.3.9 Respuestas

1. 9 (Binario)= 1001_2
2. 12 (Octal) = 14_8
3. 1974 (Hexadecimal) = $7B6_{16}$
4. 1011001_2 (Decimal) = 89
5. 312_8 (Decimal) = 202
6. $3A_{16}$ (Decimal) = 58
7. 101010_2 (Octal) = 52_8
8. $F10A_{16}$ (Binario) = 1111000100001010_2
9. 175_8 (Binario) = 001111101_2
10. 11001001010_2 (Hexadecimal) = $64A_{16}$

1.4 Operadores

Los operadores son símbolos que representan acciones dentro de una expresión. Se clasifican de acuerdo con categorías y dentro de dichas categorías existe una jerarquía que debe ser respetada al momento de resolver una expresión, a saber:

- Operadores de asignación.
- Operadores aritméticos.
- Operadores relacionales o comparativos.
- Operadores lógicos.

1.4.1 Operadores de asignación

La función de los operadores de asignación es brindar un valor. En programación se emplea una estructuras llamadas variables para almacenar valores, esto se estudiará con detalle más adelante en el capítulo 3. El símbolo utilizado para los operadores de asignación son el igual (=) o la flecha hacia la izquierda (←).

Si se desea asignar un valor de 5 a una variable llamada `PrimerNumero` se puede hacer de una de las siguientes formas:

```
PrimerNumero = 5

PrimerNumero ← 5
```

Figura 2.20. Formas de representar un valor a una variable
Fuente: Aguilera y Bejarano, 2018

Y se lee de la siguiente forma: a `PrimerNumero` asignar el valor de `5`. Esto significa que la variable `PrimerNumero` tendrá un valor de `5` de esa instrucción en adelante y se mantendrá así hasta que se le asigne otro valor.

1.4.2 Operadores aritméticos

Los operadores aritméticos son los que permiten realizar alguna operación aritmética entre valores y esto da como resultado un nuevo valor. Los operadores aritméticos son los siguientes:

Cuadro 2.1. Operadores aritméticos, su símbolo y forma de uso

Nombre	Símbolo	Sintaxis
Exponenciación	^	3 ^ 2
Multiplicación	*	3 * 2
División	/	3 / 2
Módulo	% o también MOD	3 % 2 o 3 MOD 2
Suma	+	3 + 2
Resta	-	3 - 2

El operador MOD es poco conocido y funciona de forma similar a una división, pero con la diferencia de que el resultado que brinda no es el cociente sino el **residuo**. Por ejemplo, al resolver la expresión **5 % 2** el resultado es **1**, como se muestra en la figura 2.21:

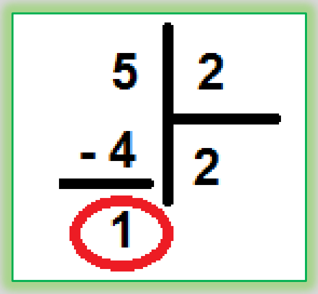


Figura 2.21. Operación matemática que explica el funcionamiento del operador MOD. El resultado que devuelve este operador es el residuo de la división, en este caso 1.
Fuente: Aguilera y Bejarano, 2018

Como podemos observar al hacer la división (entera, sin decimales), el cociente es 2 y nuestro residuo es 1, es decir, 5 entre dos es igual a 2 y sobra 1, ese 1 es el resultado del MOD.

Otro ejemplo se muestra en la figura 2.22, con la operación 13 MOD 5 = 3

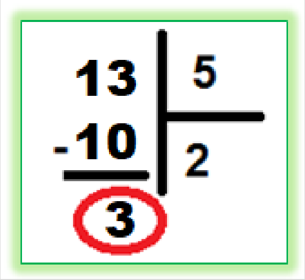


Figura 2.22. Operación matemática que explica el funcionamiento de aplicar el operador `MOD` a `13 y 5.` El resultado que devuelve este operador es el residuo de la división, en este caso `3`.
Fuente: Aguilera y Bejarano, 2018

Al analizar el ejemplo anterior, vemos que al dividir 13 entre 5 el cociente es dos, mientras que el residuo es 3.

3.1.3 Ejercicios

Resuelva las siguientes expresiones.

1. 42 MOD 5
2. 5 ^ 2
3. 101 + 10
4. 5 – 8
5. -2 + 5
6. 5 * 3
7. 150 / 3
8. 23 % 2
9. 9 - $^{-}2$
10. 2 + 1980

3.1.4 Respuestas

1. 2
2. 25
3. 111
4. $^{-}3$
5. 3
6. 15
7. 50
8. 1
9. 11
10. 1982

3.1.5 Precedencia de operadores aritméticos

La precedencia de los operadores aritméticos se muestra en el cuadro 2.2:

Cuadro 2.2. Precedencia de los operadores aritméticos

Operador	Símbolo
Exponenciación	^
Negatividad	-
Multiplicación, división y módulo	* / %
Suma y resta	+ -

Esto quiere decir que en una expresión que tenga únicamente operadores aritméticos y no contenga paréntesis se deben resolver primero los exponentes, luego, aplicar los operadores de negatividad, posteriormente, se resuelven las multiplicaciones y divisiones y, por último, se realizan las sumas y restas. En la figura 2.22 podemos analizar un ejemplo:

Paso	Explicación
2 * 2^3 + 15 / 3 - ¯5	Primero, resolvemos la potenciación 2^3, esto es igual que multiplicar 2*2*2 = 8
2 * 8 + 15 / 3 - ¯5	Luego, resolvemos el signo menos (-), por la ley de signos tenemos que menos y menos es igual a más, por ende, el - ¯5 pasa a ser un +5.
2 * 8 + 15 / 3 + 5	En este paso solo nos queda resolver las **multiplicaciones**

	y divisiones. Esto se puede hacer a como las vayamos encontrando de izquierda a derecha, en este caso resolveremos primero la multiplicación.
16 + 15 / 3 + 5	Seguidamente, resolvemos la **división** que nos queda.
16 + 5 + 5	Finalmente, resolvemos las **sumas** que nos quedaron.
26	El resultado definitivo es **26**.

Figura 2.22. Resolución paso a paso de una expresión con operadores aritméticos siguiendo el orden de precedencia de estos
Fuente: Aguilera y Bejarano, 2018

Analicemos otro ejemplo en la figura 2.23:

Paso	Explicación
7^0 - 10 % 4 + $^{-}$3 * 2 + 1	Primero, resolvemos la potenciación 7^0. Recordemos que todo número elevado a la cero (0) es igual a 1, por ende, 7^0 = 1.
1 - 10 % 4 + $^{-}$3	Luego, resolvemos el operador módulo (%). El resultado de 10

* 2 + 1	% 4 es **2**. Recordemos que el MOD lo que regresa es el **residuo** de una división entera.
1 - 2 + -3 * 2 + 1	Posteriormente, procesamos la multiplicación, cuyo resultado es **-6**, debido a que estamos multiplicando **factores con signos diferentes**.
1 - 2 + -6 + 1	Después, procedemos a resolver las sumas y restas de izquierda a derecha. Esto se puede hacer en un solo paso, pero lo desglosaremos, primero la resta **1 – 2**, la cual es igual a **-1.**
-1 + -6 + 1	Luego, ejecutamos la suma que en este caso es de dos números negativos, por lo que el resultado será negativo (**-7**)
-7 + 1	Finalmente, resolvemos la última suma, la cual tiene un

	resultado **negativo** ya que su elemento **mayor** es el número negativo.
-6	El resultado definitivo es **-6**.

Figura 2.23. Resolución paso a paso de una expresión con operadores aritméticos siguiendo el orden de precedencia de estos
Fuente: Aguilera y Bejarano, 2018

Y un último ejemplo en la figura 2.24:

Paso	Explicación
25/1 - 15/3 + $^{-}$6*2 + 15*0 /-1*2	Primero, resolvemos las divisiones y multiplicaciones que nos encontremos de izquierda a derecha.
25 – 5 + $^{-}$12 + 0 / -2	Aún nos queda una división, la cual debemos realizar antes de las sumas y restas.
25 – 5 + $^{-}$12 + 0	Ahora, resolveremos las sumas y restas de izquierda a derecha.
20 + $^{-}$12	Finalmente, resolvemos la **suma** que nos queda.
8	El resultado definitivo es **8**.

Figura 2.24. Resolución paso a paso de una expresión con operadores aritméticos siguiendo el orden de precedencia de estos
Fuente: Aguilera y Bejarano, 2018

3.1.6 Ejercicios

Resuelva cada una de las expresiones presentadas a continuación, elabore el desarrollo de cada expresión para llegar a su respuesta, tal y como se hizo en los ejemplos.

1. 8 + 5 * 1 / 5 + 1

2. 50 / 5 - 30 + 2 ^ 2

3. 3 * 4 / 2 – 1 * 3 * 4 + 18 / 2 – 3 * 8 + 5 * 2^3

4. 5 - 15 * 3 + 15 - ¯1 * 23 * 6 + 2 – 1

5. 2 * 3 * 5 – 10 / 2 + 30^2 - 155^0

6. 1 + 1 + 3 - 5 + 6 + ¯4 * 4 / 2 - 2 * 2 + 24 * 3^3

7. 45 – ¯20 – 5 * ¯2^2 + 35 – 45 / 5 * 3 + 17 – 2

8. 10 / 2 + 10 - 14 + ¯5 * 2 ^ 3 / 8 - 5 + 2 * 2 + 11 * 0 – 11

9. 3 * 30 / 5 + 5 + 14 - 6 / 1 + ¯25 * 2 + 11 * 7 – 15 – 5 / 10

10. 6 – 4 * 4 / 2 + 2 * 2^4 / 2^3 – 10

3.1.7 Respuestas

1. 8 + 5 * 1 / 5 + 1

   8 + 5 * 1 / 5 + 1

   8 + 5 / 5 + 1

   8 + 1 + 1

   R/ 10

2. 50 / 5 - 30 + 2 ^ 2

   50 / 5 - 30 + 2 ^ 2

   50 / 5 - 30 + 4

10 - 30 + 4

-20 + 4

R/ -16

3. 3 * 4 / 2 – 1 * 3 * 4 + 18 / 2 – 3 * 8 + 5 * 2^3

   3 * 4 / 2 – 1 * 3 * 4 + 18 / 2 – 3 * 8 + 5 * $\underline{2\text{^}3}$

   $\underline{3 * 4}$ / 2 – $\underline{1 * 3 * 4}$ + $\underline{18 / 2}$ – $\underline{3 * 8}$ + $\underline{5 * 8}$

   $\underline{12 / 2}$ – 12 + 9 – 24 + 40

   $\underline{6 – 12 + 9 – 24 + 40}$

   R/ 19

4. 5 - 15 * 3 + 15 - $^{-}$1 * 23 * 6 + 2 – 1

   5 - 15 * 3 + 15 $\underline{- ^{-}1}$ * 23 * 6 + 2 – 1

   5 - $\underline{15 * 3}$ + 15 + $\underline{1 * 23 * 6}$ + 2 – 1

   $\underline{5 - 45}$ + 15 + 138 + 2 – 1

   $\underline{-40 + 15}$ + 138 + 2 – 1

   $\underline{-25 + 138 + 2 – 1}$

   R/ 114

5. 2 * 3 * 5 – 10 / 2 + 30^2 - 155^0

   2 * 3 * 5 – 10 / 2 + $\underline{30\text{^}2}$ - $\underline{155\text{^}0}$

   $\underline{2 * 3 * 5}$ – 10 / 2 + 900 - 1

   30 – $\underline{10 / 2}$ + 900 – 1

   $\underline{30 – 5 + 900 – 1}$

   R/ 924

6. 1 + 1 + 3 - 5 + 6 + $^{-}$4 * 4 / 2 - 2 * 2 + 24 * 3^3

   1 + 1 + 3 - 5 + 6 + $^{-}$4 * 4 / 2 - 2 * 2 + 24 * $\underline{3\text{^}3}$

1 + 1 + 3 - 5 + 6 + ¯4 * 4 / 2 - 2 * 2 + 24 * 27

1 + 1 + 3 - 5 + 6 + ¯16 / 2 - 4 + 648

1 + 1 + 3 - 5 + 6 + ¯8 - 4 + 648

5 - 5 + 6 + ¯8 - 4 + 648

6 + ¯8 - 4 + 648

R/ 642

7. 45 – ¯20 – 5 * ¯2^2 + 35 – 45 / 5 * 3 + 17 – 2

45 + 20 – 5 * 4 + 35 – 45 / 5 * 3 + 17 – 2

45 + 20 – 20 + 35 – 9 * 3 + 17 – 2

45 + 20 – 20 + 35 – 27 + 17 – 2

R/ 68

8. 10 / 2 + 10 - 14 + ¯5 * 2 ^ 3 / 8 - 5 + 2 * 2 + 11 * 0 – 11

10 / 2 + 10 - 14 + ¯5 * 2 ^ 3 / 8 - 5 + 2 * 2 + 11 * 0 – 11

10 / 2 + 10 - 14 + ¯5 * 8 / 8 - 5 + 2 * 2 + 11 * 0 – 11

5 + 10 - 14 + ¯40 / 8 - 5 + 4 + 0 – 11

5 + 10 - 14 + ¯5 - 5 + 4 + 0 – 11

1 + ¯5 - 5 + 4 + 0 – 11

¯9 + 4 + 0 – 11

¯5 – 11

R/ ¯16

9. 3 * 30 / 5 + 5 + 14 - 6 / 1 + ¯25 * 2 + 11 * 7 – 15 – 5 / 10

3 * 30 / 5 + 5 + 14 - 6 / 1 + ¯25 * 2 + 11 * 7 – 15 – 5 / 10

<u>90 / 5</u> + 5 + 14 - 6 + ⁻50 + 77 – 15 – <u>5 / 10</u>

<u>18 + 5 + 14 - 6</u> + ⁻50 + 77 – 15 – 0.5

<u>31 + ⁻50 + 77</u> – 15 – 0.5

<u>58 – 15 – 0.5</u>

R/ 42.5

10. 6 – 4 * 4 / 2 + 2 * 2^4 / 2^3 – 10

    6 – 4 * 4 / 2 + 2 * <u>2^4</u> / <u>2^3</u> – 10

    6 – <u>4 * 4</u> / 2 + <u>2 * 16</u> / 8 – 10

    6 – <u>16 / 2</u> + <u>32 / 8</u> – 10

    <u>6 – 8 + 4 – 10</u>

    R/ ⁻8

3.1.8 Operadores relacionales o comparativos

Estos operadores evalúan **valores** (ya sea numéricos o alfabéticos) y **expresiones**, y dan como resultado un valor lógico; es decir, **`verdadero`** (`true` o `T`) o **`falso`** (`false` o `F`).

Cuando los operadores comparativos trabajan con números simplemente se debe considerar el valor de dichos números en el sistema decimal, mientras que al evaluar valores alfabéticos (caracteres que pueden ser letras u otros símbolos) se debe conocer primero el valor numérico del carácter en el código ASCII, para luego hacer la comparación respectiva utilizando dicho valor numérico.

El código ASCII (*American Standard Code for Information Interchange*) es un estándar para la representación numérica de caracteres. En total, se pueden expresar 127 (0-126) elementos, entre los cuales tenemos letras (mayúsculas y minúsculas), símbolos no imprimibles y caracteres especiales como por ejemplo: espacio en blanco, el arroba (`@`), la coma (`,`); entre otros.

Para obtener un elemento del código ASCII se deben oprimir las teclas `Alt` y alguna combinación numérica válida, por ejemplo, `Alt` y el número `65` dan como resultado la letra `A`.

A continuación, se presenta el cuadro 2.3 con los elementos imprimibles del código ASCII y su equivalencia numérica:

Cuadro 2.3. Elementos del código ASCII y su combinación numérica

Número	Símbolo	Número	Símbolo	Número	Símbolo	Número	Símbolo	Número	Símbolo
33	!	55	7	77	M	99	c	121	y
34	"	56	8	78	N	100	d	122	z
35	#	57	9	79	O	101	e	123	{
36	$	58	:	80	P	102	f	124	|
37	%	59	;	81	Q	103	g	125	}
38	&	60	<	82	R	104	h	126	~
39	'	61	=	83	S	105	i		
40	(	62	>	84	T	106	j		
41	)	63	?	85	U	107	k		
42	*	64	@	86	V	108	l		
43	+	65	A	87	W	109	m		
44	,	66	B	88	X	110	n		
45	-	67	C	89	Y	111	o		
46	.	68	D	90	Z	112	p		
47	/	69	E	91	[	113	q		
48	0	70	F	92	\	114	r		
49	1	71	G	93	]	115	s		
50	2	72	H	94	^	116	t		
51	3	73	I	95	_	117	u		
52	4	74	J	96	`	118	v		
53	5	75	K	97	a	119	w		
54	6	76	L	98	b	120	x		

Fuente: recopilación de Aguilera y Bejarano, 2018

Los operadores relacionales se representan de la siguiente forma (cuadro 2.4):

Cuadro 2.4. Operadores relacionales o comparativos

Interpretación	Símbolo
Igual que	= o también ==
Mayor que	>
Mayor o igual que	>=
Menor que	<
Menor o igual que	<=
Diferente a	!= o también <>

Fuente: recopilación de Aguilera y Bejarano, 2018

En algunos lenguajes de programación se emplea el doble igual (==) como operador comparativo y un solo igual (=) como operador de asignación. También, utilizan el operador (!=) como diferente. Para nuestros ejemplos y ejercicios de expresiones aritméticas aplicaremos el uso de cualquiera de los símbolos de la tabla anterior, mientras que para los algoritmos usaremos el igual (=) para asignación y comparación; y el Diferente a (también conocido como No igual) (!=) para expresar diferencia.

Analicemos un ejemplo, en la figura 2.25, de cómo funciona un operador relacional Diferente a:

Paso	Explicación
5 != 5	Se evalúa si el 5 es diferente (no igual) a 5.
F(False, falso)	El resultado es F (false) ya que no es cierto que 5 sea diferente a 5, son iguales, por ende, la expresión es falsa.

Figura 2.25. Resolución paso a paso del uso del operador relacional Diferente a
Fuente: Aguilera y Bejarano, 2018

Veamos otro ejemplo en la figura 2.26, esta vez del operador Mayor o igual que

Paso	Explicación
37 >= 25	Se evalúa si el 37 es mayor o igual a 25.
T(True, verdadero)	El resultado es T (true) ya que el 37 es mayor a 25. Aunque no sea igual a 25, con solo cumplir alguna de las dos opciones (mayor o igual), el resultado será verdadero.

Figura 2.26. Resolución paso a paso del uso del operador relacional Mayor o igual que
Fuente: Aguilera y Bejarano, 2018

Estudiemos un último ejemplo (figura 2.27):

Paso	Explicación
22 < 22	Se evalúa si 22 es menor a 22.
F(False, falso)	El resultado es F (false) ya que 22 no es estrictamente menor a 22, son iguales, por ende, la expresión da como resultado un falso.

Figura 2.27. Resolución paso a paso del uso del operador relacional Menor que
Fuente: Aguilera y Bejarano, 2018

En los ejemplos anteriores se evaluaron valores numéricos, pero también se pueden resolver expresiones con valores alfabéticos, analicemos un par de ejemplos en las figuras 2.28 y 2.29:

Paso	Explicación

`"A" < "B"`	Lo primero que debemos hacer es asignar el valor del código ASCII para las letras `A` y `B`, los cuales son los siguientes: `65` para `A` y `66` para `B.` Por ende, la expresión verdadera sería **`65<66.`**
`T (True, Verdadero)`	El resultado es `T` (`true`) ya que `65` es menor a `66`, o dicho con el uso de las letras `A` es menor a `B`.

Figura 2.28. Resolución paso a paso del uso del operador relacional Menor que con valores alfabéticos
Fuente: Aguilera y Bejarano, 2018

Paso	Explicación
`"R" = "r"`	De nuevo, asignamos primero el valor del código

	ASCII para las letras `R` y `r`. Los cuales son los siguientes: `82` para `R` y `114` para `r`. Por ende, la expresión verdadera sería: **`82 = 114`**.
`F (False, Falso)`	El resultado es `F` (`false`) ya que `82` no es igual a `114` o, también, `R` no es igual a `r`.

Figura 2.29. Resolución paso a paso del uso del operador relacional Igual con valores alfabéticos
Fuente: Aguilera y Bejarano, 2018

Nota: en los ejemplos y ejercicios con letras se emplean las comillas dobles, pero esas comillas no se toman en cuenta para la resolución de la expresión.

Si tenemos una expresión con operadores aritméticos y relacionales, primero se resuelve la parte aritmética y luego la relacional. Revisemos un par de ejemplos en las figuras 2.30 y 2.31:

Paso	Explicación
`2 - 1 * 3 < 5`	Lo primero que debemos hacer es resolver las expresiones con los operadores aritmético. Para esto, debemos considerar el orden de precedencia de dichos operadores.

	Lo primero que resolveremos es la multiplicación.
`2 - 3 < 5`	Luego de la multiplicación, realizamos la resta que nos queda.
`-1 < 5`	Lo que obtenemos al final es una expresión con valores numéricos, solo nos queda resolver el operador comparativo.
T (True, Verdadero)	El resultado es `T` (`true`) ya que `-1` es menor a `5`.

Figura 2.30. Resolución paso a paso del uso de operadores aritméticos y relacionales en expresiones
Fuente: Aguilera y Bejarano, 2018

Paso	Explicación
2^3 – 8 * 1 > 5 + 6 / 12 + -6	Al igual que en el ejemplo anterior debemos resolver primero las expresiones con los operadores aritméticos, pero en este caso tenemos expresiones a ambos lados del operador relacional. Podemos elegir entre resolver primero alguna de las dos expresiones por completo o hacer pequeñas resoluciones en ambas expresiones al mismo tiempo. Para el ejemplo haremos

	pequeños pasos en ambas expresiones al mismo tiempo. Por la ley de precedencia resolveremos primero la potencia (en la expresión de la izquierda) y la división (en la expresión de a derecha).
8 – 8 * 1 > 5 + 0.5 + -6	Posteriormente, hacemos la multiplicación (expresión de la izquierda) y las sumas (expresión de la derecha).
8 – 8 > -0.5	Luego, resolvemos la resta de la expresión del lado izquierdo, en el lado derecho ya tenemos un solo valor numérico.
0 > -0.5	Por último, resolvemos la expresión con el operador comparativo.
`T (True, Verdadero)`	El resultado es `T` (`true`) ya que `0` es mayor a `-0.5`

Figura 2.31. Resolución paso a paso del uso de operadores aritméticos y relacionales en expresiones
Fuente: Aguilera y Bejarano, 2018

3.1.9 Ejercicios

Resuelva cada una de las expresiones presentadas a continuación, elabore el desarrollo de cada expresión para llegar a su respuesta, tal y como se hizo en los ejemplos.

1. 8 > 5 * 1

2. 50 / 5 < 30

3. 3 * 4 / 2^0 = 6

4. “A” >= “1”

5. 18 / 2 * $^{-}$1 < $^{-}$9 + $^{-}$1

6. 155^1 != 155

7. “X” != “x”

8. 23 * 2 + 2 – 1 <= 7 * 7

9. “3” <= “@”

10. 4 * 4 / 2 = 2^3

3.1.10 Respuestas

1. 8 > $\underline{5 * 1}$

   8 > 5

R/ T

2. 50 / 5 < 30

   10 < 30

   R/ T

3. 3 * 4 / 2^0 = 6

   3 * 4 / 1 = 6

   12 / 1 = 6

   12 = 6

   R/ F

4. "A" >= "1"

   65 >= 49

   R/ T

5. $18 / 2 * {}^{-}1 < {}^{-}9 + {}^{-}1$

   $9 * {}^{-}1 < {}^{-}10$

   ${}^{-}9 < {}^{-}10$

   R/ F

6. 155^1 != 155

   155 != 155

   R/ F

7. "X" != "x"

   88 != 120

   R/ T

8. 23 * 2 + 2 – 1 <= 7 * 7

   46 + 2 – 1 <= 49

   47 <= 49

   R/ T

9. “3” <= “@”

   51 <= 64

   R/ T

10. 4 * 4 / 2 = 2^3

    16 / 2 = 8

    8 = 8

    R/ T

3.1.11 Precedencia de operadores relacionales o comparativos

Los operadores relacionales son todos de la misma jerarquía, por ende, se resuelven de **izquierda a derecha**.

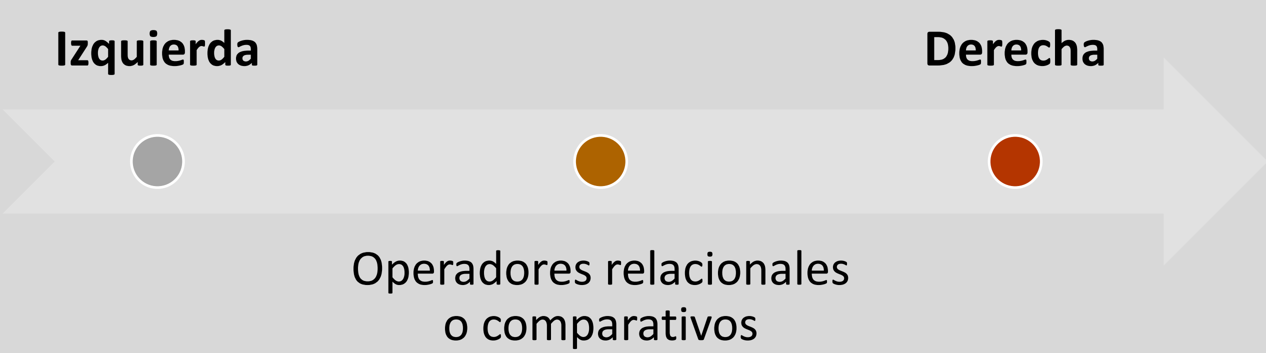


Figura 2.32. La precedencia de los operadores relacionales se establece de izquierda a derecha

Fuente: Aguilera y Bejarano, 2018

3.1.12 Operadores lógicos

Los operadores lógicos evalúan dos valores lógicos (que pueden ser `verdadero` o `falso`) y retornan un valor del mismo tipo, `verdadero` (`True, T`) o `falso` (`False, F`). El cuadro 2.5 presenta un resumen de los operadores lógicos:

Cuadro 2.5. Operadores lógicos, su representación y significado

Nombre		Símbolos		Definición
No	`Not`	`!`	¬	Negación
Y	`And`	`&&`	∧	Conjunción
O	`Or`	`||`	∨	Disyunción

Fuente: Recopilado por Aguilar y Bejarano, 2018

3.1.13 Negación (**NOT** o !)

El operador **NOT** recibe un valor y **retorna el valor inverso**, es decir, si recibe un `true` retorna un `false` y si recibe un `false` retorna un `true`.

El operador `NOT` se representa por medio de los siguientes símbolos, letras o palabras:

Cuadro 2.6. Representación del operador lógico Negación

Operador		Símbolo		Definición
NO	**NOT**	!	¬	Negación
Nota: Se recomienda utilizar el mismo símbolo durante todo el desarrollo de una expresión				

Analicemos un ejemplo del funcionamiento del operador `NOT`. Supongamos que se tiene la variable `A` con un valor de `true` (`A=T`) y se le aplica un `NOT` (`!A`), esto se resolvería de la siguiente forma:

Paso	Explicación
! A	Lo primero que debemos hacer es sustituir la letra `A` por el valor dado en la descripción del ejemplo.
! T	Luego de la sustitución, procedemos a aplicar la negación al valor lógico `true`.
F (False, Falso)	La negación o inverso de `true` es el `false`, por ende, el resultado de la expresión es `F`.

Figura 2.33. Ejemplo de aplicación del operador `NOT` a una variable `A`
Fuente: Aguilera y Bejarano, 2018

Veamos otros ejemplos, en este caso usaremos la misma variable (`A`) con el mismo valor inicial (`T, true`), pero se negará dos veces:

Paso	Explicación
!!A	Lo primero que debemos hacer es sustituir la letra `A` por el valor dado en la descripción del ejemplo.
!!T	Luego de la sustitución, procedemos a aplicar la negación más próxima al valor lógico `true`.
!F	El resultado es un `false` (`F`), el siguiente paso es aplicar la otra negación a dicho `false`.
T (True, Verdadero)	El resultado final es un `true` (`T`).

Figura 2.34. Ejemplo de aplicación del operador `NOT` dos veces a una variable `A`
Fuente: Aguilera y Bejarano, 2018

3.1.14 Ejercicios

Resuelva cada una de las expresiones presentadas a continuación, elabore el desarrollo de cada expresión para llegar a su respuesta, tal y como se hizo en los ejemplos. Tome en cuenta los siguientes valores iniciales: `A=T (true)`, `B=F (false)`, `C=T (true)`

1. !!A

2. !B

3. !!!C

3.1.15 Respuestas

1. !!A

   !!T

   !F

   R/ T

2. !B

   !F

   R/ T

3. !!!C

   !!!T

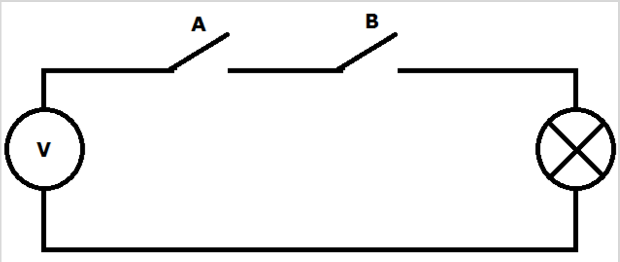
   !!F

   !T

   R/ F

3.1.16 Conjunción (`AND o &&`)

La conjunción recibe dos o más valores lógicos y retorna un solo valor lógico. Para comprender su funcionamiento se analizará un diagrama de circuito eléctrico en **serie**:



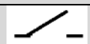
Simbología	
Símbolo	**Nombre**
	Interruptor
	Fuente de voltaje
	Bombillo

Figura 2.35. Diagrama de circuito eléctrico que representa la conjunción
Fuente: Aguilera y Bejarano, 2018

Los interruptores (`A` y `B`) tendrán dos estados, los cuales se analizarán como valores lógicos de la siguiente forma: activado (`True, T`) y desactivado (`False, F`). Asimismo, el bombillo tendrá la misma relación de estados, es decir, activado o prendido (`True, T`) y desactivado o apagado (`False, F`).

Se debe tener claro que si un interruptor está activado significa que permitiría el paso de electricidad proveniente de la fuente de voltaje y si el bombillo se activa es porque recibe la electricidad proveniente de la fuente de voltaje y habilitada por la activación de los interruptores.

A continuación, en el cuadro 2.7, se explicará el funcionamiento del operador `AND`, mediante el análisis de la tabla de verdad y el circuito eléctrico en serie de la figura 2.35.

Cuadro 2.7. Tabla de verdad del operador `And` aplicada al circuito del bombillo del diagrama 2.35

Entradas		Salida	Descripción
Interruptor A	Interruptor B	Estado del bombillo	
F	F	F	Si ambos interruptores tienen un valor de F entonces el bombillo estaría en F, ya que la electricidad no pasaría por ninguno de los dos interruptores.
F	T	F	Si se activa el interruptor B, pero se mantiene desactivado el A la electricidad no pasaría, por ende, el bombillo no se activará.
T	F	F	Si se hace lo opuesto al estado de entradas anterior, el bombillo seguiría sin activarse. Ya que la electricidad pasa por el interruptor A, pero no por el B.
T	T	T	Si se activan ambos interruptores, el bombillo se activaría. Únicamente si todos los interruptores tienen un valor de T el bombillo tendrá un valor de T.
Nota: Podemos observar que con solo un operador (interruptor) **AND** esté desactivado (**F**) el bombillo tendrá un estado de apagado (**F**).			

Fuente: Aguilera y Bejarano, 2018

El operador AND se representa por medio de los siguientes símbolos, letras o palabras:

Cuadro 2.8. Representación del operador AND

Operador	Símbolos	Definición
Y, AND	&&, Λ	Conjunción
Nota: Se recomienda utilizar el mismo símbolo durante todo el desarrollo de una expresión		

Fuente: Aguilera y Bejarano, 2018

Analicemos algunos ejemplos del funcionamiento del operador AND en las figuras 2.36 a la figura 2.38, tomemos en cuenta los siguientes valores: A=T, B=F, C=T

Paso	Explicación
A Λ B	Lo primero que debemos hacer es

	sustituir las letras `A` y `B` por el valor dado en la descripción del ejemplo.
T ʌ F	Luego de la sustitución, procedemos a aplicar el operador `AND` a los valores lógicos. Si lo resolvemos usando el circuito eléctrico, tendremos que un interruptor está activado y el otro no por lo que el valor del bombillo será apagado, es decir `F`.
F	Este sería el resultado definitivo `F`.

Figura 2.36. Ejemplo paso a paso del funcionamiento del operador `AND`
Fuente: Aguilera y Bejarano, 2018

Veamos otro ejemplo.

Paso	Explicación
C ʌ B ʌ A	Lo primero que debemos hacer es sustituir las letras por el valor dado en la descripción del ejemplo.

T Λ F Λ T	Luego de la sustitución, procedemos a resolver la primera pareja de elementos. Podemos concluir que el resultado de esa expresión es `F`, ya que uno de los interruptores tiene un valor `False` (`F`), por lo que el bombillo estaría apagado (`F`).
F Λ T	Ya con el resultado de la operación anterior (`F`), procedemos a resolver este resultado con el otro valor lógico.
F	Si analizamos la expresión con el ejemplo del circuito eléctrico, tendremos que un interruptor está activado mientras que el otro no, por ende, el bombillo, y la expresión, tendrán un valor de `F`.

Figura 2.37. Ejemplo paso a paso del funcionamiento del operador `AND` con tres variables
Fuente: Aguilera y Bejarano, 2018

Estudiemos un último ejemplo.

Paso	Explicación
C Λ A	Lo primero que debemos hacer es sustituir las letras por el valor dado en la descripción del ejemplo.
T Λ T	Luego de la sustitución, procedemos a aplicar el operador `AND`.
T	De acuerdo con el

	análisis del circuito eléctrico, ambos interruptores están activados, por lo que el valor del bombillo, y por ende de la expresión, es `T` (`true`).

Figura 2.38. Ejemplo paso a paso del funcionamiento del operador `AND`
Fuente: Aguilera y Bejarano, 2018

3.1.17 Ejercicios

Resuelva cada una de las expresiones presentadas a continuación, elabore el desarrollo de cada expresión para llegar a su respuesta, tal y como se hizo en los ejemplos. Tome en cuenta los siguientes valores iniciales: `A=T(true),B=F(false),C=F(false)`

1. A Λ B Λ B Λ A Λ C

2. B Λ B

3. C Λ C Λ A

4. C Λ A Λ B Λ A

5. A Λ A Λ C

3.1.18Respuestas

1. $\underline{A \wedge B \wedge B \wedge A \wedge C}$

   $\underline{T \wedge F} \wedge F \wedge T \wedge F$

   $\underline{F \wedge F} \wedge T \wedge F$

   $\underline{F \wedge T} \wedge F$

   $\underline{F \wedge F}$

   R/ F

2. $\underline{B \wedge B}$

   $\underline{F \wedge F}$

   R/ F

3. $\underline{C \wedge C \wedge A}$

   $\underline{F \wedge F} \wedge T$

   $\underline{F \wedge T}$

   R/ F

4. $\underline{C \wedge A \wedge B \wedge A}$

   $\underline{F \wedge T \wedge F \wedge T}$

   $\underline{F \wedge T} \wedge F \wedge T$

   $\underline{F \wedge F} \wedge T$

   $\underline{F \wedge T}$

   R/ F

5. $\underline{A \wedge A} \wedge C$

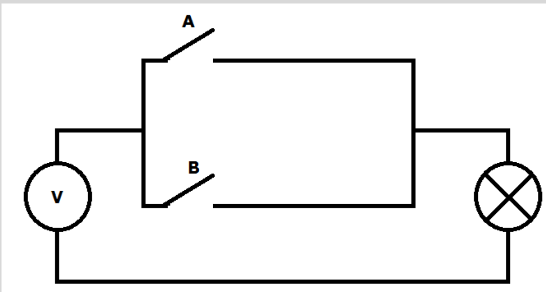
   $\underline{T \wedge T} \wedge F$

   $\underline{T \wedge F}$

R/ F

3.1.19 Disyunción (`OR` o `||`)

La disyunción recibe dos o más valores lógicos y retorna un solo valor lógico. Para comprender su funcionamiento se analizará un diagrama de circuito eléctrico en **paralelo**:



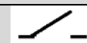
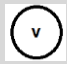
Simbología	
Símbolo	**Nombre**
	Interruptor
	Fuente de voltaje
	Bombillo

Figura 2.39. Diagrama de circuito eléctrico que representa la Disyunción
Fuente: Aguilera y Bejarano, 2018

Los interruptores (`A` y `B`) tendrán dos estados, estos estados se tomarán como valores lógicos de la siguiente forma: activado (`True`, `T`) y desactivado (`False`, `F`). Asimismo, el bombillo tendrá la misma relación de estados, es decir, activado o prendido (`True`, `T`) y desactivado o apagado (`False`, `F`).

Se debe tener claro que si un interruptor está activado significa que permitiría el paso de electricidad proveniente de la fuente de voltaje y si el bombillo se activa es porque recibe la electricidad proveniente de la fuente de voltaje y habilitada por la activación de los interruptores.

A continuación, se explicará el funcionamiento del operador `OR`, mediante el análisis de la tabla de verdad y el circuito eléctrico en paralelo que se muestra en el cuadro 2.9:

Cuadro 2.9. Tabla de verdad del operador `OR` aplicada al circuito del bombillo del diagrama de la figura 2.39

Entradas		Salida	Descripción
Interruptor A	**Interruptor B**	**Estado del bombillo**	
`F`	`F`	`F`	Si ambos interruptores tienen un valor de `F` entonces el bombillo estaría en `F`, ya que la electricidad no pasaría por ninguno de los dos interruptores.
`F`	`T`	`T`	Si se activa el interruptor `B`, pero se mantiene desactivado el `A` la

			electricidad pasaría por el interruptor B, por ende, el bombillo se activará.
T	F	T	Si se hace lo opuesto al estado de entradas anterior, el bombillo se activaría. Ya que la electricidad pasa el interruptor A, aunque no pase por el B.
T	T	T	Si se activan ambos interruptores, el bombillo se activaría.
Nota: Podemos observar que con solo un operador (interruptor) OR esté activado (T) el bombillo se encenderá (T).			

El operador OR se representa por medio de los siguientes símbolos, letras o palabras (cuadro 2.10):

Cuadro 2.10. Representación del operador OR

Operador	Símbolo	Definición
O, OR	||, V	Disyunción
Nota: Se recomienda utilizar el mismo símbolo durante todo el desarrollo de una expresión		

Analicemos algunos ejemplos del funcionamiento del operador OR, tomando como base los siguientes valores: A=T,B=F,C=T. La figura 2.40 muestra la forma de aplicar el operador OR a estas variables:

Paso	Explicación
A V B	Lo primero que debemos hacer es sustituir las letras A y B por el valor dado en la descripción del ejemplo.
T V F	Luego de la sustitución, procedemos a aplicar el operador OR a los valores lógicos.
T	Según el análisis del operador OR y siguiendo el ejemplo del circuito eléctrico, podemos decir que un interruptor está activado y otro desactivado, por lo que el bombillo se activaría, es decir, tiene un valor de True (T), este sería el resultado de la expresión.

Figura 2.40. Ejemplo paso a paso del funcionamiento del operador OR
Fuente: Aguilera y Bejarano, 2018

Estudiemos otro ejemplo.

Paso	Explicación
C V B V A	Lo primero que debemos hacer es sustituir las letras por el valor dado en la descripción del ejemplo.
T V F V T	Luego de la sustitución, procedemos a resolver la primera pareja de elementos. Si resolvemos esta expresión basándonos en el circuito eléctrico, tendremos un resultado de T, ya que uno de los interruptores está activado, por ende, el bombillo también (T).
T V T	El resultado de la operación anterior es T, luego procedemos a resolver este resultado con el otro valor lógico.
T	Si analizamos la expresión del paso anterior con el ejemplo del circuito eléctrico, notamos que ambos interruptores tienen un valor de T, por lo que el bombillo también tendrá ese valor (T).

Figura 2.41. Ejemplo paso a paso del funcionamiento del operador OR con tres variables
Fuente: Aguilera y Bejarano, 2018

Veamos un último ejemplo:

Paso	Explicación
C V A	Lo primero que debemos hacer es sustituir las letras por el valor dado en la descripción del ejemplo.
T V T	Luego de la sustitución, procedemos a resolver la pareja de elementos. Si resolvemos esta expresión basándonos en el circuito eléctrico, tendremos un resultado de T, ya que ambos interruptores están activados, por ende, el bombillo también (T).
T	Este sería el resultado final de la expresión.

Figura 2.42. Ejemplo paso a paso del funcionamiento del operador OR
Fuente: Aguilera y Bejarano, 2018

3.1.20Ejercicios

Resuelva cada una de las expresiones presentadas a continuación, elabore el desarrollo de cada expresión para llegar a su respuesta, tal y como se hizo en los ejemplos. Tome en cuenta los siguientes valores iniciales: A=T (true), B=F (false), C=F (false).

1. A V B V B V C
2. B V B

3. A V A V B
4. C V A V B V A
5. A V B V C

3.1.21 Respuestas

1. A V B V B V C

   T V F V F V T

   T V F V T

   T V T

   R/ T

2. B V B

   F V F

   R/ F

3. A V A V B

   T V T V F

   T V F

   R/ T

4. C V A V B V A

   T V T V F V T

   T V F V T

   T V T

   R/ T

5. A V A V C

   T V T V F

`T V F`

R/ T

3.1.22 Precedencia operadores lógicos

La precedencia de los operadores lógicos dicta que primero se resuelven las negaciones (**NO / NOT**), luego las conjunciones (**Y / AND**) y por último las disyunciones (**O / OR**).

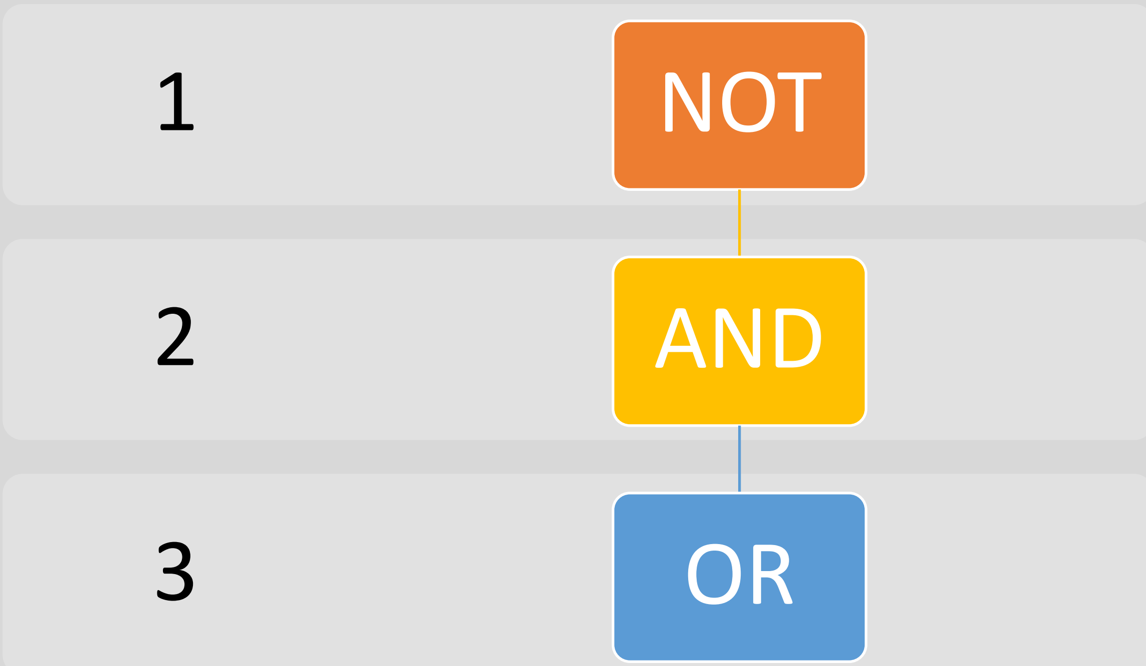


Figura 2.43. Precedencia de los operadores lógicos
Fuente: Aguilera y Bejarano, 2018

Veamos un par de ejemplos, considerando los valores `A=T`, `B=T`, `C=F`, en las figuras 2.44 y 2.45:

Paso	Explicación
`!A V B ∧ !C`	Lo primero que debemos hacer es sustituir las letras por el valor dado en la descripción del ejemplo.
`!T V T ∧ !F`	Luego de la sustitución, procedemos a aplicar el operador `NOT` a los valores lógicos.
`F V T ∧ T`	Como podemos ver a los elementos a los que se les aplicó el `NOT` ya cambiaron su valor. Ahora resolveremos el operador `AND`.
`F V T`	Por último, ejecutamos el operador `OR`.
`T`	El resultado definitivo de la expresión es `T` (`True`)

Figura 2.44. Ejemplo paso a paso d la aplicación de la precedencia de los operadores lógicos
Fuente: Aguilera y Bejarano, 2018

Analicemos un ejemplo más, con los mismos valores lógicos para las letras A, B y C del ejemplo anterior.

Paso	Explicación
!C V !!B ∧ A ∧ !B ∧ C	Lo primero que debemos hacer es sustituir las letras por el valor dado en la descripción del ejemplo.
!F V !!T ∧ T ∧ !T ∧ F	Luego de la sustitución, procedemos a aplicar los operadores `NOT` a los valores lógicos. Notemos que hay un operador `NOT` doble (`!!T`), por lo que primero tomamos uno y, en el siguiente paso, resolveremos el otro.
T V !F ∧ T ∧ F ∧ F	Ahora procedemos con el último operador `NOT` que tiene la expresión.
T V T ∧ T ∧ F ∧ F	Ya no tenemos más operadores `NOT`, por ende, seguiremos con los `AND`. Tomaremos la primera pareja que contenga dicho operador.
T V T ∧ F ∧ F	El resultado de la expresión anterior es `T`. En este paso tomamos la siguiente pareja formada por el resultado de la anterior y por el siguiente valor lógico.
T V F ∧ F	Después de haber procesado la expresión anterior, nos ocuparemos del último `AND` que queda en la expresión.
T V F	Posterior a la resolución de los últimos elementos, resolveremos el único `OR`.
T	El resultado final de la expresión es `T (True)`.

Figura 2.45. Ejemplo paso a paso de la aplicación de la precedencia de los operadores lógicos
Fuente: Aguilera y Bejarano, 2018

3.1.23Ejercicios

Resuelva cada una de las expresiones presentadas a continuación, elabore el desarrollo de cada expresión para llegar a su respuesta, tal y como se hizo en los ejemplos. Tome en cuenta los siguientes valores iniciales: `A=F(false),B=F(false),C=T(true).`

1. `A V B Λ !B V C V !!C`
2. `B V B Λ A`
3. `!A V A Λ C V B Λ !A Λ`

4. !C V !A V !!B Λ A
5. A Λ B Λ C V !A

3.1.24Respuestas

1. A V B Λ !B V C V !!C

   F V F Λ !F V T V !!T

   F V F Λ T V T V !F

   F V F Λ T V T V T

   F V F V T V T

   F V T V T

   T V T

   R/ T

2. B V B Λ A

   F V F Λ F

   F V F

   R/ F

3. !A V A Λ C V B Λ !A Λ !B

   !F V F Λ T V F Λ !F Λ !F

   T V F Λ T V F Λ T Λ T

   T V F V F Λ T Λ T

   T V F V F Λ T

   T V F V F

   T V F

   R/ T

4. !C V !A V !!B Λ A

```
!T V !F V !!F Λ F
F V T V !T Λ F
F V T V F Λ F
F V T V F
T V F
```

R/ T

```
5. A Λ B Λ C V !A
   F Λ F Λ T V !F
   F Λ F Λ T V T
   F Λ T V T
   F V T
```

R/ T

3.5 Otras precedencias

Como ya vimos, la precedencia es el orden en que se resolverá una expresión, existen diferentes tipos de precedencias: la **explícita**, por **categoría** y la **posicional**, cada una debe evaluarse y aplicar según lo que especifica.

Recordemos que, además de las tres precedencias anteriores, también están las **precedencias de los diferentes tipos de operadores**.

La precedencia **explícita** es la que se debe resolver **primero**, es decir, se resuelven las expresiones entre paréntesis, corchetes y llaves, antes de las que se ubican fuera de los mismos. Al resolver las expresiones entre paréntesis, se deben aplicar también las otras precedencias que se encuentran dentro de estos.

Luego, se aplica la precedencia por **categoría**, paralelamente a esta se debe **considerar la precedencia por operador y la posicional**.

3.5.1 Precedencia explícita

Este tipo de precedencia es la que se brinda por medio de los paréntesis **( )**, corchetes **[ ]** o llaves **{ }**. Se resuelven las expresiones de **adentro hacia afuera**.

Figura 2.45. Orden de aplicación de la precedencia explicita
Fuente: Aguilera y Bejarano, 2018

A continuación, analizaremos algunos ejemplos utilizando diversos tipos de operadores. El primero será con los operadores aritméticos (figura 2.46):

Paso	Explicación

`2 (2^3 + 16) / 3 - -5`
 	
`2 (8 + 16) / 3 - -5`
	Primero resolvemos lo que está entre paréntesis. Al analizar esta expresión notamos que debemos aplicar la precedencia de los operadores aritméticos, es decir, resolver primero la potencia y luego la suma.

`2 (24) / 3 - -5`
 	
`2 * 24 / 3 - -5`
	Notemos que entre el primer número de la expresión (el 2) y el paréntesis no hay ningún signo, en este caso se asume que el signo es una **multiplicación**.
	Por ende, al retirar los paréntesis debemos considerar ese signo.
	Ya sin paréntesis en la expresión podemos resolver los signos primero.

`2 * 24 / 3 + 5`	En este paso solo nos queda resolver las **multiplicaciones y divisiones**, esto se puede hacer a como las vayamos encontrando de **izquierda a derecha**, en este caso resolveremos primero la multiplicación.

`48 / 3 + 5`	Seguidamente, resolvemos la **división** que nos queda.
`16 + 5`	Finalmente, resolvemos la **suma** que nos queda.
`21`	El resultado definitivo es **21**.

Figura 2.46. Ejemplo de aplicación de la precedencia explicita con operadores aritméticos
Fuente: Aguilera y Bejarano, 2018

Analicemos otro ejemplo con operadores aritméticos (figura 2.47):

Paso	Explicación
5 + { 2 [(7^0 * 2 + 3) – (10 % 4 + $^{-}$3) (2 + 1 – 3)] }	Primero resolvemos algunos términos de cada paréntesis, esto lo podemos hacer de izquierda a derecha, ya que cada paréntesis tiene la misma jerarquía. En cada uno de los grupos se deben aplicar las precedencias de los operadores aritméticos.
5 + { 2 [(1 * 2 + 3) – (2 + $^{-}$3) * 0] }	En el primer paréntesis resolvimos la potenciación, ahora continuaremos con la multiplicación. En el segundo paréntesis resolvemos la suma y en el tercero eliminamos el paréntesis y colocamos el operador de multiplicación.
5 + { 2 [(2 + 3) – ($^{-}$1) * 0] }	Ahora, resolvemos la suma del primer paréntesis y eliminamos los segundos paréntesis.
5 + { 2 [(5) – $^{-}$1 * 0] }	En este paso eliminamos los primeros paréntesis y aplicamos la ley de signos.
5 + { 2 [5 + 1 * 0] }	Ya eliminamos todos los paréntesis, ahora nos queda resolver la expresión entre los corchetes.
5 + { 2 [5 + 0] }	Después, hacemos la suma entre corchetes y agregamos el operador entre el 2 y los corchetes.
5 + { 2 * 5 }	Ahora, realizamos la multiplicación entre las llaves y eliminamos la mismas.
5 + 10	Por último, hacemos la suma.

`15`	El resultado final es 15.

Figura 2.47. Ejemplo de aplicación de la precedencia explicita con operadores aritméticos
Fuente: Aguilera y Bejarano, 2018

Estudiemos un par de ejemplos con operadores relacionales, empleando expresiones y números (figura 2.48):

Paso	Explicación
`(37 - 25) = (2 + 1 - 12 / 3)`	Lo primero que debemos hacer es resolver las expresiones entre paréntesis, notemos que en el segundo paréntesis debemos aplicar la precedencia de los operadores aritméticos.
`12 = (2 + 1 - 4)`	En el primer paréntesis resolvimos la resta y de una vez eliminamos el paréntesis. Del segundo paréntesis hicimos la división, ahora, ejecutaremos la suma y la resta.

12 = -1	Ahora, debemos resolver el operador relacional que nos quedó.
F	El resultado final es `F`(`false`), ya que `12` no es igual a `-1`.

Figura 2.48. Ejemplo de aplicación de la precedencia explicita con operadores relacionales utiliazndo números y expresiones
Fuente: Aguilera y Bejarano, 2018

Veamos el segundo ejemplo (figura 2.49):

Paso	Explicación
[(-7 – 2 + (2 – <u>1 * 3</u>)) * 2] <= 5 **[(-7 – 2 + (2 – 3)) * 2] <= 5**	Lo primero que debemos hacer es resolver la expresión entre los paréntesis internos a los primeros. En esta expresión debemos aplicar la precedencia de los operadores aritméticos, por lo que resolveremos primero la multiplicación y luego la suma.

[(-7 – 2 + -1) * 2] <= 5	Luego, resolvemos la expresión que nos queda en los paréntesis externos.
[-10 * 2] <= 5	Ahora, debemos hacer la expresión entre corchetes.
-20 <= 5	En este paso, procesamos el operador relacional.
T	El resultado de la expresión es `T`(`true`) ya que el `-20` es menor a `5`. Aunque no es igual, es menor y recordemos que en esos operadores con solo una de las condiciones que se cumpla es suficiente para tener un resultado verdadero.

Figura 2.49. Ejemplo de aplicación de la precedencia explicita con operadores aritméticos
Fuente: Aguilera y Bejarano, 2018

Ahora, estudiemos un ejemplo con operadores lógicos en la figura 2.50. Consideremos los siguientes valores para las letras `A=T` (`true`), `B=F` (`false`), `C=F` (`false`). Además, si en la expresión aparece una `T` o una `F` asuma que significan `true` (`verdadero`) y `false` (`falso`) respectivamente.

Paso	Explicación
`!C V ( !!B V F ) ∧ !( A ∧ !T V C )`	Lo primero que debemos hacer es sustituir las letras por el valor dado en la descripción del ejemplo.
`!F V ( !!F V F ) ∧ !( T ∧ !T V F )`	Luego de la sustitución, procedemos a aplicar los operadores `NOT` a los valores lógicos que están entre paréntesis.
`!F V ( !T V F ) ∧ !( T ∧ F V F )`	Ahora procedemos con el último operador `NOT` que tiene la expresión en los primeros paréntesis y también con la expresión de los segundos paréntesis, tomando en cuenta la precedencia de los operadores lógicos.
`!F V ( F V F ) ∧ !( F V F )`	En este paso, hacemos las expresiones entre los paréntesis y los eliminamos. El `NOT` que está afuera de los segundos paréntesis afectará al resultado final de dicha expresión.
`!F V F ∧ !F`	A continuación, resolveremos los `NOT` que nos quedan.
`T V F ∧ T`	Ya, con ningún `NOT` en la expresión, resolvemos el `AND`.
`T V F`	Posterior a la resolución de los últimos elementos, resolveremos el único `OR`.
`T`	El resultado final de la expresión es `T`(`True`).

Figura 2.50. Ejemplo de aplicación de la precedencia explicita con operadores lógicos
Fuente: Aguilera y Bejarano, 2018

3.5.2 Ejercicios

Resuelva cada una de las expresiones presentadas a continuación, elabore el desarrollo de cada expresión para llegar a su respuesta, tal y como se hizo en los ejemplos. Tome en cuenta los siguientes valores iniciales: `A=F`(`false`) `B=F`(`false`) `C=T`(`true`). También, si en la expresión aparece una `T` o una `F` asuma que significan `true` (`verdadero`) y `false` (`falso`) respectivamente.

```
1. 3 + 4 − 2 − ( 1 * 3 * 4 + 18 / 2 ) ( − 3 * 8 + 5 * 2 )^2
```

```
2. 18 / 2 ( 5 * ¯1 + ( 2 − ¯10 / 5 ) ) < ¯9 + ¯1
```

```
3. ! ( A V B V T Λ !( B V C ) V !C )
```

4. ! (! (B V B Λ A) Λ F)

5. - (3 * 30 / 5 + (5 + 14 - 16 / (1 + 3)) + $^{-}$25 * (2 + 11 - 14)
   7 – 15 – 5 / 10) + 8

6. !A V A Λ [C V (B Λ !A Λ !(F V C Λ !C)) V F] V !A Λ T

7. 3 * (2 + 2 / 4) – 1 <= 7 * 7

8. 1 + 3 – (5 + 6 + $^{-}$4) 4 / 2 – 2 (2 + 24 - 10) * 3^3

9. 4 * 4 / 2 = 2^3 – (2^3 + -11)

3.5.3 Respuestas

1. 3 + 4 – 2 – (1 * 3 * 4 + 18 / 2) ($^{-}$3 * 8 + 5 * 2)^2

   3 + 4 – 2 – (12 + 18 / 2) (24 + 5 * 2)^2

   3 + 4 – 2 – (12 + 9) (24 + 10)^2

   3 + 4 – 2 – 21 * 34^2

   3 + 4 – 2 – 21 * 1156

   3 + 4 – 2 – 24276

   R/ 24271

2. 18 / 2 (5 * $^{-}$1 + (2 – $^{-}$10 / 5)) < $^{-}$9 + $^{-}$1

   18 / 2 (5 * $^{-}$1 + (2 +10 / 5)) < $^{-}$9 + $^{-}$1

   18 / 2 (5 * $^{-}$1 + (2 + 2)) < $^{-}$9 + $^{-}$1

   18 / 2 (5 * $^{-}$1 + 4) < $^{-}$9 + $^{-}$1

   18 / 2 ($^{-}$5 + 4) < $^{-}$9 + $^{-}$1

18 / 2 * $^{-}$1 < $^{-}$9 + $^{-}$1

9 * $^{-}$1 < $^{-}$10

$^{-}$9 < $^{-}$10

R/ F

3. ! (A V B V T Λ !(B V C) V !C)

! (F V F V T Λ !(F V T) V !T)

! (F V F V T Λ !(T) V !T)

! (F V F V T Λ F V !T)

! (F V F V T Λ F V F)

! (F V F V F V F)

!F

R/ T

4. ! (! (B V !B Λ A) Λ F)

! (! (F V !F Λ F) Λ F)

! (! (F V T Λ F) Λ F)

! (! (F V F) Λ F)

! (!F Λ F)

! (T Λ F)

!F

R/ T

5. - (3 * 30 / 5 + (5 + 14 - 16 / (1 + 3)) + $^{-}$25 * (2 + 11 - 14) 7 – 15 – 5 / 10) + 8

- (3 * 30 / 5 + (5 + 14 - 16 / 4) + $^{-}$25 * $^{-}$1 * 7 – 15 – 5 / 10) + 8

- (3 * 30 / 5 + (5 + 14 - 4) + $^{-}$25 * $^{-}$1 * 7 – 15 – 5 / 10) + 8

- (3 * 30 / 5 + 15 + $^{-}$25 * $^{-}$1 * 7 – 15 – 5 / 10) + 8

- (90 / 5 + 15 + $^{-}$25 * $^{-}$1 * 7 – 15 – 5 / 10) + 8

- (18 + 15 + -25 * -1 * 7 – 15 – 5 / 10) + 8

- (18 + 15 + 175 – 15 – 5 / 10) + 8

- (18 + 15 + 175 – 15 – 0.5) + 8

-192.5 + 8

R/ -184.5

6. !A V A Λ [C V (B Λ !A Λ !(F V C Λ !C)) V F] V !A Λ T

!F V F Λ [T V (F Λ !F Λ !(F V T Λ !T)) V F] V !F Λ T

!F V F Λ [T V (F Λ !F Λ !(F V T Λ F)) V F] V !F Λ T

!F V F Λ [T V (F Λ !F Λ !(F V F)) V F] V !F Λ T

!F V F Λ [T V (F Λ !F Λ !F) V F] V !F Λ T

!F V F Λ [T V (F Λ !F Λ T) V F] V !F Λ T

!F V F Λ [T V (F Λ T Λ T) V F] V !F Λ T

!F V F Λ [T V F V F] V !F Λ T

!F V F Λ T V !F Λ T

T V F Λ T V T Λ T

T V F V T Λ T

T V F V T

R/ T

7. 3 * (2 + 2 / 4) – 1 <= 7 * 7

3 * (2 + 0.5) – 1 <= 7 * 7

3 * 2.5 – 1 <= 7 * 7

7.5 – 1 <= 49

6.5 <= 49

R/ T

8. 1 + 3 – (5 + 6 + -4) 4 / 2 – 2 (2 + 24 - 10) * 3^3

```
1 + 3 – 7 * 4 / 2 – 2 * 16 * 3^3

1 + 3 – 7 * 4 / 2 – 2 * 16 * 27

1 + 3 – 28 / 2 – 2 * 16 * 27

1 + 3 – 14 – 2 * 16 * 27

1 + 3 – 14 – 864
```

R/ $^{-}$874

```
9. 4 * 4 / 2 = 2^3 – (2^3 + -11)

   4 * 4 / 2 = 2^3 – (8 + -11)

   4 * 4 / 2 = 2^3 – -3

   4 * 4 / 2 = 2^3 + 3

   4 * 4 / 2 = 8 + 3

   16 / 2 = 11

   8 = 11
```

R/ F

3.5.4 Precedencia posicional

La precedencia posicional se aplica cuando se tienen varios operadores de la misma jerarquía, en este caso se deben resolver los operadores de la misma categoría de **izquierda a derecha**.

Analicemos un ejemplo con operadores aritméticos en la figura 2.51:

Paso	Explicación
2 + (2^3 + 16) + $^{-}$3 - $^{-}$5 **2 + (8 + 16) + $^{-}$3 - $^{-}$5**	Primero resolvemos lo que está entre paréntesis. Al analizar esta expresión notamos que debemos aplicar la precedencia de los operadores aritméticos, es decir, resolver primero la potencia y luego la suma.
2 + 24 + $^{-}$3 - $^{-}$5	En este paso aplicaremos la ley de signos.
2 + 24 + $^{-}$3 + 5	Ahora, solo tenemos operadores aritméticos de la misma jerarquía, por lo que los resolveremos de izquierda a derecha.
28	El resultado definitivo es **28**.

Figura 2.51. Ejemplo de aplicación de la precedencia posicional con operadores aritméticos
Fuente: Aguilera y Bejarano, 2018

Vemos un ejemplo con operadores lógicos (figura 2.52):

Paso	Explicación
T V (! !F V F) ∧ !(T ∧ !T V F)	Lo primero que debemos hacer es aplicar los operadores `NOT` a los valores lógicos que están entre paréntesis.
T V (!T V F) ∧ !(T ∧ F V F)	Ahora procedemos con el último operador `NOT` que tiene la expresión en los primeros paréntesis y también con la expresión de los segundos paréntesis, tomando en cuenta la precedencia de los operadores lógicos.
T V (F V F) V !(F V F)	En este paso, hacemos las expresiones entre los paréntesis y los eliminamos. El `NOT` que está afuera de los segundos paréntesis afectará al resultado final de dicha expresión.
T V F V !F	A continuación, resolveremos el `NOT` que nos queda.
T V F V T	Ya, con ningún `NOT` en la expresión, resolvemos los `OR` de derecha a izquierda.
T	El resultado final de la expresión es `T`(`True`).

Figura 2.52. Ejemplo de aplicación de la precedencia posicional con operadores lógicos
Fuente: Aguilera y Bejarano, 2018

3.5.5 Ejercicios

Resuelva cada una de las expresiones presentadas a continuación, elabore el desarrollo de cada expresión para llegar a su respuesta, tal y como se hizo en los ejemplos. Tome en cuenta los siguientes valores iniciales: `A=F (false)`, `B=F (false)`, `C=T (true)`. También, si en la expresión aparece una `T` o una `F` asuma que significan true (verdadero) y false (falso) respectivamente.

1. 3 + 4 – 2 – (3 * 4 + 8 / 2) + (– 3 + 5 + 2)^2

2. ! (A V B V T Λ !(B V C) V !C V F V !T)

3. ! (! (B V B Λ A) Λ F Λ C Λ !A Λ B)

4. - (2 * 30 / 5 + (5 + 14) + $^{-}$25 + (2 + 11 – 14) 7 – 15 – 5 / 10) + 8

5. 1 + 3 – (5 + 6 + $^{-}$4) 4 + $^{-}$2 – 2 (2 + 24 - 10) * 2^3

6. C V B V T V !(B Λ C Λ !A) V !C V F

3.5.6 Respuestas

1. 3 + 4 – 2 – (3 * 4 + 8 / 2) + (– 3 + 5 + 2)^2

   3 + 4 – 2 – (12 + 4) + (4)^2

   3 + 4 – 2 – (16) + 4^2

   3 + 4 – 2 – 16 + 16

   12 – 2 – 16 + 16

   10 – 16 + 16

   $^{-}$6 + 16

   R/ 10

2. ! (A V B V T Λ !(B V C) V !C V F V !T)

   ! (F V F V T Λ !(F V T) V !T V F V !T)

   ! (F V F V T Λ !T V !T V F V !T)

   ! (F V F V T Λ F V F V F V F)

   ! (F V F V F V F V F V F)

   ! (F V F V F V F V F)

   ! (F V F V F V F)

   ! (F V F V F)

   ! (F V F)

!F

R/ T

3. ! (! (B V B Λ A) Λ F Λ C Λ !A Λ B)

! (! (F V F Λ F) Λ F Λ T Λ !F Λ F)

! (! (F V F) Λ F Λ T Λ !F Λ F)

! (!F Λ F Λ T Λ !F Λ F)

! (T Λ F Λ T Λ T Λ F)

! (F Λ T Λ T Λ F)

! (F Λ T Λ F)

! (F Λ F)

!F

R/ T

4. - (2 * 30 / 5 + (5 + 14) + ¯25 + (2 + 11 − 14) 7 − 15 − 5 / 10) + 8

- (2 * 30 / 5 + 19 + ¯25 + (13 − 14) 7 − 15 − 5 / 10) + 8

- (2 * 30 / 5 + 19 + ¯25 + ¯1 * 7 − 15 − 5 / 10) + 8

- (60 / 5 + 19 + ¯25 + ¯1 * 7 − 15 − 5 / 10) + 8

- (12 + 19 + ¯25 + ¯1 * 7 − 15 − 5 / 10) + 8

- (12 + 19 + ¯25 + ¯7 − 15 − 5 / 10) + 8

- (12 + 19 + ¯25 + ¯7 − 15 − 0.5) + 8

- (31 + ¯25 + ¯7 − 15 − 0.5) + 8

- (6 + ¯7 − 15 − 0.5) + 8

- (¯1 − 15 − 0.5) + 8

- (¯16 − 0.5) + 8

- (¯16.5) + 8

- ¯16.5 + 8

16.5 + 8

R/ 24.5

```
5. 1 + 3 – ( 5 + 6 + -4 ) 4 + -2 – 2 ( 2 + 24 - 10) * 2^3
   1 + 3 – ( 11 + -4 ) 4 + -2 – 2 ( 26 - 10) * 2^3
   1 + 3 – 7 * 4 + -2 – 2 * 16 * 2^3
   1 + 3 – 7 * 4 + -2 – 2 * 16 * 8
   1 + 3 – 28 + -2 – 2 * 16 * 8
   1 + 3 – 28 + -2 – 32 * 8
   1 + 3 – 28 + -2 – 256
   4 – 28 + -2 – 256
   -24 + -2 – 256
   -26 – 256
```

R/ 282

```
6. C V B V T V !( B Λ C Λ !A ) V !C V F
   T V F V T V !( F Λ T Λ !F ) V !T V F
   T V F V T V !( F Λ T Λ T ) V !T V F
   T V F V T V !( F Λ T ) V !T V F
   T V F V T V !F V !T V F
   T V F V T V T V F V F
   T V T V T V F V F
   T V T V F V F
   T V F V F
   T V F
```

R/ T

3.5.7 Precedencia por categoría

De acuerdo con la categoría, en las expresiones se deben resolver primero los operadores **aritméticos**, luego los **relacionales** y por último los **lógicos**.

Debemos recordar que también se consideran las otras precedencias, es decir, se deben resolver primero los operadores aritméticos, pero entre esos se debe considerar la precedencia por operador, o sea, resolver primero las potencias y los signos, luego las multiplicaciones y divisiones y por último las sumas y las restas. A la vez, es necesario analizar si se debe aplicar la precedencia posicional y la explícita.

Estas combinaciones entre precedencias aplican también para los operadores relacionales y lógicos. La figura 2.53 muestra el orden en que se deben aplicar las precedencias de acuerdo a la categoría.

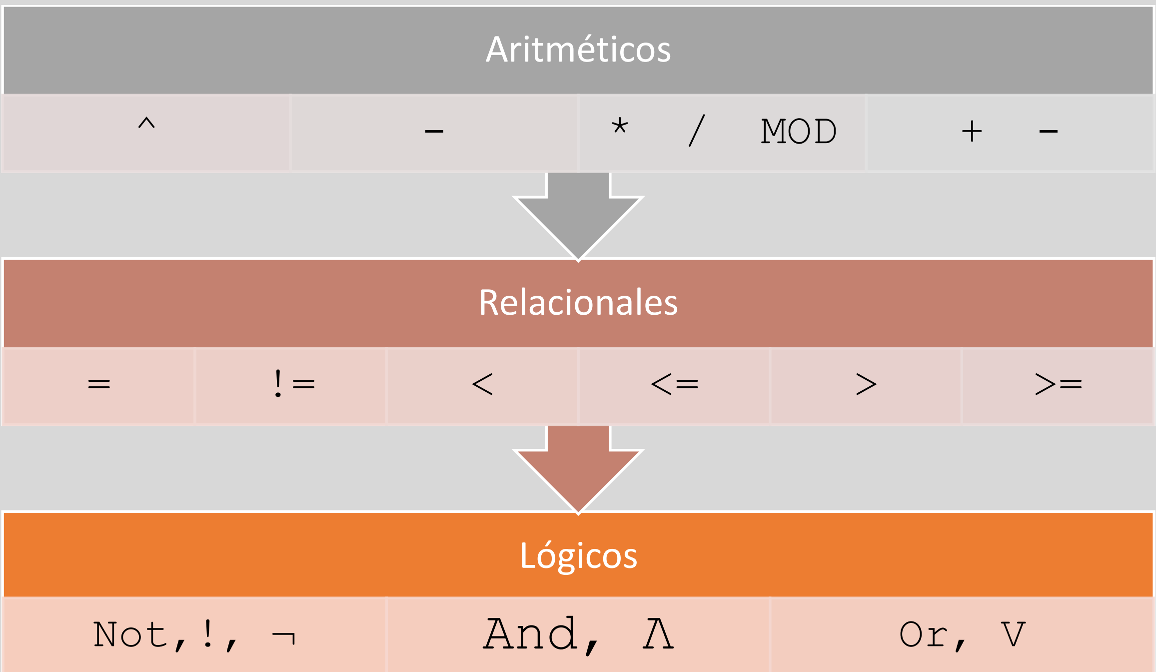


Figura 2.53. Orden de aplicación de las precedencias por categoría
Fuente: Aguilera y Bejarano, 2018

3.5.8 Ejemplos de precedencia

Analicemos algunos ejemplos con operadores aritméticos.

Paso	Explicación
`0 * 2^3 - ¯5 + 10`	Primero resolvemos la potencia, el resultado de esta será 8. También, resolveremos el signo antes del ¯5, esto porque tienen la misma jerarquía.
`0 * 8 + 5 + 10`	Ahora, ejecutamos la multiplicación. Debemos notar que, tanto en el paso anterior como en este, estamos aplicando la precedencia por operador, la razón es que en la expresión solo hay operadores aritméticos y no hay paréntesis, corchetes, ni llaves.
`0 + 5 + 10`	En este paso, procesamos las sumas, aplicando la precedencia posicional, es decir, de izquierda a derecha; esto se debe a que todos los operadores

	que nos quedan son de la misma categoría y jerarquía.
`5 + 10`	Aquí, resolvemos la última suma de la expresión.
`15`	El resultado definitivo es `15`

Figura 2.54. Orden de aplicación de las precedencias con operadores aritméticos
Fuente: Aguilera y Bejarano, 2018

Analicemos un ejemplo, un poco más complejo.

Paso	Explicación

`2 * {10^0 – 5 + [25 / (3 + 2 * 1)] * 3}`
 	
`2 * {10^0 – 5 + [25 / (3 + 2)] * 3}`
 	
`2 * {10^0 – 5 + [25 / 5] * 3}`
	En esta expresión tenemos paréntesis, corchetes y llaves, por lo que debemos aplicar precedencia explícita primero, recordemos que en este tipo de precedencia los elementos se resuelven de adentro hacia afuera.
	
	Iniciaremos con los paréntesis, dentro de los mismos debemos resolver la expresión usando la precedencia por operador, por ende, hacemos la multiplicación antes de la suma.

`2 * {10^0 – 5 + [25 / 5] * 3}`	Ahora, ejecutamos la división dentro de los corchetes.

`2 * {10^0 – 5 + 5 * 3}`
 	
`2 * {1 – 5 + 5 * 3}`
 	
`2 * {1 – 5 + 15}`
 	
`2 * {-4 + 15}`
	En este paso, nos queda la expresión dentro de las llaves, lo que haremos será aplicar la precedencia por operador.
	
	Primero, la potencia, luego la multiplicación y finalmente la resta y la suma. En estos dos últimos, utilizamos la precedencia posicional.

`2 * 11`	Aquí, resolvemos la multiplicación que nos queda.
`22`	El resultado definitivo es **22.**

Figura 2.55. Orden de aplicación de las precedencias explicita y con operadores aritméticos
Fuente: Aguilera y Bejarano, 2018

Veamos un ejemplo combinando operadores aritméticos y relacionales.

Paso	Explicación
2 + 3^2 + -5 + 10 < 2 * 3 + 4 / $^{-}$2	Lo que haremos primero es resolver los operadores aritméticos, es decir, aplicar la precedencia por categoría. Al resolver estos operadores usaremos la precedencia por operador. Resolveremos primero la potencia.
2 + 9 + -5 + 10 < 2 * 3 + 4 / $^{-}$2	Ahora, haremos la multiplicación y luego la división, aquí aplicamos la precedencia posicional, ya que estos dos operadores (`*` y `/`) tienen la misma jerarquía.
2 + 9 + -5 + 10 < 6 + $^{-}$2 **7 + -5 + 10 < 6 + $^{-}$2** **2 + 10 < 6 + $^{-}$2** **12 < 6 + $^{-}$2** **12 < 4**	En este paso, emplearemos la precedencia posicional para resolver las sumas y restas. En el último paso nos queda una expresión con el operador relacional. Al analizarlo, tenemos que es falso que `12` sea menor a `4`, por lo que el resultado final es `falso` (`F`).
F	El resultado definitivo es `F`.

Figura 2.56. Orden de aplicación de las precedencias con operadores aritméticos y relacionales
Fuente: Aguilera y Bejarano, 2018

Analicemos un segundo ejemplo, utilizando los mismos operadores, pero agregando elementos para utilizar la precedencia explícita.

Paso	Explicación
11 = -1 {^2-5+[20/(3+2*$^{-}$2)]} **11 = -1 {5^2-5+[20/(3+$^{-}$4)]}** **11 = -1 {5^2-5+[20/$^{-}$1]}**	Lo primero es aplicar la precedencia explícita para resolver lo que está entre los paréntesis. En esta expresión aplicaremos la precedencia por operador, es decir, procesaremos primero la multiplicación y luego la suma.
11 = -1 {5^2-5+[20/$^{-}$1]}	Ahora, haremos la división dentro de los corchetes.
11 = -1 {5^2- 5+$^{-}$20} **11 = -1 {25-5+$^{-}$20}** **11 = -1*0**	La expresión que nos queda está entre las llaves, aplicamos la precedencia por operador y resolveremos primero la potencia. Luego, mediante la precedencia posicional resolveremos la resta y la suma.

11 = 0	Ahora, resolvemos la multiplicación y luego ejecutamos el operador relacional.
F	Es falso que 11 sea igual a 0, por lo que la respuesta definitiva de la expresión es False (F).

Figura 2.57. Orden de aplicación de las precedencias explicita y con operadores aritméticos y relacionales
Fuente: Aguilera y Bejarano, 2018

Ahora, resolvamos una expresión que incluye los tres operadores: aritméticos, relacionales y lógicos.

Explicación y pasos
Lo primero que debemos resolver son todos los operadores aritméticos, una vez que ya no tengamos este tipo de operadores en la expresión, procederemos con los relacionales y por último con los lógicos. En cada caso, consideraremos la aplicación de las precedencias explícita y posicional, empecemos con las potencias y signos. **0*2^3-¯5+10<2*3+4-2 V 2-10/2*3<3+2*1 Λ ¯10 > ¯1**
Ahora, procederemos con las multiplicaciones y divisiones; considerando también la precedencia posicional. **0 * 8 + 5 + 10 < 2 * 3 + 4 - 2 V 2 - 10 / 2 * 3 < 3 + 2 * 1 Λ ¯10 > ¯1**
Aún nos queda una multiplicación más. **0 + 5 + 10 < 6 + 4 - 2 V 2 - 5 * 3 < 3 + 2 Λ ¯10 > ¯1**
En este paso, iniciamos la resolución de sumas y restas. **0 + 5 + 10 < 6 + 4 - 2 V 2 - 15 < 3 + 2 Λ ¯10 > ¯1**
Como podemos ver, en la expresión ya no hay operadores aritméticos, por ende, podemos resolver los relacionales. **15 < 8 V ¯13 < 5 Λ ¯10 > ¯1**

Notemos que solo nos quedan operadores lógicos. Debemos aplicar la precedencia por operador, como la expresión no tiene operadores `NOT`, procederemos con el `AND` y luego el `OR`. **F V T Λ F**
Acá, resolveremos el `OR`. **F V F**
El resultado definitivo es `false` (`F`) **F**
Este es un resumen de la solución del ejemplo: **0 * 2^3 - ⁻5 + 10 < 2 * 3 + 4 - 2 V 2 – 10 / 2 * 3 < 3 + 2 * 1 Λ ⁻10 > ⁻1** **0 * 8 + 5 + 10 < 2 * 3 + 4 - 2 V 2 – 10 / 2 * 3 < 3 + 2 * 1 Λ ⁻10 > ⁻1** **0 + 5 + 10 < 6 + 4 - 2 V 2 – 5 * 3 < 3 + 2 Λ ⁻10 > ⁻1** **0 + 5 + 10 < 6 + 4 - 2 V 2 – 15 < 3 + 2 Λ ⁻10 > ⁻1** **15 < 8 V ⁻13 < 5 Λ ⁻10 > ⁻1** **F V T Λ F** **F V F** **F**

Figura 2.58. Orden de aplicación de las precedencias con operadores aritméticos, relacionales y lógicos
Fuente: Aguilera y Bejarano, 2018

Ahora, hagamos un ejemplo más complejo. Consideremos los siguientes valores para las letras: `A=F, B=T, C=F.`

Explicación y pasos
Lo primero será sustituir las letras por los términos lógicos, dados en el enunciado del ejemplo. ! {2 * {10^0 – 5 + [25 / (3 + 2 * 1)] * 3} <= ⁻22 * ⁻1 Λ C V [5 + 2 + 1 – 3 = 2 – 2 * ⁻3] V ! (!A Λ B Λ !C)}

Al analizar la expresión, notamos que debemos aplicar primero la precedencia explícita, es decir, resolver lo que está entre paréntesis, corchetes o llaves y de adentro hacia afuera. Por ende, iniciaremos con los paréntesis.

Dentro de esos paréntesis debemos aplicar la precedencia por operador, resolveremos la multiplicación y luego la suma, el resto de la expresión sigue igual.

! {2 * {10^0 – 5 + [25 / (3 + 2 * 1)] * 3} <= ¯22 * ¯1 ∧ F V [5 + 2 + 1 – 3 = 2 – 2 * ¯3] V ! (!F ∧ T ∧ !F)}

! {2 * {10^0 – 5 + [25 / (3 + 2)] * 3} <= ¯22 * ¯1 ∧ F V [5 + 2 + 1 – 3 = 2 – 2 * ¯3] V ! (!F ∧ T ∧ !F)}

Ahora, hacemos la división que está entre los corchetes.

! {2 * {10^0 – 5 + [25 / 5] * 3} <= ¯22 * ¯1 ∧ F V [5 + 2 + 1 – 3 = 2 – 2 * ¯3] V ! (!F ∧ T ∧ !F)}

Ya resolvimos lo que estaba dentro de los paréntesis y de los corchetes, procederemos con la expresión que está dentro de las llaves.

Recordemos que debemos aplicar la precedencia por operador, ya que hay operadores aritméticos diferentes ejecutaremos primero la potencia, luego la multiplicación y posteriormente la resta y suma.

! {2 * {10^0 – 5 + 5 * 3} <= ¯22 * ¯1 ∧ F V [5 + 2 + 1 – 3 = 2 – 2 * ¯3] V ! (!F ∧ T ∧ !F)}

! {2 * {1 – 5 + 5 * 3} <= ¯22 * ¯1 ∧ F V [5 + 2 + 1 – 3 = 2 – 2 * ¯3] V ! (!F ∧ T ∧ !F)}

! {2 * {1 – 5 + 15} <= ¯22 * ¯1 ∧ F V [5 + 2 + 1 – 3 = 2 – 2 * ¯3] V ! (!F ∧ T ∧ !F)}

! {2 * 11 <= ¯22 * ¯1 ∧ F V [5 + 2 + 1 – 3 = 2 – 2 * ¯3] V ! (!F ∧ T ∧ !F)}

Ahora, iniciaremos con la solución de los siguientes elementos de la precedencia explícita, estos corchetes: **`[ 5 + 2 + 1 − 3 = 2 − 2 * −3]`**. En este grupo tenemos operadores aritméticos y uno relacional, por lo que debemos procesar primero los aritméticos.

Además, al lado izquierdo del igual (=) debemos aplicar la precedencia posicional, ya que todos los operadores aritméticos de esa parte tienen la misma jerarquía; al lado derecho debemos aplicar la precedencia por operador, ya que hay operadores diferentes.

`!{2 * 11 <= ¯22*¯1 ∧ F V [5+2+1−3 = 2−2* ¯3] V !(!F ∧ T ∧ !F)}`

$$!\ \{2 * 11 <= {}^{-}22 * {}^{-}1 \land F \lor [\ 5 = 2 - {}^{-}6\ ] \lor\ !\ (!F \land T \land !F)\}$$

Siguiendo con la resolución de los corchetes, de un lado ya tenemos un solo valor numérico, pero del otro debemos aplicar los signos antes de hacer la resta, esto se debió al resultado de la multiplicación en el paso anterior.

$$!\{2 * 11 <= {}^{-}22 * {}^{-}1 \land F \lor [\ 5 = 2\ \underline{-\ {}^{-}6}] \lor\ !(!F \land T \land !F)\}$$

$$!\{2 * 11 <= {}^{-}22 * {}^{-}1 \land F \lor [\ 5 = \underline{2 + 6}] \lor\ !(!F \land T \land !F)\}$$

$$!\{2 * 11 <= {}^{-}22 * {}^{-}1 \land F \lor [\ 5 = 8] \lor\ !(!F \land T \land !F)\}$$

En este paso, resolvemos el operador relacional, es falso que 5 sea igual a 8, por lo que el resultado es F, de una vez podemos eliminar los corchetes.

$$!\ \{2 * 11 <= {}^{-}22 * {}^{-}1 \land F \lor [\ \underline{5 = 8}] \lor\ !\ (!F \land T \land !F)\}$$

$$!\ \{2 * 11 <= {}^{-}22 * {}^{-}1 \land F \lor F \lor\ !\ (!F \land T \land !F)\}$$

En este punto, nos enfocaremos en la resolución de los últimos paréntesis, estos tienen operadores lógicos, por lo que debemos aplicar la precedencia por operador.

Primero resolveremos los `NOT` y luego los `AND`, afuera de los paréntesis hay un `NOT`, este se aplicará al resultado final de la expresión dentro de los paréntesis.

$$!\ \{2 * 11 <= {}^{-}22 * {}^{-}1 \land F \lor F \lor\ !\ (\ \underline{!F} \land T \land \underline{!F}\ )\}$$

$$!\ \{2 * 11 <= {}^{-}22 * {}^{-}1 \land F \lor F \lor\ !\ (\ \underline{T \land T} \land T\ )\}$$

$$!\ \{2 * 11 <= {}^{-}22 * {}^{-}1 \land F \lor F \lor\ !\ (\ \underline{T \land T}\ )\}$$

$$!\ \{2 * 11 <= {}^{-}22 * {}^{-}1 \land F \lor F \lor \underline{!T}\ \}$$

$$!\ \{2 * 11 <= {}^{-}22 * {}^{-}1 \land F \lor F \lor F\ \}$$

Ya tenemos una sola expresión entre llaves, la misma tiene operadores aritméticos, relacionales y lógicos, por lo que aplicaremos la precedencia por categoría.

! { 2 * 11 <= $^{-}$22 * $^{-}$1 Λ F V F V F }

! { 22 <= $^{-}$22 * $^{-}$1 Λ F V F V F }

Una vez que resolvimos los operadores aritméticos, procesamos los relacionales.

! { 22 <= 22 Λ F V F V F }

Ahora, solo nos quedan operadores lógicos, como no hay `NOT`, iniciaremos con los que siguen en la jerarquía, el `AND` y al final los `OR`.

! { T Λ F V F V F }

! { F V F V F }

! { F V F }

! { F }

En el paso final, nos quedó un `F` entre llaves y un `NOT` fuera de ellas, por lo que el `NOT` se aplicará a este `F`. El resultado definitivo de la expresión es **T**.

!F

T

Este es un resumen de la solución del ejemplo:

! {2 * {10^0 – 5 + [25 / (3 + 2 * 1)] * 3} <= $^{-}$22 * $^{-}$1 Λ C V [5 + 2 + 1 – 3 = 2 – 2 * $^{-}$3] V ! (!A Λ B Λ !C)}

! {2 * {10^0 – 5 + [25 / (3 + 2 * 1)] * 3} <= $^{-}$22 * $^{-}$1 Λ F V [5 + 2 + 1 – 3 = 2 – 2 * $^{-}$3] V ! (!F Λ T Λ !F)}

! {2 * {10^0 – 5 + [25 / (3 + 2)] * 3} <= $^{-}$22 * $^{-}$1 Λ F V [5 + 2 + 1 – 3 = 2 – 2 * $^{-}$3] V ! (!F Λ T Λ !F)}

```
! {2 * {10^0 - 5 + [25 / 5] * 3} <= -22 * -1 ∧ F V [5 + 2 + 1 - 3 = 2 - 2 * -3] V ! (!F ∧ T ∧ !F)}
! {2 * {10^0 - 5 + 5 * 3} <= -22 * -1 ∧ F V [5 + 2 + 1 - 3 = 2 - 2 * -3] V ! (!F ∧ T ∧ !F)}
! {2 * {1 - 5 + 5 * 3} <= -22 * -1 ∧ F V [5 + 2 + 1 - 3 = 2 - 2 * -3] V ! (!F ∧ T ∧ !F)}
! {2 * {1 - 5 + 15} <= -22 * -1 ∧ F V [5 + 2 + 1 - 3 = 2 - 2 * -3] V ! (!F ∧ T ∧ !F)}
! {2 * 11 <= -22 * -1 ∧ F V [5 + 2 + 1 - 3 = 2 - 2 * -3] V ! (!F ∧ T ∧ !F)}
! {2 * 11 <= -22 * -1 ∧ F V [ 5 + 2 + 1 - 3 = 2 - 2 * -3 ] V ! (!F ∧ T ∧ !F)}
! {2 * 11 <= -22 * -1 ∧ F V [ 5 = 2 - -6 ] V ! (!F ∧ T ∧ !F)}
! {2 * 11 <= -22 * -1 ∧ F V [ 5 = 2 - -6 ] V ! (!F ∧ T ∧ !F)}
! {2 * 11 <= -22 * -1 ∧ F V [ 5 = 2 + 6 ] V ! (!F ∧ T ∧ !F)}
! {2 * 11 <= -22 * -1 ∧ F V [ 5 = 8 ] V ! (!F ∧ T ∧ !F)}
! {2 * 11 <= -22 * -1 ∧ F V [ 5 = 8 ] V ! (!F ∧ T ∧ !F)}
! {2 * 11 <= -22 * -1 ∧ F V F V ! (!F ∧ T ∧ !F)}
! {2 * 11 <= -22 * -1 ∧ F V F V ! ( !F ∧ T ∧ !F )}
! {2 * 11 <= -22 * -1 ∧ F V F V ! ( T ∧ T ∧ T )}
! {2 * 11 <= -22 * -1 ∧ F V F V ! ( T ∧ T )}
! {2 * 11 <= -22 * -1 ∧ F V F V !T }
! {2 * 11 <= -22 * -1 ∧ F V F V F }
! { 2 * 11 <= -22 * -1 ∧ F V F V F }
! { 22 <= -22 * -1 ∧ F V F V F }
! { 22 <= 22 ∧ F V F V F }
! { T ∧ F V F V F }
! { F V F V F }
! { F V F }
! { F }
!F
T
```

Figura 2.59. Orden de aplicación de las precedencias explicita y con operadores aritméticos, relacionales y lógicos
Fuente: Aguilera y Bejarano, 2018

3.5.9 Ejercicios

Resuelva cada una de las expresiones presentadas a continuación, elabore el desarrollo de cada expresión para llegar a su respuesta, tal y como se hizo en los ejemplos. Tome en cuenta los siguientes puntos:

- `A=T` (`true`), `B=F (false)`, `C=F (false)`, `D=T (true)`, `X = 5,Y = 7`
- Si en la expresión aparece una `T` o una `F` asuma que significan `true` (`verdadero`) y `false` (`falso`) respectivamente.
- Utilice máximo dos decimales para expresar los datos que así lo requieran.

a) 2 + 1 + 3 – 2 * 5 / 2

b) 2 * 5 / 2 < 50

c) ! (D ∧ (5 > 15 V 30 < 15 ∧ A) ∧ (C V ! ((23 * 6 + 2 - 1) < 100)))

d) (950 + 30 * 90 + 100) / (10 * (4 / 2) * 5^2 * 3)

e) ¬ (1 > 1 ∧ 3 < 5 ∧ (6 – 4 * 4 / 2 = 2) ∨ ((2 > 24) ∨ 3^2 > 10))

f) ! ((X + Y) >= – ((Y MOD 2) * (4 * Y)))

g) ! (“a” = ”A” V (20 * 3 * 5 > 10 * 40) ∧ (30^3 > 15^2))

h) (45 >= 20 – 5 * 2) ∧ ¬ ((35 – 45 / 5 * 3 > 45) V (50 / 5 * 3 <> 10)) V (17 – 2 <= 16)

i) !{[(1 + 2 * 2 < 3 – 2^3 / 2)] V [(2 – $^{-}$1 * 3 + $^{-}$1) > (4 * 0) V (15 % 4) = (3^0 * 3)] ∧![(1 * 2 / 2) <= (1 + 3 * 2)]}

j) ⌐ [10 / 2 + 10 – 14 < $^{-}$5 ∧ (2^3 / 8 – 5 < 0 V (2 * 2 + 11 < 70 – 11) V B)] V A ∧ ⌐ B

k) {3 * [30 / 4 + 5 + 14 – (6 / 1 + $^{-}$25 * (2 + 11 * 7 – 15) – 5) / 12]} < 1 V T

l) ¬ (1 > 1 ∧ 3 < 5 ∧ (4^2 * $^{-}$1 > 2) V (F V 9 < $^{-}$10))

3.5.10 Respuestas

a) $2 + 1 + 3 - \underline{2 * 5} / 2$

$2 + 1 + 3 - \underline{10 / 2}$

$\underline{2 + 1 + 3 - 5}$

R/ 1

b) $\underline{2 * 5} / 2 < 50$

$\underline{10 / 2} < 50$

$\underline{5 < 50}$

R/ T

c) ! (D ∧ (5 > 15 V 30 < 15 ∧ A) ∧ (C V ! ((23 * 6 + 2 - 1) < 100)))

! (T ∧ (5 > 15 V 30 < 15 ∧ T) ∧ (F V ! ((23 * 6 + 2 - 1) < 100)))

! (T ∧ (5 > 15 V 30 < 15 ∧ T) ∧ (F V ! ((138 + 2 - 1) < 100)))

! (T ∧ (F V 30 < 15 ∧ T) ∧ (F V ! (139 < 100)))

! (T ∧ (F V F ∧ T) ∧ (F V ! F))

! (T ∧ (F V F) ∧ (F V T))

! (T ∧ F ∧ T)

! (F ∧ T)

! F

R/ T

d) (950 + 30 * 90 + 100) / (10 * (4 / 2) * 5^2 * 3)

(950 + 2700 + 100) / (10 * 2 * 5^2 * 3)

3750 / (10 * 2 * 25 * 3)

3750 / 1500

R/ 2.5

e) ¬ (1 > 1 ∧ 3 < 5 ∧ (6 – 4 * 4 / 2 = 2) ∨ ((2 > 24) ∨ 3^2 > 10))

¬ (1 > 1 ∧ 3 < 5 ∧ (6 – 16 / 2 = 2) ∨ (F ∨ 3^2 > 10))

¬ (1 > 1 ∧ 3 < 5 ∧ (6 – 8 = 2) ∨ (F ∨ 9 > 10))

¬ (1 > 1 ∧ 3 < 5 ∧ (-2 = 2) ∨ (F ∨ F))

¬ (1 > 1 ∧ 3 < 5 ∧ F ∨ F)

¬ (F ∧ 3 < 5 ∧ F ∨ F)

¬ (F ∧ T ∧ F ∨ F)

¬ (F ∧ F ∨ F)

¬ (F ∧ F ∨ F)

¬ (F ∨ F)

¬ F

R/ T

f) ! ((X + Y) >= – ((Y MOD 2) * (4 * Y)))

! ((5 + 7) >= – ((7 MOD 2) * (4 * 7)))

! (12 >= - (1 * 28))

! (12 >= -28)

!T

R/ F

g) ! ("a" = "A" V (20 * 3 * 5 > 10 * 40) ∧ (30^3 > 15^2))

! ("a" = "A" V (300 > 10 * 40) ∧ (27000 > 15^2))

! ("a" = "A" V (300 > 400) ∧ (27000 > 225))

! ("a" = "A" V F ∧ T)

! (F V F ∧ T)

! (F V F)

!F

R/ T

h) (45 >= 20 – 5 * 2) ∧ ¬ ((35 – 45 / 5 * 3 > 45) V (50 / 5 * 3 <> 10)) V (17 – 2 <= 16)

(45 >= 20 – 10) ∧ ¬ ((35 – 9 * 3 > 45) V (10 * 3 <> 10)) V (15 <= 16)

(45 >= 10) ∧ ¬ ((35 – 27 > 45) V (30 <> 10)) V T

T ∧ ¬ ((8 > 45) V T) V T

T ∧ ¬ (F V T) V T

T ∧ ¬ T V T

T ∧ F V T

F V T

R/ T

i) !{(1 + 2 * 2 < 3 - 2^3 / 2) V [(2 – ¯1 * 3 + ¯1) > (4 * 0) V (15 % 4) = (3^0 * 3)] ∧![(1 * 2 / 2) <= (1 + 3 * 2)]}

!{(1 + 2 * 2 < 3 - 8 / 2) V [(2 + 1 * 3 + ¯1) > 0 V 3 = (1 * 3)] ∧![(2 / 2) <= (1 + 9)]}

!{(1 + 4 < 3 - 4) V [(2 + 3 + ¯1) > 0 V 3 = 3] ∧![1 <= 10]}

!{ (5 < ¯1) V [4 > 0 V 3 = 3] ∧!T }

!{F V [T V T] ∧!T }

!{ F V T ∧!T }

!{ F V T ∧F }

!{ F V F }

!F

R/ T

j) ⌐[10 / 2 + 10 - 14 < ¯5 ∧ (2^3 / 8 - 5 < 0 V (2 * 2 + 11 < 70 – 11) V B)] V A ∧ ⌐B

⌐[10 / 2 + 10 - 14 < ¯5 ∧ (2^3 / 8 - 5 < 0 V (2 * 2 + 11 < 70 – 11) V F)] V T ∧ ⌐F

⌐[10 / 2 + 10 - 14 < ¯5 ∧ (2^3 / 8 - 5 < 0 V (4 + 11 < 70 – 11) V F)] V T ∧ ⌐F

⌐[10 / 2 + 10 - 14 < ¯5 ∧ (2^3 / 8 - 5 < 0 V (15 < 59) V F)] V T ∧ ⌐F

$\neg [10 / 2 + 10 - 14 < {}^{-}5 \wedge (2\text{^}3 / 8 - 5 < 0 \vee T \vee F)] \vee T \wedge \neg F$

$\neg [10 / 2 + 10 - 14 < {}^{-}5 \wedge (8 / 8 - 5 < 0 \vee T \vee F)] \vee T \wedge \neg F$

$\neg [10 / 2 + 10 - 14 < {}^{-}5 \wedge (1 - 5 < 0 \vee T \vee F)] \vee T \wedge \neg F$

$\neg [10 / 2 + 10 - 14 < {}^{-}5 \wedge ({}^{-}4 < 0 \vee T \vee F)] \vee T \wedge \neg F$

$\neg [10 / 2 + 10 - 14 < {}^{-}5 \wedge (T \vee T \vee F)] \vee T \wedge \neg F$

$\neg [10 / 2 + 10 - 14 < {}^{-}5 \wedge T] \vee T \wedge \neg F$

$\neg [5 + 10 - 14 < {}^{-}5 \wedge T] \vee T \wedge \neg F$

$\neg [1 < {}^{-}5 \wedge T] \vee T \wedge \neg F$

$\neg [F \wedge T] \vee T \wedge \neg F$

$\neg F \vee T \wedge \neg F$

$T \vee T \wedge \neg F$

$T \vee T \wedge T$

$T \vee T$

R/ T

k) $\{3 * [30 / 4 + 5 + 14 - (6 / 1 + {}^{-}25 * (2 + \underline{11 * 7} - 15) - 5) / 12]\} < 1 \vee T$

$\{3 * [30 / 4 + 5 + 14 - (6 / 1 + {}^{-}25 * (\underline{2 + 77 - 15}) - 5) / 12]\} < 1 \vee T$

$\{3 * [30 / 4 + 5 + 14 - (\underline{6 / 1} + {}^{-}25 * 64 - 5) / 12]\} < 1 \vee T$

$\{3 * [30 / 4 + 5 + 14 - (6 + \underline{{}^{-}25 * 64} - 5) / 12]\} < 1 \vee T$

$\{3 * [30 / 4 + 5 + 14 - (\underline{6 + {}^{-}1600 - 5}) / 12]\} < 1 \vee T$

$\{3 * [30 / 4 + 5 + 14 \underline{- {}^{-}1599} / 12]\} < 1 \vee T$

$\{3 * [\underline{30 / 4} + 5 + 14 + 1599 / 12]\} < 1 \vee T$

{3 * [7.5 + 5 + 14 + 1599 / 12]} < 1 V T

{3 * [7.5 + 5 + 14 +133.25]} < 1 V T

{ 3 * 159.75 } < 1 V T

479.25 < 1 V T

F V T

R/ T

l) ¬ (1 > 1 ∧ 3 < 5 ∧ (4^2 * -1 > 2) V (F V 9 < -10))

¬ (1 > 1 ∧ 3 < 5 ∧ (16 * -1 > 2) V (F V 9 < -10))

¬ (1 > 1 ∧ 3 < 5 ∧ (-16 > 2) V (F V 9 < -10))

¬ (1 > 1 ∧ 3 < 5 ∧ F V (F V 9 < -10))

¬ (1 > 1 ∧ 3 < 5 ∧ F V (F V F))

¬ (1 > 1 ∧ 3 < 5 ∧ F V F)

¬ (F ∧ 3 < 5 ∧ F V F)

¬ (F ∧ T ∧ F V F)

¬ (F ∧ F V F)

¬ (F V F)

¬ F

R/ T