Guía temas para ingreso a la carrera de Ingeniería en Sistemas Computacionales en ESCOM – IPN

Sección Física

I. ACERCA DE LA FÍSICA. 1.

INTRODUCCIÓN.

1.1. Importancia de la física en la naturaleza y en la vida cotidiana (ciencia, tecnología y sociedad).

1.2. Sistemas físicos.

1.3. Magnitudes y variables físicas.

1.4. Elementos teóricos y experimentales de la metodología de la física: Planteamiento de problemas, formulación y prueba de hipótesis y elaboración de modelos.

1.5. Ejemplos de hechos históricos.

II. FENÓMENOS MECÁNICOS.

1 . PRIMERA LEY DE NEWTON.

1.1. Sistemas o marcos de referencia en reposo.

1.2. Interacciones y fuerzas, aspectos cualitativos.

1.3. Fuerza resultante cero, (vectores desde un punto de vista operativo, diferencia entre vector y escalar), 1ª Ley de Newton y Movimiento Rectilíneo Uniforme.

1.4. Masa inercial y momento lineal.

2. SEGUNDA LEY DE NEWTON.

2.2. Fuerza constante en la dirección del movimiento y MRUA.

2.3. Diferencias entre el MRU y el MRUA.

2.4. Fuerza constante con dirección perpendicular al movimiento: MCU.

2.5. Resolución de problemas relativos al MRU, MRUA y MCU.

3. TERCERA LEY DE NEWTON.

3.1. Tercera ley de Newton.

3.2. Conservación de ímpetu.

3.3. Colisiones: Elásticas, inelásticas y elásticas perfectas.

4. GRAVITACIÓN UNIVERSAL.

4.1. Caída libre y peso.

4.2. Ley de la Gravitación Universal.

4.3. Interacciones gravitacionales y movimiento de planetas, satélites y cometas. Leyes de Kepler.

5. ENERGÍA MECÁNICA Y TRABAJO.

5.1. Energía y tipos de energía.

5.2. Energía cinética y energía potencial (gravitacional y elástica).

5.3. Transferencia de energía mecánica.

5.4. Trabajo, Relación del trabajo con cambio de energía cinética y potencial.

5.5. Potencia mecánica.

5.6. Conservación de la energía mecánica.

5.7. Procesos disipativos y fricción.

III. FENÓMENOS ELECTROMAGNÉTICOS.

10. CARGAS ELÉCTRICAS EN REPOSO Y CAMPO ELÉCTRICO.

10.1. Carga eléctrica: Formas en que se genera y su conservación.

10.2. Ley de Coulomb y fuerza de Coulomb.

10.3. Campos eléctricos, potencial eléctrico, energía potencial eléctrica y diferencia de potencial.

10.4. Comente eléctrica, resistencia y Ley de Ohm.

10.5. Circuitos y potencia eléctricos (Consumo).

11. CARGAS ELÉCTRICAS EN MOVIMIENTO Y CAMPO MAGNÉTICO.

11.1. Experimento de Oersted.

11.2. Polos magnéticos e Imanes.

11.3. Fuerza magnética en cargas y en comentes.

11.4. Ley de Ampere y el experimento de Oersted.

11.5. Movimiento de imanes, campos magnéticos variables e inducción magnética.

11.6. Ley de Faraday y Ley de Lenz. Corriente alterna.

11.7. Motores y generadores eléctricos.

11.8. Interacción de los campos eléctricos y magnéticos.

11.9. Representación gráfica de las ondas electromagnéticas.

11.10. Espectro electromagnético: Características de los siete tipos de ondas.

11.11. Aplicaciones de cada tipo de las ondas electromagnéticas.

IV. SISTEMAS ELECTROMECÁNICOS Y ELECTRÓNICOS.

16. ELECTROMAGNETISMO.

16.1 Carga eléctrica y formas de adquirir carga eléctrica.

16.2 Ley de Coulomb y definición de campo eléctrico.

16.3 Campo eléctrico y ley de Gauss.

16.4 Potencial eléctrico y energía potencial eléctrica.

16.5 Diferencia de potencial:

16.6 Capacitancia:

16.7 Corriente eléctrica:

16.8 Potencia (Efecto Joule):

16.9 Ley de Ohm:

16.10 Campo magnético y ley de Gauss.

16.11 Ley de Ampere:

16.12 Ley de Faraday:

16.13 Ley de Ampere-Maxwell:

16.14. Elementos utilizados en electrónica: Resistor, capacitor, inductor, diodo, transistor y circuito integrados.

16.15. Espectro electromagnético

Sección de Algebra

Introducción

CAPITULO 1: INTRODUCCION

1. INTRODUCCION

El lenguaje de programación C está caracterizado por ser de uso general, con una sintaxis sumamente compacta y de alta portabilidad.
Es común leer que se lo caracteriza como un lenguaje de "bajo nivel". No debe confundirse el término "bajo" con "poco", ya que el significado del mismo es en realidad "profundo", en el sentido que C maneja los elementos básicos presentes en todas las computadoras: caracteres, números y direcciones.
Esta particularidad, junto con el hecho de no poseer operaciones de entrada-salida, manejo de arreglo de caracteres, de asignación de memoria, etc , puede al principio parecer un grave defecto; sin embargo el hecho de que estas operaciones se realicen por medio de llamadas a Funciones contenidas en Librerías externas al lenguaje en sí, es el que confiere al mismo su alto grado de portabilidad, independizándolo del "Hardware" sobre el cual corren los programas, como se irá viendo a lo largo de los siguientes capítulos.
La descripción del lenguaje se realiza siguiendo las normas del ANSI C, por lo tanto, todo lo expresado será utilizable con cualquier compilador que se adopte; sin embargo en algunos casos particulares se utilizaron funciones Compilador ó Sistema Operativo-dependientes, explicitándose en estos casos la singularidad de las mismas.

2. ANATOMIA DE UN PROGRAMA C

Siguiendo la tradición, la mejor forma de aprender a programar en cualquier lenguaje es editar, compilar, corregir y ejecutar pequeños programas descriptivos. Analicemos por lo tanto el primer ejemplo :

EJEMPLO 1

```
#include <stdio.h>
main()

{

        printf("Bienvenido a la Programación en lenguaje C \n");

        return 0;

}
```

FUNCION main()

Dejemos de lado por el momento el análisis de la primer linea del programa, y pasemos a la segunda.
La función main() indica donde empieza el programa, cuyo cuerpo principal es un conjunto de sentencias delimitadas por dos llaves, una inmediatamente después de la declaración main() " { ", y otra que finaliza el listado " } ". Todos los programas C arrancan del mismo punto: la primer sentencia dentro de dicha función, en este caso printf ("......").
En el EJEMPLO 1 el programa principal está compuesto por sólo dos sentencias: la primera es un llamado a una función denominada printf(), y la segunda, return, que finaliza el programa retornando al Sistema Operativo.
Recuérdese que el lenguaje C no tiene operadores de entrada-salida por lo que para escribir en video es necesario llamar a una función externa. En este caso se invoca a la

función printf(argumento) existente en la Librería y a la cual se le envía como argumento aquellos caracteres que se desean escribir en la pantalla. Los mismos deben estar delimitados por comillas. La secuencia \n que aparece al final del mensaje es la notación que emplea C para el caracter "nueva línea" que hace avanzar al cursor a la posición extrema izquierda de la línea siguiente. Más adelante analizaremos otras secuencias de escape habituales.
La segunda sentencia (return 0) termina el programa y devuelve un valor al Sistema operativo, por lo general cero si la ejecución fué correcta y valores distintos de cero para indicar diversos errores que pudieron ocurrir. Si bien no es obligatorio terminar el programa con un return, es conveniente indicarle a quien lo haya invocado, sea el Sistema Operativo o algún otro programa, si la finalización ha sido exitosa, o no. De cualquier manera en este caso, si sacamos esa sentencia el programa correrá exactamente igual, pero al ser compilado, el compilador nos advertirá de la falta de retorno.
Cada sentencia de programa queda finalizada por el terminador "; ", el que indica al compilador el fin de la misma. Esto es necesario ya que, sentencias complejas pueden llegar a tener más de un renglón, y habrá que avisarle al compilador donde terminan.
Es perfectamente lícito escribir cualquier sentencia abarcando los renglones que la misma necesite, por ejemplo podría ser:

```
printf("Bienvenido a la Programacion"

                        "en lenguaje C \n");
```

3. ENCABEZAMIENTO

Las líneas anteriores a la función main() se denominan ENCABEZAMIENTO (HEADER) y son informaciones que se le suministran al Compilador.
La primera línea del programa está compuesta por una directiva: " #include " que implica la orden de leer un archivo de texto especificado en el nombre que sigue a la misma (<stdio.h>) y reemplazar esta línea por el contenido de dicho archivo.
En este archivo están incluidas declaraciones de las funciones luego llamadas por el programa (por ejemplo printf()) necesarias para que el compilador las procese. Por ahora no nos preocupemos por el contenido del archivo ya que más adelante, en el capítulo de funciones, analizaremos exhaustivamente dichas declaraciones.
Hay dos formas distintas de invocar al archivo, a saber, si el archivo invocado está delimitado por comillas (por ejemplo "stdio.h") el compilador lo buscará en el directorio activo en el momento de compilar y si en cambio se lo delimita con los signos <.......> lo buscará en algun otro directorio, cuyo nombre habitualmente se le suministra en el momento de la instalación del compilador en el disco (por ejemplo C:\TC\INCLUDE).
Por lo general estos archivos son guardados en un directorio llamado INCLUDE y el nombre de los mismos está terminado con la extensión .h.
La razón de la existencia de estos archivos es la de evitar la repetición de la escritura de largas definiciones en cada programa.
Nótese que la directiva "#include" no es una sentencia de programa sino una orden de que se copie literalmente un archivo de texto en el lugar en que ella está ubicada ,por lo que no es necesario terminarla con "; ".

4. COMENTARIOS

La inclusión de comentarios en un programa es una saludable práctica, como lo reconocerá cualquiera que haya tratado de leer un listado hecho por otro programador ó por sí mismo, varios meses atrás. Para el compilador, los comentarios son inexistentes, por lo que no generan lineas de código, permitiendo abundar en ellos tanto como se desee.

En el lenguaje C se toma como comentario todo caracter interno a los símbolos: /* */ . Los comentarios pueden ocupar uno o más renglones, por ejemplo:
COMENTARIOS

```
/* este es un comentario corto */

/* este otro

              es mucho

              más largo

              que el anterior */
```

Todo caracter dentro de los símbolos delimitadores es tomado como comentario incluyendo a " * " ó " (" , etc.

CAPITULO 2: VARIABLES Y CONSTANTES

1. DEFINICION DE VARIABLES

Si yo deseara imprimir los resultados de multiplicar un número fijo por otro que adopta valores entre 0 y 9 , la forma normal de programar esto sería crear una CONSTANTE para el primer número y un par de VARIABLES para el segundo y para el resultado del producto. Una variable , en realidad , no es más que un nombre para identificar una (o varias) posiciones de memoria donde el programa guarda los distintos valores de una misma entidad . Un programa debe DEFINIR a todas las variables que utilizará , antes de comenzar a usarlas , a fin de indicarle al compilador de que tipo serán , y por lo tanto cuanta memoria debe destinar para albergar a cada una de ellas. Veamos el EJEMPLO 2:

EJEMPLO 2

```
#include <stdio.h>
main()
{
  int multiplicador;         /* defino multiplicador como un entero */
  int multiplicando;         /* defino multiplicando como un entero */
  int resultado;             /* defino resultado como un entero */
  multiplicador = 1000 ;     /* les asigno valores */
  multiplicando = 2 ;
  resultado = multiplicando * multiplicador ;
  printf("Resultado = %d\n", resultado);         /* muestro el resultado */
  return 0;
}
```

En las primeras líneas de texto dentro de main() defino mis variables como números enteros , es decir del tipo "int" seguido de un identificador (nombre) de la misma . Este identificador puede tener la cantidad de caracteres que se desee , sin embargo de acuerdo al Compilador que se use , este tomará como significantes sólo los primeros n de ellos ; siendo por lo general n igual a 32 . Es conveniente darle a los identificadores de las variables , nombres que tengan un significado que luego permita una fácil lectura del programa. Los identificadores deben comenzar con una letra ó con el símbolo de subrayado "_" , pudiendo continuar con cualquier otro caractér alfanumérico ó el símbolo "_" . El único símbolo no alfanumérico aceptado en un nombre es el "_" . El lenguaje C es sensible al tipo de letra usado ; así tomará como variables distintas a una llamada "variable" , de otra escrita como "VARIABLE". Es una convención entre los programadores de C escribir los nombres de las variables y las funciones con minúsculas, reservando las mayúsculas para las constantes.

El compilador dará como error de "Definición incorrecta" a la definición de variables con nombres del tipo de :

4pesos $variable   primer-variable !variable etc.etc

NOTA: Los compiladores reservan determinados términos ó palabras claves (Keywords) para el uso sintáctico del lenguaje, tales como: asm, auto, break, case, char, do, for, etc. Si bien estas palabras están definidas para el ANSI C, los distintos compiladores extienden esta definición a OTROS términos, por lo que es aconsejable leer la tabla completa de palabras reservadas del compilador que se vaya a usar, para no utilizarlas en nombres de variables.

Vemos en las dos lineas subsiguientes a la definición de las variables, que puedo ya asignarles valores (1000 y 2) y luego efectuar el cálculo de la variable "resultado". Si prestamos ahora atención a la función printf(), ésta nos mostrará la forma de visualizar el valor de una variable. Insertada en el texto a mostrar, aparece una secuencia de control de impresión "%d" que indica, que en el lugar que ella ocupa, deberá ponerse el contenido de la variable (que aparece luego de cerradas las comillas que marcan la finalización del texto , y separada del mismo por una coma) expresado como un un número entero decimal. Así, si compilamos y corremos el programa, obtendremos una salida:

SALIDA DEL EJEMPLO 2

Resultado = 2000

2. INICIALIZACION DE VARIABLES

Las variables del mismo tipo pueden definirse mediante una definición múltiple separándolas mediante " , " a saber :
int multiplicador, multiplicando, resultado;
Esta sentencia es equivalente a las tres definiciones separadas en el ejemplo anterior.
Las variables pueden también ser inicializadas en el momento de definirse .
int multiplicador = 1000, multiplicando = 2, resultado;
De esta manera el EJEMPLO 2 podría escribirse:

EJEMPLO 2 BIS

```
#include <stdio.h>

main()

{

  int multiplicador=1000 , multiplicando=2 ;

  printf("Resultado = %d\n", multiplicando * multiplicador);

  return 0;

}
```

Obsérvese que en la primer sentencia se definen e inicializan simultaneamente ambas variables.La variable "resultado" la hemos hecho desaparecer ya que es innecesaria. Si analizamos la función printf() vemos que se ha reemplazado "resultado" por la operación entre las otras dos variables. Esta es una de las particularidades del lenguaje C : en los parámetros pasados a las funciones pueden ponerse operaciones (incluso llamadas a otras funciones) , las que se realizan ANTES de ejecutarse la función , pasando finalmente a esta el valor resultante de las mismas. El EJEMPLO 2 funciona exactamente igual que antes pero su código ahora es mucho más compacto y claro.

3. TIPOS DE VARIABLES

VARIABLES DEL TIPO ENTERO

En el ejemplo anterior definimos a las variables como enteros (int).
De acuerdo a la cantidad de bytes que reserve el compilador para este tipo de variable,

queda determinado el "alcance" ó máximo valor que puede adoptar la misma.
Debido a que el tipo int ocupa dos bytes su alcance queda restringido al rango entre -32.768 y +32.767 (incluyendo 0).
En caso de necesitar un rango más amplio, puede definirse la variable como "long int nombre_de_variable" ó en forma más abreviada "long nombre_de_variable"
Declarada de esta manera, nombre_de_variable puede alcanzar valores entre -2.347.483.648 y +2.347.483.647.
A la inversa, si se quisiera un alcance menor al de int, podría definirse "short int " ó simplemente "short", aunque por lo general, los compiladores modernos asignan a este tipo el mismo alcance que "int".
Debido a que la norma ANSI C no establece taxativamente la cantidad de bytes que ocupa cada tipo de variable, sino tan sólo que un "long" no ocupe menos memoria que un "int" y este no ocupe menos que un "short",los alcances de los mismos pueden variar de compilador en compilador , por lo que sugerimos que confirme los valores dados en este parágrafo (correspondientes al compilador de Borland C++) con los otorgados por su compilador favorito.
Para variables de muy pequeño valor puede usarse el tipo "char" cuyo alcance está restringido a -128, +127 y por lo general ocupa un único byte.
Todos los tipos citados hasta ahora pueden alojar valores positivos ó negativos y, aunque es redundante, esto puede explicitarse agregando el calificador "signed" delante; por ejemplo:

signed int

signed long

signed long int

signed short

signed short int

signed char

Si en cambio, tenemos una variable que sólo puede adoptar valores positivos (como por ejemplo la edad de una persona) podemos aumentar el alcance de cualquiera de los tipos , restringiéndolos a que sólo representen valores sin signo por medio del calificador "unsigned" . En la TABLA 1 se resume los alcances de distintos tipos de variables enteras

TABLA 1 VARIABLES DEL TIPO NUMERO ENTERO

TIPO	BYTES	VALOR MINIMO	VALOR MAXIMO
signed char	1	-128	127
unsigned char	1	0	255
signed short	2	-32.768	+32.767
unsigned short	2	0	+65.535
signed int	2	-32.768	+32.767
unsigned int	2	0	+65.535
signed long	4	-2.147.483.648	+2.147.483.647
unsigned long	4	0	+4.294.967.295

NOTA: Si se omite el calificador delante del tipo de la variable entera, éste se adopta por omisión (default) como "signed".

VARIABLES DE NUMERO REAL O PUNTO FLOTANTE

Un número real ó de punto flotante es aquel que además de una parte entera, posee fracciones de la unidad. En nuestra convención numérica solemos escribirlos de la siguiente manera : 2,3456, lamentablemente los compiladores usan la convención del PUNTO decimal (en vez de la coma) . Así el numero Pi se escribirá : 3.14159 Otro formato de escritura, normalmente aceptado, es la notación científica. Por ejemplo podrá escribirse 2.345E+02, equivalente a 2.345 * 100 ó 234.5

De acuerdo a su alcance hay tres tipos de variables de punto flotante , las mismas están descriptas en la TABLA 2

TABLA 2 TIPOS DE VARIABLES DE PUNTO FLOTANTE

TIPO	BYTES	VALOR MINIMO	VALOR MAXIMO
float	4	3.4E-38	3.4E+38
double	8	1.7E-308	1.7E+308
long double	10	3.4E-4932	3.4E+4932

Las variables de punto flotante son SIEMPRE con signo, y en el caso que el exponente sea positivo puede obviarse el signo del mismo.

5. CONVERSION AUTOMATICA DE TIPOS

Cuando dos ó mas tipos de variables distintas se encuentran DENTRO de una misma operación ó expresión matemática , ocurre una conversión automática del tipo de las variables. En todo momento de realizarse una operación se aplica la siguiente secuencia de reglas de conversión (previamente a la realización de dicha operación):

- 1) Las variables del tipo char ó short se convierten en int
- 2) Las variables del tipo float se convierten en double
- 3) Si alguno de los operandos es de mayor precisión que los demás , estos se convierten al tipo de aquel y el resultado es del mismo tipo.
- 4) Si no se aplica la regla anterior y un operando es del tipo unsigned el otro se convierte en unsigned y el resultado es de este tipo.

Las reglas 1 a 3 no presentan problemas, sólo nos dicen que previamente a realizar alguna operación las variables son promovidas a su instancia superior. Esto no implica que se haya cambiado la cantidad de memoria que las aloja en forma permanente

Otro tipo de regla se aplica para la conversión en las asignaciones.

Si definimos los términos de una asignación como,"lvalue" a la variable a la izquierda del signo igual y "rvalue" a la expresión a la derecha del mismo, es decir:

"lvalue" = "rvalue" ;

Posteriormente al cálculo del resultado de "rvalue" (de acuerdo con las reglas antes descriptas), el tipo de este se iguala al del "lvalue". El resultado no se verá afectado si el tipo de "lvalue" es igual ó superior al del "rvalue", en caso contrario se efectuará un truncamiento ó redondeo, segun sea el caso.

Por ejemplo, el pasaje de float a int provoca el truncamiento de la parte fraccionaria, en cambio de double a float se hace por redondeo.

5. ENCLAVAMIENTO DE CONVERSIONES (casting)

Las conversiones automáticas pueden ser controladas a gusto por el programador, imponiendo el tipo de variable al resultado de una operación. Supongamos por ejemplo tener:

```
double  d , e , f = 2.33 ;

int i = 6 ;

e = f * i ;

d = (int) ( f * i ) ;
```

En la primer sentencia calculamos el valor del producto (f * i) , que según lo visto anteriormente nos dará un double de valor 13.98 , el que se ha asignado a e. Si en la variable d quisiéramos reservar sólo el valor entero de dicha operación bastará con anteponer, encerrado entre paréntesis, el tipo deseado. Así en d se almacenará el número 13.00.

También es factible aplicar la fijación de tipo a una variable, por ejemplo obtendremos el mismo resultado, si hacemos:

```
d = (int) f * i ;
```

En este caso hemos convertido a f en un entero (truncando sus decimales)

6. VARIABLES DE TIPO CARACTER

El lenguaje C guarda los caracteres como números de 8 bits de acuerdo a la norma ASCII extendida , que asigna a cada caracter un número comprendido entre 0 y 255 (un byte de 8 bits) Es común entonces que las variables que vayan a alojar caracteres sean definidas como:

```
char c ;
```

Sin embargo, también funciona de manera correcta definirla como

```
int c ;
```

Esta última opción desperdicia un poco más de memoria que la anterior ,pero en algunos casos particulares presenta ciertas ventajas . Pongamos por caso una función que lee un archivo de texto ubicado en un disco. Dicho archivo puede tener cualquier caracter ASCII de valor comprendido entre 0 y 255. Para que la función pueda avisarme que el archivo ha finalizado deberá enviar un número NO comprendido entre 0 y 255 (por lo general se usa el -1 , denominado EOF, fin de archivo ó End Of File), en este caso dicho número no puede ser mantenido en una variable del tipo char, ya que esta sólo puede guardar entre 0 y 255 si se la define unsigned ó no podría mantener los caracteres comprendidos entre 128 y 255 si se la define signed (ver TABLA 1). El problema se obvia facilmente definiéndola como int.

Las variables del tipo caractér también pueden ser inicializadas en su definición, por ejemplo es válido escribir:

```
char c = 97 ;
```

para que c contenga el valor ASCII de la letra "a", sin embargo esto resulta algo engorroso , ya que obliga a recordar dichos códigos . Existe una manera más directa de asignar un caractér a una variable ; la siguiente inicialización es idéntica a la anterior :

```
char c = 'a' ;
```

Es decir que si delimitamos un caracter con comilla simple , el compilador entenderá que debe suplantarlo por su correspondiente código numérico .

Lamentablemente existen una serie de caracteres que no son imprimibles , en otras palabras que cuando editemos nuestro programa fuente (archivo de texto) nos resultará difícil de asignarlas a una variable ya que el editor las toma como un COMANDO y no como un caracter . Un caso típico sería el de "nueva linea" ó ENTER .

Con el fin de tener acceso a los mismos es que aparecen ciertas secuencias de escape convencionales . Las mismas estan listadas en la TABLA 3 y su uso es idéntico al de los caracteres normales , asi para resolver el caso de una asignación de "nueva linea " se escribirá:

char c = '\n' ;   /* secuencia de escape */

TABLA 3 SECUENCIAS DE ESCAPE

CODIGO	SIGNIFICADO	VALOR ASCII (decimal)	VALOR ASCII (hexadecimal)
'\n'	nueva línea	10	0x0A
'\r'	retorno de carro	13	0x0D
'\f'	nueva página	2	x0C
'\t'	tabulador horizontal	9	0x09
'\b'	retroceso (backspace)	8	0x08
'\''	comilla simple	39	0x27
'\"'	comillas	4	0x22
'\\ '	barra	92	0x5C
'\? '	interrogación	63	0x3F
'\nnn'	cualquier caracter (donde nnn es el código ASCII expresado en octal)		
'\xnn'	cualquier caracter (donde nn es el código ASCII expresado en hexadecimal)		

7. TAMAÑO DE LAS VARIABLES (sizeof)

En muchos programas es necesario conocer el tamaño (cantidad de bytes) que ocupa una variable, por ejemplo en el caso de querer reservar memoria para un conjunto de ellas. Lamentablemente, como vimos anteriormente este tamaño es dependiente del compilador que se use, lo que producirá, si definimos rigidamente (con un número dado de bytes) el espacio requerido para almacenarlas, un problema serio si luego se quiere compilar el programa con un compilador distinto del original

Para salvar este problema y mantener la portabilidad, es conveniente que cada vez que haya que referirse al TAMAÑO en bytes de las variables, se lo haga mediante un operador llamado "sizeof" que calcula sus requerimientos de almacenaje

Está también permitido el uso de sizeof con un tipo de variable, es decir:

sizeof(int)

sizeof(char)

sizeof(long double) , etc.

8. DEFINICION DE NUEVOS TIPOS (typedef)

A veces resulta conveniente crear otros tipos de variables , ó redefinir con otro nombre las existentes , esto se puede realizar mediante la palabra clave "typedef" , por ejemplo:

typedef unsigned long double enorme ;

A partir de este momento ,las definiciones siguientes tienen idéntico significado:

unsigned long double nombre_de_variable ;

enorme nombre_de_variable ;

9. CONSTANTES

Aquellos valores que , una vez compilado el programa no pueden ser cambiados , como por ejemplo los valores literales que hemos usado hasta ahora en las inicializaciones de las variables (1000 , 2 , 'a' , '\n' , etc), suelen denominarse CONSTANTES .
Como dichas constantes son guardadas en memoria de la manera que al compilador le resulta más eficiente suelen aparecer ciertos efectos secundarios , a veces desconcertantes , ya que las mismas son afectadas por las reglas de RECONVERSION AUTOMATICA DE TIPO vista previamente.
A fin de tener control sobre el tipo de las constantes, se aplican la siguientes reglas :

- Una variable expresada como entera (sin parte decimal) es tomada como tal salvo que se la siga de las letras F ó L (mayúsculas ó minúsculas) ejemplos :
  1 : tomada como ENTERA
  1F : tomada como FLOAT
  1L : tomada como LONG DOUBLE
- Una variable con parte decimal es tomada siempre como DOUBLE, salvo que se la siga de la letra F ó L
  1.0 : tomada como DOUBLE
  1.0F : tomada como FLOAT
  1.0L : tomada como LONG FLOAT
- Si en cualquiera de los casos anteriores agregamos la letra U ó u la constante queda calificada como UNSIGNED (consiguiendo mayor alcance) :
  1u : tomada como UNSIGNED INT
  1.0UL : tomada como UNSIGNED LONG DOUBLE
- Una variable numérica que comienza con "0" es tomado como OCTAL asi : 012 equivale a 10 unidades decimales
- Una variable numérica que comienza con "0x" ó "0X" es tomada como hexadecimal : 0x16 equivale a 22 unidades decimales y 0x1A a 26 unidades decimales.

10. CONSTANTES SIMBOLICAS

Por lo general es una mala práctica de programación colocar en un programa constantes en forma literal (sobre todo si se usan varias veces en el mismo) ya que el texto se hace dificil de comprender y aún más de corregir, si se debe cambiar el valor de dichas constantes.
Se puede en cambio asignar un símbolo a cada constante, y reemplazarla a lo largo del programa por el mismo, de forma que este sea más legible y además, en caso de querer modificar el valor, bastará con cambiarlo en la asignación.
El compilador, en el momento de crear el ejecutable, reemplazará el símbolo por el valor asignado.
Para dar un símbolo a una constante bastará, en cualquier lugar del programa (previo a su uso) poner la directiva: #define, por ejemplo:

```
#define VALOR_CONSTANTE 342

#define PI 3.1416
```

CAPITULO 3: OPERADORES

1. INTRODUCCION

Si analizamos la sentencia siguiente:
var1 = var2 + var3;
estamos diciéndole al programa, por medio del operador +, que compute la suma del valor de dos variables , y una vez realizado ésto asigne el resultado a otra variable var1. Esta última operación (asignación) se indica mediante otro operador, el signo =.
El lenguaje C tiene una amplia variedad de operadores, y todos ellos caen dentro de 6 categorias , a saber : aritméticos , relacionales, lógicos, incremento y decremento, manejo de bits y asignacion. Todos ellos se irán describiendo en los párrafos subsiguientes.

2. OPERADORES ARITMETICOS

Tal como era de esperarse los operadores aritméticos ,mostrados en la TABLA 4 , comprenden las cuatro operaciones basicas , suma , resta , multiplicación y división , con un agregado , el operador módulo .

TABLA 4 OPERADORES ARITMETICOS

SIMBOLO	DESCRIPCION	EJEMPLO	ORDEN DE EVALUACION
+	SUMA	a + b	3
-	RESTA	a - b	3
*	MULTIPLICACION	a * b	2
/	DIVISION	a / b	2
%	MODULO	a % b	2
-	SIGNO	-a	2

El operador módulo (%) se utiliza para calcular el resto del cociente entre dos ENTEROS , y NO puede ser aplicado a variables del tipo float ó double .
Si bien la precedencia (orden en el que son ejecutados los operadores) se analizará más adelante, en este capítulo, podemos adelantar algo sobre el orden que se realizan las operaciones aritméticas.
En la TABLA 4, última columna, se da el orden de evaluación de un operador dado. Cuanto más bajo sea dicho número mayor será su prioridad de ejecución. Si en una operación existen varios operadores, primero se evaluarán los de multiplicación , división y módulo y luego los de suma y resta . La precedencia de los tres primeros es la misma , por lo que si hay varios de ellos, se comenzará a evaluar a aquel que quede más a la izquierda . Lo mismo ocurre con la suma y la resta .
Para evitar errores en los cálculos se pueden usar paréntesis , sin limitación de anidamiento, los que fuerzan a realizar primero las operaciones incluidas en ellos . Los paréntesis no disminuyen la velocidad a la que se ejecuta el programa sino que tan sólo obligan al compilador a realizar las operaciones en un orden dado dado, por lo que es una buena costumbre utilizarlos ampliamente .
Los paréntesis tienen un orden de precedencia 0, es decir que antes que nada se evalúa lo que ellos encierran .
Se puede observar que no existen operadores de potenciación, radicación, logaritmación, etc, ya que en el lenguaje C todas estas operaciones (y muchas otras) se realizan por medio de llamadas a Funciones.
El último de los operadores aritméticos es el de SIGNO . No debe confundirselo con el de resta, ya que este es un operador unitario que opera sobre una única variable cambiando el signo de su contenido númerico. Obviamente no existe el operador +

unitario, ya que su operación sería DEJAR el signo de la variable, lo que se consigue simplemente por omisión del signo.

3. OPERADORES RELACIONALES

Todas las operaciones relacionales dan sólo dos posibles resultados : VERDADERO ó FALSO . En el lenguaje C, Falso queda representado por un valor entero nulo (cero) y Verdadero por cualquier número distinto de cero En la TABLA 5 se encuentra la descripción de los mismos .

TABLA 5 OPERADORES RELACIONALES

SIMBOLO	DESCRIPCION	EJEMPLO	ORDEN DE EVALUACION
<	menor que	(a < b)	5
>	mayor que	(a >b)	5
< =	menor o igual que	(a < = b)	5
>=	mayor o igual que	(a >>= b)	5
= =	igual que	(a = = b)	6
! =	distinto que	(a != b)	6

Uno de los errores más comunes es confundir el operador relacional IGUAL QUE (= =) con el de asignacion IGUAL A (=). La expresión a=b copia el valor de b en a, mientras que a = = b retorna un cero , si a es distinto de b ó un número distinto de cero si son iguales.

Los operadores relacionales tiene menor precedencia que los aritméticos , de forma que a < b + c se interpreta como a < (b + c), pero aunque sea superfluo recomendamos el uso de paréntesis a fin de aumentar la legilibilidad del texto.

Cuando se comparan dos variables tipo char el resultado de la operación dependerá de la comparación de los valores ASCII de los caracteres contenidos en ellas. Asi el caracter a (ASCII 97) será mayor que el A (ASCII 65) ó que el 9 (ASCII 57).

4. OPERADORES LOGICOS

Hay tres operadores que realizan las conectividades lógicas Y (AND) , O (OR) y NEGACION (NOT) y están descriptos en la TABLA 6 .

TABLA 6 OPERADORES LOGICOS

SIMBOLO	DESCRIPCION	EJEMPLO	ORDEN DE EVALUACION
&&	Y (AND)	(a>b) && (c < d)	10
||	O (OR)	(a>b) || (c < d)	11
!	NEGACION (NOT)	!(a>b)	1

Los resultados de la operaciones lógicas siempre adoptan los valores CIERTO ó FALSO. La evaluación de las operaciones lógicas se realiza de izquierda a derecha y se interrumpe cuando se ha asegurado el resultado .

El operador NEGACION invierte el sentido lógico de las operaciones , así será

```
 !( a >> b )    equivale a    ( a < b )
 !( a == b )      "     "    ( a != b )
 etc.
```

En algunas operaciones suele usárselo de una manera que se presta a confusión , por ejemplo : (!i) donde i es un entero. Esto dará un resultado CIERTO si i tiene un valor 0 y un resultado FALSO si i es distinto de cero .

5. OPERADORES DE INCREMENTO Y DECREMENTO

Los operadores de incremento y decremento son sólo dos y están descriptos en la TABLA 7

TABLA 7 OPERADORES DE INCREMENTO Y DECREMENTO

SIMBOLO	DESCRIPCION	EJEMPLO	ORDEN DE EVALUACION
++	incremento	++i ó i++	1
--	decremento	--i ó i--	1

Para visualizar rapidamente la función de los operadores antedichos , digamos que las sentencias :

a = a + 1 ;

a++ ;

tienen una acción idéntica , de la misma forma que

a = a - 1 ;

a-- ;

es decir incrementa y decrementa a la variable en una unidad

Si bien estos operadores se suelen emplear con variables int , pueden ser usados sin problemas con cualquier otro tipo de variable . Así si a es un float de valor 1.05 , luego de hacer a++ adoptará el valor de 2.05 y de la misma manera si b es una variable del tipo char que contiene el caracter 'C' , luego de hacer b-- su valor será 'B' .

Si bien las sentencias

i++ ;

++i ;

son absolutamente equivalentes, en la mayoria de los casos la ubicación de los operadores incremento ó decremento indica CUANDO se realiza éste .

Veamos el siguiente ejemplo :

int i = 1 , j , k ;

j = i++ ;

k = ++i ;

acá j es igualado al valor de i y POSTERIORMENTE a la asignación i es incrementado por lo que j será igual a 1 e i igual a 2 , luego de ejecutada la sentencia . En la siguiente instrucción i se incrementa ANTES de efectuarse la asignacion tomando el valor de 3 , él que luego es copiado en k .

6. OPERADORES DE ASIGNACION

En principio puede resultar algo futil gastar papel en describir al operador IGUAL A (=) , sin embargo es necesario remarcar ciertas características del mismo .

Anteriormente definimos a una asignación como la copia del resultado de una expresión (rvalue) sobre otra (lvalue) , esto implica que dicho lvalue debe tener LUGAR (es decir poseer una posición de memoria) para alojar dicho valor .

Es por lo tanto válido escribir

a = 17 ;

pero no es aceptado , en cambio

17 = a ;   /* incorrecto */

ya que la constante numérica 17 no posee una ubicación de memoria donde alojar al valor de a .

Aunque parezca un poco extraño al principio las asignaciones , al igual que las otras operaciones , dan un resultado que puede asignarse a su vez a otra expresión .

De la misma forma que (a + b) es evaluada y su resultado puedo copiarlo en otra variable : c = (a + b) ; una asignación (a = b) da como resultado el valor de b , por lo que es lícito escribir

c = (a = b) ;

Debido a que las asignaciones se evalúan de derecha a izquierda , los paréntesis son superfluos , y podrá escribirse entonces :

c = a = b = 17 ;

con lo que las tres variables resultarán iguales al valor de la contante .

El hecho de que estas operaciones se realicen de derecha a izquierda también permite realizar instrucciones del tipo :

a = a + 17 ;

significando esto que al valor que TENIA anteriormente a , se le suma la constante y LUEGO se copia el resultado en la variable .

Como este último tipo de operaciones es por demás común , existe en C un pseudocódigo , con el fín de abreviarlas .

Asi una operación arítmetica o de bit cualquiera (simbolizada por OP)

a = (a) OP (b) ;

puede escribirse en forma abreviada como :

a OP= b ;

Por ejemplo

```
a += b   ;   /* equivale :  a = a + b */
a -= b   ;   /* equivale :  a = a - b */
a *= b   ;   /* equivale :  a = a * b */
a /= b   ;   /* equivale :  a = a / b */
a %= b   ;   /* equivale :  a = a % b */
```

Nótese que el pseudooperador debe escribirse con los dos símbolos seguidos , por ejemplo += , y no será aceptado +(espacio) = .

Los operadores de asignación estan resumidos en la TABLA 8 .

TABLA 8 OPERADORES DE ASIGNACION

SIMBOLO	DESCRIPCION	EJEMPLO	ORDEN DE EVALUACION
=	igual a	a = b	13
op=	pseudocodigo	a += b	13
=?:	asig.condicional	a = (c>b)?d:e	12

Vemos de la tabla anterior que aparece otro operador denominado ASIGNACION CONDICIONAL . El significado del mismo es el siguiente :

lvalue = (operación relacional ó logica) ? (rvalue 1) : (rvalue 2) ;

de acuerdo al resultado de la operación condicional se asignará a lvalue el valor de rvalue 1 ó 2 . Si aquella es CIERTA será lvalue = rvalue 1 y si diera FALSO , lvalue = rvalue 2 .

Por ejemplo, si quisiéramos asignar a c el menor de los valores a ó b , bastará con escribir :

c = (a < b) ? a : b ;

7. OPERADORES DE MANEJO DE BITS

Estos operadores muestran una de las armas más potentes del lenguaje C , la de poder manipulear INTERNAMENTE , es decir bit a bit , las variables .

Debemos anticipar que estos operadores sólo se aplican a variables del tipo char , short , int y long y NO pueden ser usados con float ó double ,

Sabemos que las computadoras guardan los datos organizados en forma digital , en bytes , formado por números binarios de 8 bits y como se vió anteriormente cuando se analizó el tamaño de las variables , un char ocupará un byte de 8 bits , mientras que los

short e int se forman con dos bytes (16 bits) y los long por cuatro bytes (32 bits).
Para el manejo de dichos bits , contamos con los operadores descriptos en la TABLA 9 .

TABLA 9 OPERADORES DE MANEJO DE BITS

<table>
<tr><th>SIMBOLO</th><th>DESCRIPCION</th><th>EJEMPLO</th><th>ORDEN DE EVAL.</th></tr>
<tr><td>&</td><td>Y ó AND (bit a bit)</td><td>a & b</td><td>7</td></tr>
<tr><td>|</td><td>O ú OR INCLUSIVA</td><td>a | b</td><td>9</td></tr>
<tr><td>^</td><td>O ú OR EXCLUSIVA</td><td>a ^ b</td><td>8</td></tr>
<tr><td><<</td><td>ROTACION A LA IZQUIER</td><td>a << b</td><td>4</td></tr>
<tr><td>>></td><td>ROTACION A LA DERECHA</td><td>a >> b</td><td>4</td></tr>
<tr><td>~</td><td>COMPLEMENTO A UNO</td><td>~a</td><td>1</td></tr>
</table>

Describiremos mediante unos pocos ejemplos la operatoria de manejo de bits.
Analicemos primero como funciona el operador Y, también llamado BITWISE AND , las reglas para la operación son las dadas en la TABLA 10 .

TABLA 10 REGLAS PARA LA OPERACION Y (BITWISE AND)

bit a	&	bit b	=	bit c
0	&	0	=	0
0	&	1	=	0
1	&	0	=	0
1	&	1	=	1

Si suponemos tener dos variables del tipo char, una de ella de valor 85 (hex. 55), otra de valor 71 (hex. 47) y realizamos el AND a nivel bits de ellas, obtendremos :

```
        bits                  decimal hexadecimal
        0 1 0 1 0 1 0 1          85          55
        &                     &            &
        0 1 0 0 0 1 1 1          71          47
    -------------------------      -------   -------
        0 1 0 0 0 1 0 1          69          45
```

Nótese que la operación es del tipo lógico entre bits, por lo que los resultados numéricos tienen poco ó ningún significado y sólo se han puesto con el fin de ejemplificar .
De la misma manera para la operacion O INCLUSIVA, cuyas reglas se dan en la TABLA 11, será:

TABLA 11 REGLAS PARA LA OPERACION O INCLUSIVA (BITWISE OR)

<table>
<tr><th>bit a</th><th>|</th><th>bit b</th><th>=</th><th>bit c</th></tr>
<tr><td>0</td><td>|</td><td>0</td><td>=</td><td>0</td></tr>
<tr><td>0</td><td>|</td><td>1</td><td>=</td><td>1</td></tr>
<tr><td>1</td><td>|</td><td>0</td><td>=</td><td>1</td></tr>
<tr><td>1</td><td>|</td><td>1</td><td>=</td><td>1</td></tr>
</table>

Para las mismas variables anteriores obtendremos :
0 1 0 1 0 1 1 1 87 57
Analizando ahora la O EXCLUSIVA (ó EXOR) tendremos :

TABLA 12 REGLAS PARA LA OPERACION O EXCLUSIVA (EXOR)

bit a	^	bit b	=	bit c
0	^	0	=	0
0	^	1	=	1
1	^	0	=	1
1	^	1	=	0

Para las mismas variables anteriores obtendremos :

0 0 0 1 0 0 1 0   18   12

Veamos ahora las operaciones de desplazamiento , la sentencia

c = a << b

implica asignarle a c, el valor de a con sus bits corridos a la izquierda en b lugares , los bits que van "saliendo" por la izquierda , se desechan ; y los bits que van quedando libres a la derecha se completan con cero .

Se procede de la misma manera para el corrimiento a la derecha >>.

El operador COMPLEMENTO A UNO es del tipo unitario , es decir que realiza una operación sobre una única variable , y su efecto es dar a la variable un valor igual a restar de (-1) el valor que traía . Quizás es más visible decir que este operador cambia los bits en 1 de la variable en 0 y viceversa.

TABLA 13 PRECEDENCIA DE LOS OPERADORES

<table>
<tr><td>PRECEDENCIA</td><td>OPERADORES</td><td>ASOCIATIVIDAD</td></tr>
<tr><td>0</td><td>()[] -> .</td><td>izq. a derecha</td></tr>
<tr><td>1</td><td>sizeof (tipo) ! ~ ++ -- signo* &</td><td>derecha a izq.</td></tr>
<tr><td>2</td><td>* / %</td><td>izq. a derecha</td></tr>
<tr><td>3</td><td>+ -</td><td>izq. a derecha</td></tr>
<tr><td>4</td><td>></td><td>izq. a derecha</td></tr>
<tr><td>5</td><td>>=</td><td>izq. a derecha</td></tr>
<tr><td>6</td><td>== !=</td><td>izq. a derecha</td></tr>
<tr><td>7</td><td>&</td><td>izq. a derecha</td></tr>
<tr><td>8</td><td>^</td><td>izq. a derecha</td></tr>
<tr><td>9</td><td>|</td><td>izq. a derecha</td></tr>
<tr><td>10</td><td>&&</td><td>izq. a derecha</td></tr>
<tr><td>11</td><td>||</td><td>izq. a derecha</td></tr>
<tr><td>12</td><td>?:</td><td>derecha a izq.</td></tr>
<tr><td>13</td><td>= += -= *= etc</td><td>derecha a izq.</td></tr>
</table>

NOTA: en el renglón de los operadores de precedencia cero hemos agregado ubicándolos a la derecha del mismo para diferenciarlos, tres operadores , [] ->> y . que serán analizados más adelante, de la misma manera en el renglón siguiente hemos colocado al final a dos operadores: * y & ya que aunque coinciden en símbolo con los de PRODUCTO y AND A NIVEL BITS, son OTRO tipo de operadores que se describirán en capítulos sucesivos. En ese mismo renglón se ha consignado como SIGNO al unitario - .

CAPITULO 4: PROPOSICIONES PARA EL CONTROL DE FLUJO DE PROGRAMA

1. INTRODUCCION

En lo que sigue de este capítulo, denominaremos BLOQUE DE SENTENCIAS al conjunto de sentencias individuales incluídas dentro un par de llaves. Por ejemplo :

```
{
sentencia 1 ;
sentencia 2 ;
............
sentencia n ;
}
```

Este conjunto se comportará sintacticamente como una sentencia simple y la llave de cierre del bloque NO debe ir seguida de punto y coma .
Un ejemplo de bloque ya visto , es el cuerpo del programa principal de la función main() .

```
main()
{
bloque de sentencias
}
```

En las proposiciones de control de flujo de programa , trabajaremos alternativamente con sentencias simples y bloques de ellas .

2. PROPOSICION IF - ELSE

Esta proposición sirve para ejecutar ciertas sentencias de programa , si una expresión resulta CIERTA ú otro grupo de sentencias, si aquella resulta FALSA. Su interpretación literal sería : SI es CIERTA tal cosa , haga tal otra , si no lo es salteéla .
El caso más sencillo sería :

```
if(expresión)
 sentencia ;
```

ó

```
if(expresión)  sentencia ;
```

Cuando la sentencia que sigue al IF es única, las dos formas de escritura expresadas arriba son equivalentes . La sentencia sólo se ejecutará si el resultado de "expresión" es distinto de cero (CIERTO) , en caso contrario el programa salteará dicha sentencia , realizando la siguiente en su flujo.

Veamos unos ejemplos de las distintas formas que puede adoptar la "expresión" dentro de un IF :

if(a > b)	if((a > b) != 0)	las dos expresiones son idénticas, aunque a veces resulta más claro expresarla de la segunda manera, sobre todo en los primeros contactos con el lenguaje.
if(a)	if(a != 0)	
if(!a)	if(a == 0)	Las dos superiores son idénticas entre sí , al igual que las dos inferiores Obsérvese que (!a) dará un valor CIERTO sólo cuando a sea FALSO. (ver operador NEGACION en

		el capítulo anterior)
if(a == b)		
if(a = b)	if(a == b)	La primera es una expresión correcta , el IF se realizará sólo si a es igual a b. En cambio la segunda es un error , ya que no se está comparando a con b , sino ASIGNANDO el valor de esta a aquella . Sin embargo, a veces puede usarse como un truco (un poco sucio) de programacion , ya que primero se realiza la asignación y luego se evalúa el resultado de esta para realizar el IF , es entonces equivalente a escribir :
		a = b ;
if(a)	....	con el ahorro de una linea de programa (a costa de la legibilidad del mismo).

En casos más complejos que los anteriores , la proposición IF puede estar seguida por un bloque de sentencias :

```
if(expresión)               if(expresión) {

{                           sentencia 1 ;

sentencia 1 ;               sentencia 2 ;

sentencia 2 ;               ..............

............            }

}
```

Las dos maneras son equivalentes , por lo que la posición de la llave de apertura del bloque queda librada al gusto del programador . El indentado de las sentencias (sangría) es también optativo , pero sumamente recomendable ,sobre todo para permitir la lectura de proposiciones muy complejas ó anidadas , como se verá luego. El bloque se ejecutará en su conjunto si la expresion resulta CIERTA. El uso del ELSE es optativo , y su aplicación resulta en la ejecución de una , ó una serie de sentencias en el caso de que la expresión del IF resulta FALSA.

Su aplicación puede verse en el ejemplo siguiente :

```
  if(expresión)            if(expresión)

  {                    {

  sentencia 1 ;            sentencia 1 ;

  sentencia 2 ;            sentencia 2 ;

  }                    }
```

```
sentencia 3 ;           else
sentencia 4 ;           {
sentencia 5 ;           sentencia 3 ;
                        sentencia 4 ;
                        }
                        sentencia 5 ;
```

En el ejemplo de la izquierda no se usa el ELSE y por lo tanto las sentencias 3 , 4 y 5 se ejecutan siempre . En el segundo caso , las sentencias 1 y 2 se ejecutan solo si la expresión es CIERTA , en ese caso las 3 y 4 NO se ejecutarán para saltarse directamente a la 5 , en el caso de que la expresión resulte FALSA se realizarán las 3 y 4 en lugar de las dos primeras y finalmente la 5 .
La proposición ELSE queda siempre asociada al IF más cercano , arriba de él .
Es común también , en caso de decisiones múltiples , el uso de anidamientos ELSE-IF de la forma indicada abajo:

```
if(exp.1)               if(exp.1)
 sentencia1 ;            sentencia1 ;
else  if(exp.2)         else  if(exp.2)
 sentencia2 ;                 sentencia2 ;
else if(exp.3)                else if(exp.3)
 sentencia3 ;                      sentencia3 ;
else                          else
 sentencia5 ;                      sentencia5 ;
```

Si bién se suele escribir según la modalidad de la izquierda , a la derecha hemos expresado las asociaciones entre los distintos ELSE é IF por medio del indentado del texto.

3. PROPOSICION SWITCH

El SWITCH es una forma sencilla de evitar largos , tediosos y confusos anidamientos de ELSE-IF .
Supongamos que estamos implementando un Menu , con varias elecciones posibles . El esqueleto de una posible solución al problema usando if-else podría ser el siguiente :

```
#include <<stdio.h>>

main()

{
```

```
int c  ;
printf("\nMENU :") ;
printf("\n       A = ADICIONAR A LA LISTA ") ;
printf("\n       B = BORRAR DE LA LISTA   ") ;
printf("\n       O = ORDENAR LA LISTA     ") ;
printf("\n       I = IMPRIMIR LA LISTA    ") ;
printf("\n\nESCRIBA SU SELECCION , Y LUEGO <<ENTER>> : ") ;
if( (c = getchar()) != '\n' )
 {
  if( c == 'A')
    printf("\nUD. SELECCIONO AGREGAR") ;
  else
    if( c == 'B')
     printf("\nUD. SELECCIONO BORRAR") ;
    else
     if( c == 'O' )
      printf("\nUD. SELECCIONO ORDENAR") ;
     else
      if( c == 'I' )
       printf("\nUD. SELECCIONO IMPRIMIR") ;
      else
       printf("\n\a\aUD. APRETO UN CARACTER ILEGAL" ) ;
 }
 else
  printf("\n¡ UD. NO HA SELECCIONADO NADA !" ) ;
```

```
}
```

Como es fácil de ver , cuando las opciones son muchas, el texto comienza a hacerse difícil de entender y engorroso de escribir.
El mismo programa, utilizando un SWITCH , quedaría mucho más claro de leer, y sencillo de escribir, como se aprecia en el EJEMPLO siguiente.

```
#include <stdio.h>

#include <conio.h>

main()
{

int c  ;

printf("\nMENU :") ;

printf("\n       A = ADICIONAR A LA LISTA ") ;

printf("\n       B = BORRAR DE LA LISTA   ") ;

printf("\n       O = ORDENAR LA LISTA     ") ;

printf("\n       I = IMPRIMIR LA LISTA    ") ;

printf("\n\nESCRIBA SU SELECCION , Y LUEGO <<ENTER>> : ") ;

c = getchar() ;

switch (c)

 {

  case 'A' :

    printf("\nUD. SELECCIONO AGREGAR") ;

    break ;

  case 'B' :

    printf("\nUD. SELECCIONO BORRAR") ;

    break ;
```

```
 case 'O' :
   printf("\nUD. SELECCIONO ORDENAR") ;
   break ;
 case 'I' :
   printf("\nUD. SELECCIONO IMPRIMIR") ;
   break ;
 case '\n':
   printf("\n¡ UD. NO HA SELECCIONADO NADA !" ) ;
   break ;
 default  :
   printf("\n\a\aUD. APRETO UN CARACTER ILEGAL" ) ;
   break ;
}
}
```

El SWITCH empieza con la sentencia : switch (expresión) . La expresión contenida por los paréntesis debe ser ENTERA , en nuestro caso un caracter ; luego mediante una llave abre el bloque de las sentencias de comparación . Cada una de ellas se representa por la palabra clave "case" seguida por el valor de comparación y terminada por dos puntos . Seguidamente se ubican las sentencias que se quieren ejecutar , en el caso que la comparación resulte CIERTA . En el caso de resultar FALSA , se realizará la siguiente comparación , y así sucesivamente .
Prestemos atención tambien a la sentencia BREAK con la que se termina cada CASE. Una característica poco obvia del SWITCH , es que si se eliminan los BREAK del programa anterior , al resultar CIERTA una sentencia de comparación, se ejecutarán las sentencias de ese CASE particular pero TAMBIEN la de todos los CASE por debajo del que ha resultado verdadero. Quizás se aclare esto diciendo que , las sentencias propias de un CASE se ejecutarán si su comparación ú otra comparación ANTERIOR resulta CIERTA . La razón para este poco "juicioso" comportamiento del SWITCH es que así se permite que varias comparaciones compartan las mismas sentencias de programa , por ejemplo :

```
.................
case 'X' :
case 'Y' :
```

```
case 'Z' :

  printf(" UD. ESCRIBIO X , Y , ó  Z ") ;

  break ;

.................
```

La forma de interrumpir la ejecución luego de haber encontrado un CASE cierto es por medio del BREAK , el que dá por terminado el SWITCH .
Al final del bloque de sentencias del SWITCH , aparece una optativa llamada DEFAULT , que implica : si no se ha cumplido ningun CASE , ejecute lo que sigue. Es algo superfluo poner el BREAK en este caso , ya que no hay más sentencias despues del DEFAULT , sin embargo , como el orden en que aparecen las comparaciones no tiene importancia para la ejecución de la instrucción, puede suceder que en futuras correcciones del programa se agregue algún nuevo CASE luego del DEFAULT , por lo que es conveniente preveerlo , agregando el BREAK , para evitar errores de laboriosa ubicación .
Más adelante volveremos sobre otros usos del BREAK.

4. LA ITERACION WHILE

El WHILE es una de las tres iteraciones posibles en C . Su sintaxis podría expresarse de la siguiente forma :

```
while(expresion)          ó          while(expresión) {

  proposición 1 ;                   proposición 1 ;

                                    proposición 2 ;

                                    ...............

                                    proposición n ;

                                    }
```

Esta sintaxis expresada en palabras significaria: mientras (expresión) dé un resultado CIERTO ejecútese la proposición 1 , en el caso de la izquierda ó ejecútese el bloque de sentencias , en el caso de la derecha.
Por lo general , dentro de la proposición ó del bloque de ellas , se modifican términos de la expresión condicional , para controlar la duración de la iteración .

5. LA ITERACION DO - WHILE

Su sintaxis será :

```
do {

 proposición 1 ;

 proposición 2 ;

 ...............

} while (expresión) ;
```

Expresado en palabras , esto significa : ejecute las proposiciones , luego repita la ejecución mientras la expresión dé un resultado CIERTO . La diferencia fundamental entre esta iteración y la anterior es que el DO-WHILE se ejecuta siempre AL MENOS una vez , sea cual sea el resultado de expresión.

6. ITERACION FOR

El FOR es simplemente una manera abreviada de expresar un WHILE , veamos su sintaxis :

```
for ( expresión1 ; expresión2 ; expresion3 ) {

        proposición1 ;

        proposición2 ;

        .............

}
```

Esto es equivalente a :

```
expresión1 ;

while ( expresión2 ) {

        proposición1 ;

        proposición2 ;

        .............

        expresion3   ;

}
```

La expresión1 es una asignación de una ó más variables , (equivale a una inicialización de las mismas) , la expresión2 es una relación de algun tipo que , mientras dé un valor CIERTO , permite la iteración de la ejecución y expresión3 es otra asignación , que comunmente varía alguna de las variables contenida en expresión2 .

Todas estas expresiones , contenidas en el paréntesis del FOR deben estar separadas por PUNTO Y COMA y NO por comas simples .

No es imprescindible que existan TODAS las expresiones dentro del paréntesis del FOR , pudiendose dejar en blanco algunas de ellas , por ejemplo :

for (; exp2 ; exp3)   ó

for (exp1 ; ;)   ó

for (; ;)

Estas dos últimas expresiónes son interesantes desde el punto de vista de su falta de término relacional , lo que implica que el programador deberá haber previsto alguna manera alternativa de salir del lazo (probablemente mediante BREAK ó RETURN como veremos más adelante) ya que sinó , la ejecución del mismo es infinita (ó tan larga como se mantenga encendida la computadora) .

7. LA SENTENCIA BREAK

El BREAK , ya brevemente descripto con el SWITCH , sirve también para terminar loops producidos por WHILE , DO-WHILE y FOR antes que se cumpla la condición normal de terminación . En el EJEMPLO siguiente vemos su uso para terminar un WHILE indeterminado.

```
#include <stdio.h>

#include <conio.h>

main()

{

  char c ;

  printf("ESTE ES UN LOOP INDEFINIDO ") ;

  while(1)  {

   printf( "DENTRO DEL LOOP INDEFINIDO (apriete una tecla):" ) ;

   if( (c = getch()) == 'Q' )

    break ;

   printf( "\nNO FUE LA TECLA CORRECTA PARA ABANDONAR EL LOOP ") ;

  }

   printf("\nTECLA CORRECTA : FIN DEL WHILE ") ;

}
```

Obsérvese que la expresión while(1) SIEMPRE es cierta , por lo que el programa correrá imparable hasta que el operador oprima la tecla "secreta" Q . Esto se consigue en el IF , ya que cuando c es igual al ASCII Q se ejecuta la instrucción BREAK ,dando por finalizado el WHILE .

El mismo criterio podría aplicarse con el DO-WHILE ó con FOR , por ejemplo haciendo

```
 for (;;)  {     /* loop indefinido */

 ............

  if( expresión )
```

```
 break ;     /* ruptura del loop cuando expresión sea verdadera */

}
```

8. LA SENTENCIA CONTINUE

La sentencia CONTINUE es similar al BREAK con la diferencia que en vez de terminar violentamente un loop , termina con la realización de una iteración particular y permitiendo al programa continuar con la siguiente.

9. LA FUNCION EXIT()

La función EXIT() tiene una operatoria mucho más drastica que las anteriores , en vez de saltear una iteración ó abandonar un lazo de programa , esta abandona directamente al programa mismo dándolo por terminado . Realiza también una serie de operaciones útiles como ser , el cerrado de cualquier archivo que el programa hubiera abierto , el vaciado de los buffers de salida , etc.

Normalmente se la utiliza para abortar los programas en caso de que se esté por cometer un error fatal é inevitable . Mediante el valor que se le ponga en su argumento se le puede informar a quien haya llamado al programa (Sistema Operativo , archivo .bat , u otro programa) el tipo de error que se cometió.

10 SENTENCIA GOTO

Si Ud. se ha admirado de que C tenga la operación GOTO , recuerde que el hecho de existir NO lo obliga a usarla , en el mismo sentido que por tener puertas los aviones no está obligado a saltar por ellas en pleno vuelo.

El uso del GOTO implica un salto incondicional de un lugar a otro del programa . Esta práctica hace que los programas sean muy dificiles de corregir ó mantener.

Si no quedara más remedio que usarlo, (y en programación estructurada SIEMPRE hay remedio) debe marcarse el destino del salto mediante un nombre seguido por dos puntos .

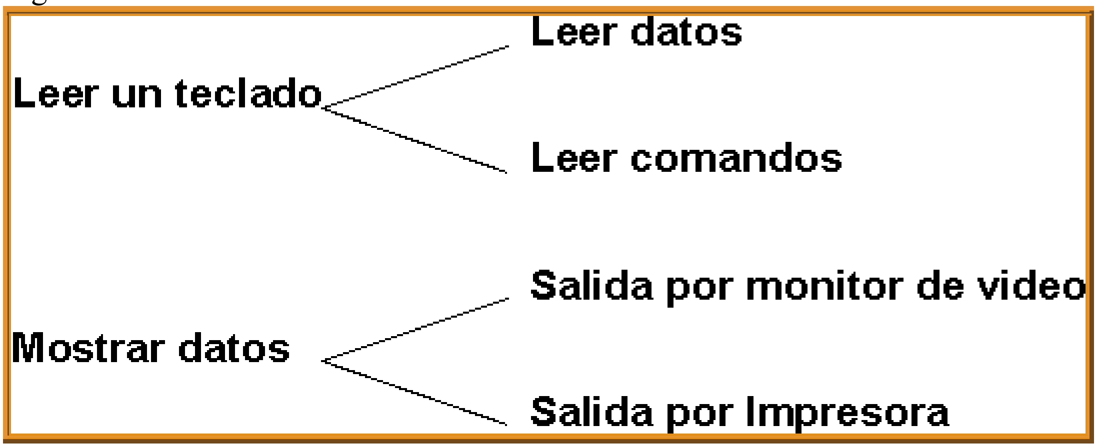
```
if( c == 0 )  goto OTRO_LADO ;
.............................
OTRO_LADO:
 printf(........
```

En este caso si c es cero se saltean todas las sentencias entre el if y el destino , continuandose con la ejecución del printf() . El destino puede ser tanto posterior como anterior al GOTO invocante .

CAPITULO 5: FUNCIONES

1.INTRODUCCION

La forma más razonable de encarar el desarrollo de un programa complicado es aplicar lo que se ha dado en llamar "Programación Top - Down" .
Esto implica que, luego de conocer cual es la meta a alcanzar, se subdivide esta en otras varias tareas concurrentes, por ejemplo :
Leer un teclado, procesar datos, mostrar los resultados .
Luego a estas se las vuelve a dividir en otras menores :



Y así se continúa hasta llegar a tener un gran conjunto de pequeñas y simples tareas, del tipo de "leer una tecla" ó "imprimir un caracter".
Luego sólo resta abocarse a resolver cada una de ellas por separado.
De esta forma el programador, sólo se las tendrá que ver con diminutas piezas de programa, de pocas lineas, cuya escritura y corrección posterior es una tarea simple.
Tal es el criterio con que está estructurado el lenguaje C, donde una de sus herramientas fundamentales són las funciones. Todo compilador comercial trae una gran cantidad de Librerias de toda índole, matematicas, de entrada - salida, de manejo de textos, de manejo de gráficos, etc, que solucionan la mayor parte de los problemas básicos de programación .
Sin embargo será inevitable que en algún momento tenga que crear mis propias funciones, las reglas para ello son las que desarrollaremos en este capítulo .
Comencemos con algunos conceptos básicos: para hacer que las instrucciones contenidas en una función, se ejecuten en determinado momento, no es necesario más que escribir su nombre como una linea de sentencia en mi programa.
Convencionalmente en C los nombres de las funciones se escriben en minúscula y siguen las reglas dadas anteriormente para los de las variables, pero deben ser seguidos, para diferenciarlas de aquellas por un par de paréntesis .
Dentro de estos paréntesis estarán ubicados los datos que se les pasan a las funciones. Está permitido pasarles uno, ninguno ó una lista de ellos separados por comas, por ejemplo: pow10(a), getch(), strcmp(s1, s2) .
Un concepto sumamente importante es que los argumentos que se les envían a las funciones son los VALORES de las variables y NO las variables mismas. En otras palabras, cuando se invoca una función de la forma pow10(a) en realidad se está copiando en el "stack" de la memoria el valor que tiene en ese momento la variable a, la función podrá usar este valor para sus cálculos, pero está garantizado que los mismos no

afectan en absoluto a la variable en sí misma.
Como veremos más adelante, es posible que una función modifique a una variable, pero para ello, será necesario comunicarle la DIRECCION EN MEMORIA de dicha variable .
Las funciones pueden ó no devolver valores al programa invocante. Hay funciones que tan sólo realizan acciones, como por ejemplo clrscr(), que borra la pantalla de video, y por lo tanto no retornan ningun dato de interés; en cambio otras efectuan cálculos, devolviendo los resultados de los mismos.
La invocación a estos dos tipos de funciones difiere algo, por ejemplo escribiremos :

```
clrscr() ;

c = getch() ;
```

donde en el segundo caso el valor retornado por la función se asigna a la variable c. Obviamente ésta deberá tener el tipo correcto para alojarla .

2. DECLARACION DE FUNCIONES

Antes de escribir una función es necesario informarle al Compilador los tamaños de los valores que se le enviarán en el stack y el tamaño de los valores que ella retornará al programa invocante .
Estas informaciones están contenidas en la DECLARACION del PROTOTIPO DE LA FUNCION.
Formalmente dicha declaración queda dada por :

  tipo del valor de retorno   nombre_de_la_función(lista de tipos de parámetros)

Pongamos algunos ejemplos :

```
float mi_funcion(int i, double j ) ;

double otra_funcion(void) ;

       otra_mas(long p) ;

void   la_ultima(long double z, char y, int x, unsigned long w) ;
```

El primer término del prototipo da, como hemos visto el tipo del dato retornado por la función; en caso de obviarse el mismo se toma, por omisión, el tipo int. Sin embargo, aunque la función devuelva este tipo de dato, para evitar malas interpretaciones es conveniente explicitarlo .
Ya que el "default" del tipo de retorno es el int, debemos indicar cuando la función NO retorna nada, esto se realiza por medio de la palabra VOID (sin valor).
De la misma manera se actúa, cuando no se le enviarán argumentos.
Más adelante se profundizará sobre el tema de los argumentos y sus características.
La declaración debe anteceder en el programa a la definición de la función. Es normal, por razones de legibilidad de la documentación, encontrar todas las declaraciones de las funciones usadas en el programa, en el HEADER del mismo, junto con los include de los archivos *.h que tienen los prototipos de las funciones de Librería.
Si una ó más de nuestras funciones es usada habitualmente, podemos disponer su prototipo en un archivo de texto, e incluirlo las veces que necesitemos, según se vio en capítulos previos.

3. DEFINICION DE LAS FUNCIONES

La definición de una función puede ubicarse en cualquier lugar del programa, con sólo dos restricciones: debe hallarse luego de dar su prototipo, y no puede estar dentro de la

definición de otra función (incluida main()). Es decir que a diferencia de Pascal, en C las definiciones no pueden anidarse.
NOTA: no confundir definición con llamada; una función puede llamar a tantas otras como desee .
La definición debe comenzar con un encabezamiento, que debe coincidir totalmente con el prototipo declarado para la misma, y a continuación del mismo, encerradas por llaves se escribirán las sentencias que la componen; por ejemplo:

```
#include <stdio.h>

float mi_funcion(int i, double j ); /* DECLARACION observe que termina en  ";"  */

main()

{

float k ;

int p ;

double z ;

...........
k = mi_funcion( p, z );            /* LLAMADA a la función */

...........

}                              /* fin de la función main() */
float mi_funcion(int i, double j )   /* DEFINICION  observe que NO lleva  ";"  */

{

float n

..................

printf("%d", i );                /* LLAMADA  a otra función */

..................

return ( 2 * n );                /* RETORNO devolviendo un valor float */

}
```

Pasemos ahora a describir más puntualmente las distintas modalidades que adoptan las funciones .

4. FUNCIONES QUE NO RETORNAN VALOR NI RECIBEN PARAMETROS

Veamos como ejemplo la implementacion de una funcion "pausa"

```
#include <stdio.h>

void pausa(void) ;
```

```
main()

{

  int contador = 1;
  printf("VALOR DEL CONTADOR DENTRO DEL while \n");
  while (contador <= 10) {

    if(contador == 5 ) pausa();

    printf("%d\n", contador++);

  }

  pausa() ;

  printf("VALOR DEL CONTADOR LUEGO DE SALIR DEL while: %d", contador) ;

  return 0;

}

void pausa(void)

{

char c ;
printf("\nAPRIETE ENTER PARA CONTINUAR ") ;

while( (c = getchar()) != '\n') ;

}
```

Analicemos lo hecho, en la segunda linea hemos declarado la función pausa, sin valor de retorno ni parámetros.
Luego esta es llamada dos veces por el programa principal, una cuando contador adquiere el valor de 5 (antes de imprimirlo) y otra luego de finalizar el loop.
Posteriormente la función es definida. El bloque de sentencias de la misma está compuesto, en este caso particular, por la definición de una variable c, la impresión de un mensaje de aviso y finalmente un while que no hace nada, solo espera recibir un caracter igual a <ENTER>.
En cada llamada, el programa principal transfiere el comando a la función, ejecutandose, hasta que ésta finalice, su propia secuencia de instrucciones. Al finalizar la función esta retorna el comando al programa principal, continuandose la ejecución por la instrucción que sucede al llamado .
Si bien las funciones aceptan cualquier nombre, es una buena técnica de programación nombrarlas con términos que representen, aunque sea vagamente, su operatoria .
Se puede salir prematuramente de una función void mediante el uso de RETURN, sin que este sea seguido de ningun parámetro ó valor .

5. FUNCIONES QUE RETORNAN VALOR
Analicemos por medio de un ejemplo dichas funciones :

```
#include <stdio.h>

#include <conio.h>
#define FALSO 0

#define CIERTO 1
int finalizar(void);

int lea_char(void) ;
main()

{

  int i = 0;

  int fin = FALSO;

  printf("Ejemplo de Funciones que retornan valor\n");

  while (fin == FALSO) {

    i++;

    printf("i == %d\n", i);

    fin = finalizar();

  }

  printf("\n\nFIN DEL PROGRAMA........");

  return 0;

}

int finalizar(void)

{

  int c;

  printf("Otro número ? (s/n) ");

  do {

    c = lea_char() ;
```

```
  } while ((c != 'n') && (c != 's'));

  return (c == 'n');

}
int lea_char(void)

{

int j ;
if( (j = getch()) >>= 'A' && j <<= 'Z' )

 return( j + ( 'a' - 'A') ) ;

else

 return j ;

}
```

Analicemos paso a paso el programa anterior; las dos primeras lineas incluiran, en el programa los prototipos de las funciones de librería usadas, (en este caso printf() y getch()). En las dos siguientes damos nombres simbólicos a dos constantes que usaremos en las condiciones lógicas y posteriormente damos los prototipos de dos funciones que hemos creado.
Podrían haberse obviado, en este caso particular, estas dos últimas declaraciones, ya que ambas retornan un int (default), sin embargo el hecho de incluirlas hará que el programa sea más facilmente comprensible en el futuro.
Comienza luego la función main(), inicializando dos variables, i y fin, donde la primera nos servirá de contador y la segunda de indicador lógico. Luego de imprimir el rótulo del programa, entramos en un loop en el que permaneceremos todo el tiempo en que fin sea FALSO.
Dentro de este loop, incrementamos el contador, lo imprimimos, y asignamos a fin un valor que es el retorno de la función finalizar() .
Esta asignación realiza la llamada a la función, la que toma el control del flujo del programa, ejecutando sus propias instrucciones.
Saltemos entonces a analizar a finalizar(). Esta define su variable propia, c, (de cuyas propiedades nos ocuparemos más adelante) y luego entra en un do-while, que efectúa una llamada a otra función, lea_char(), y asigna su retorno a c iterando esta operativa si c no es 'n' ó 's', note que: c != 'n' && c != 's' es equivalente a: !(c == 'n' || c == 's') .
La función lea_char() tiene como misión leer un caracter enviado por el teclado, (lo realiza dentro de la expresión relacional del IF) y salvar la ambigüedad del uso de mayúsculas ó minúsculas en las respuestas, convirtiendo las primeras en las segundas. Es facil de ver que, si un caracter esta comprendido entre A y Z, se le suma la diferencia entre los ASCII de las minúsculas y las mayúsculas (97 - 65 = 32) para convertirlo, y luego retornarlo al invocante.
Esta conversión fué incluida a modo de ejemplo solamente, ya que existe una de Librería, tolower() declarada en ctype.h, que realiza la misma tarea.

Cuando lea_char() devuelva un caracter n ó s, se saldrá del do-while en la función finalizar() y se retornará al programa principal, el valor de la comparación lógica entre el contenido de c y el ASCII del caracter n. Si ambos son iguales, el valor retornado será 1 (CIERTO) y en caso contrario 0 (FALSO) .
Mientras el valor retornado al programa principal sea FALSO, este permanecerá dentro de su while imprimiendo valores sucesivos del contador, y llamadas a las funciones, hasta que finalmente un retorno de CIERTO (el operador presionó la tecla n) hace terminar el loop e imprimir el mensaje de despedida.
Nota: preste atención a que en la función finalizar() se ha usado un do-while .¿Cómo modificaría el programa para usar un while ?. En la función lea_char se han usado dos returns, de tal forma que ella sale por uno u otro. De esta manera si luego de finalizado el else se hubiera agregado otra sentencia, esta jamás sería ejecutada.
En el siguiente ejemplo veremos funciones que retornan datos de tipo distinto al int.
Debemos presentar antes, otra función muy común de entrada de datos: scanf(), que nos permitirá leer datos completos (no solo caracteres) enviados desde el teclado, su expresión formal es algo similar a la del printf() ,
scanf("secuencia de control", dirección de la variable) ;
Donde en la secuencia de control se indicará que tipo de variable se espera leer, por ejemplo :

%d si se desea leer un entero decimal (int)

%o " " " " " " octal "

%x " " " " " " hexadecimal "

%c " " " " " caracter

%f leerá un flot

%ld leerá un long int

%lf leerá un double

%Lf leerá un long double

Por "dirección de la variable" deberá entenderse que se debe indicar, en vez del nombre de la variable en la que se cargará el valor leido, la dirección de su ubicación en la memoria de la máquina. Esto suena sumamente apabullante, pero por ahora solo diremos, (más adelante abundaremos en detalles) que para ello es necesario simplemente anteponer el signo & al nombre de la misma .
Así, si deseo leer un entero y guardarlo en la variable "valor_leido" escribiré: scanf("%d",&valor_leido); en cambio si deseara leer un entero y un valor de punto flotante será: scanf("%d %f", &valor_entero, &valor_punto_flotante) ;
El tipo de las variables deberá coincidir EXACTAMENTE con los expresados en la secuencia de control, ya que de no ser así, los resultados son impredecibles.
Por supuesto esta función tiene muchísimas más opciones, (consulte el Manual de Librerias de su Compilador, si tiene curiosidad) sin embargo, por simplicidad, por ahora nos conformaremos con las antedichas.
El prototipo de scanf() esta declarado en stdio.h .

Usaremos también otra función, ya citada, clrscr(). Recordemos que esta es solo válida para máquinas tipo PC compatibles y no corre bajo Windows.
Encaremos ahora un programa que nos presente primero, un menú para seleccionar la conversión de ºC a Fahrenheit ó de centímetros a pulgadas, hecha nuestra elección, nos pregunte el valor a convertir y posteriormente nos de el resultado .
Si suponemos que las funciones que usaremos en el programa serán frecuentemente usadas, podemos poner las declaraciones de las mismas, así como todas las contantes que usen, en un archivo texto, por ejemplo convers.h. Este podrá ser guardado en el subdirectorio donde están todos los demás (INCLUDE) ó dejado en el directorio activo, en el cual se compila el programa fuente de nuestro problema. Para variar, supongamos que esto último es nuestro caso .

CONVERS.H

```
#include <conio.h>

#define FALSO  0

#define CIERTO 1

#define CENT_POR_INCH  25.4
void pausa(void)                ;

void mostrar_menu(void)         ;

int seleccion(void)             ;

void cm_a_pulgadas(void)        ;

void grados_a_fahrenheit(void)  ;

double leer_valor(void)         ;
```

Vemos que un Header puede incluir llamadas a otros (en este caso conio.h). Hemos puesto tambien la definición de todas las constantes que usaran las funciones abajo declaradas. De dichas declaraciones vemos que usaremos funciones que no retornan nada, otra que retorna un entero y otra que devuelve un double .
Veamos ahora el desarrollo del programa en sí. Observe que la invocación a conversión.h se hace con comillas, por haber decidido dejarlo en el directorio activo .

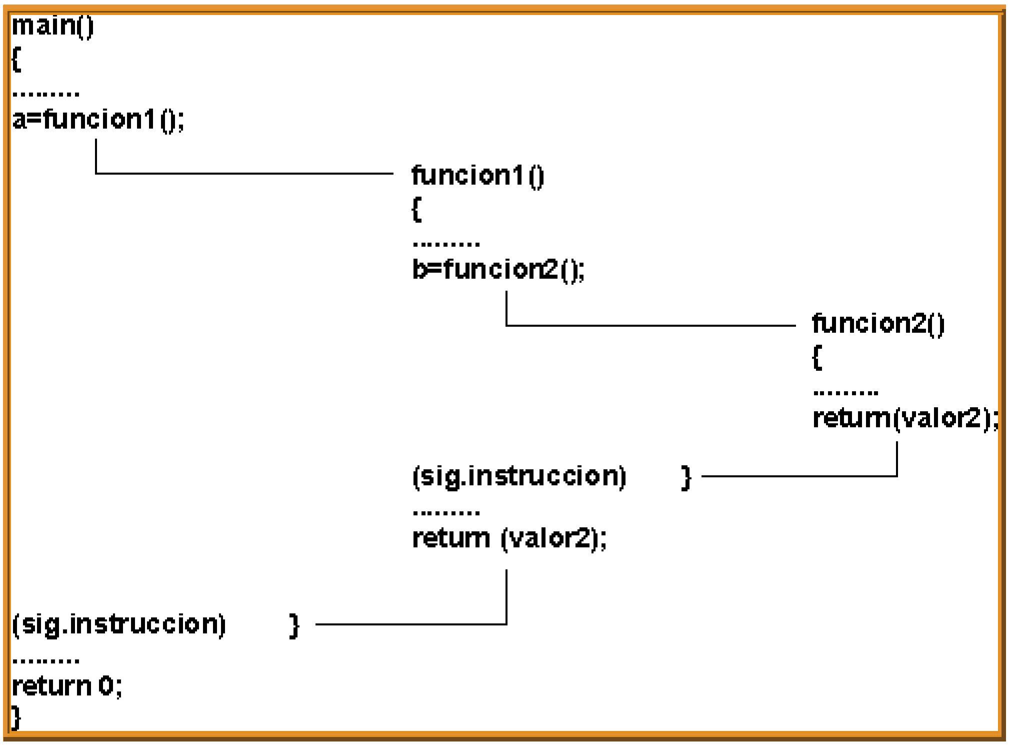
```
#include <stdio.h>

#include "convers.h"
main()

{

  int fin = FALSO;
  while (!fin) {
```

```
    mostrar_menu();

    switch(seleccion()) {

      case 1:

        cm_a_pulgadas();

        break;

      case 2:

        grados_a_fahrenheit();

        break;

      case 3:

        fin = CIERTO;

        break;

      default:

        printf("\n¡Error en la Seleccion!\a\a\n");

        pausa() ;

    }

  }

  return 0;

}
/* Funciones */

void pausa(void)

{

char c = 0;
printf("\n\n\nAPRIETE ENTER PARA CONTINUAR ") ;

while( (c = getch()) != '\r') ;

}
void mostrar_menu(void)
```

```
{

  clrscr();

  printf("\n         Menu\n");

  printf("--------------------------\n");

  printf("1:  Centimetros a pulgadas\n");

  printf("2:  Celsius a Fahrenheit\n");

  printf("3:  Terminar\n");

}
int seleccion(void)

{

  printf("\nEscriba el número de su Selección: ");

  return (getche() - '0');

}

void cm_a_pulgadas(void)

{

  double centimetros;  /* Guardará el valor pasado por leer_valor() */

  double pulgadas    ; /* Guardará el valor calculado */
  printf("\nEscriba los Centimetros a convertir: ");

  centimetros = leer_valor();

  pulgadas = centimetros  * CENT_POR_INCH;

  printf("%.3f Centimetros = %.3f Pulgadas\n", centimetros, pulgadas);

  pausa()   ;

}
void grados_a_fahrenheit(void)

{

  double grados;  /* Guardará el valor pasado por leer_valor() */

  double fahrenheit  ; /* Guardará el valor calculado */
```

```
  printf("\nEscriba los Grados a convertir: ");

  grados = leer_valor();

  fahrenheit = (((grados * 9.0)/5.0) + 32.0) ;

  printf("%.3f Grados = %.3f Fahrenheit", grados, fahrenheit);

  pausa();

}
double leer_valor(void)

{

  double valor;  /* Variable para guardar lo leido del teclado */
  scanf("%lf", &valor);

  return valor;

}
```

Veamos que hemos hecho: primero incluimos todas las definiciones presentes en el archivo convers.h que habiamos previamente creado. Luego main() entra en un loop, que finalizará cuando la variable fin tome un valor CIERTO, y dentro del cual lo primero que se hace es llamar a mostrar_menú(), que pone los rótulos de opciones . Luego se entra en un SWITCH que tiene como variable ,el retorno de la función selección() (recuerde que tiene que ser un entero), según sea éste se saldrá por alguno de los tres CASE. Observe que selección() lee el teclado mediante un getche(), (similar a getch() antes descripta, pero con la diferencia que aquella hace eco del caracter en la pantalla) y finalmente devuelve la diferencia entre el ASCII del número escrito menos el ASCII del número cero, es decir, un entero igual numericamente al valor que el operador quizo introducir .
Si este fue 1, el SWITCH invoca a la función cm_a_pulgadas() y en caso de ser 2 a grados_a_fahrenheit() .
Estas dos últimas proceden de igual manera: indican que se escriba el dato y pasan el control a leer_valor(), la que mediante scanf() lo hace, retornando en la variable valor, un double, que luego es procesado por aquellas convenientemente. Como hasta ahora la variable fin del programa principal no ha sido tocada, y por lo tanto continua con FALSO ,la iteración del while sigue realizandose, luego que se ejecuta el BREAK de finalización del CASE en cuestión. En cambio, si la selección() hubiera dado un resultado de tres, el tercer case, la convierte en CIERTO, con lo que se finaliza el WHILE y el programa termina.
Vemos en este ejemplo, la posibilidad de múltiples llamados a funciones, una llama a otra, que a su vez llama a otra, la cual llama a otra, etc ,etc, dando un esquema de flujo de programa de la forma :



6. AMBITO DE LAS VARIABLES (SCOPE)

VARIABLES GLOBALES

Hasta ahora hemos diferenciado a las variable segun su "tipo" (int, char double, etc), el cual se refería, en última instancia, a la cantidad de bytes que la conformaban. Veremos ahora que hay otra diferenciación de las mismas, de acuerdo a la clase de memoria en la que residen .

Si definimos una variable AFUERA de cualquier función (incluyendo esto a main()), estaremos frente a lo denominado VARIABLE GLOBAL. Este tipo de variable será ubicada en el segmento de datos de la memoria utilizada por el programa, y existirá todo el tiempo que esté ejecutandose este .

Este tipo de variables son automaticamente inicializadas a CERO cuando el programa comienza a ejecutarse .

Son accesibles a todas las funciones que esten declaradas en el mismo, por lo que cualquiera de ellas podrá actuar sobre el valor de las mismas. Por ejemplo :

```
#include <stdio.h>

double una_funcion(void);

double variable_global ;

main()

{

double i ;

printf("%f", variable_global );      /* se imprimirá 0 */
```

```
i = una_funcion() ;

printf("%f", i );                    /* se imprimirá 1 */

printf("%f", variable_global );       /* se imprimirá 1 */

variable_global += 1 ;

printf("%f", variable_global );       /* se imprimirá 2 */

return 0 ;

}
double una_funcion(void)

{

return( variable_global  += 1) ;

}
```

Observemos que la variable_global está definida afuera de las funciones del programa, incluyendo al main(), por lo que le pertenece a TODAS ellas. En el primer printf() del programa principal se la imprime, demostrandose que está automaticamente inicializada a cero .

Luego es incrementada por una_funcion() que devuelve ademas una copia de su valor, el cual es asignado a i ,la que, si es impresa mostrará un valor de uno, pero tambien la variable_global ha quedado modificada, como lo demuestra la ejecución de la sentencia siguiente. Luego main() tambien modifica su valor , lo cual es demostrado por el printf() siguiente.

Esto nos permite deducir que dicha variable es de uso público, sin que haga falta que ninguna función la declare, para actuar sobre ella.

Las globales son a los demás tipos de variables, lo que el GOTO es a los otros tipos de sentencias .

Puede resultar muy difícil evaluar su estado en programas algo complejos, con múltiples llamados condicionales a funciones que las afectan, dando comunmente orígen a errores muy engorrosos de corregir .

VARIABLES LOCALES

A diferencia de las anteriores, las variables definidas DENTRO de una función, son denominadas VARIABLES LOCALES a la misma, a veces se las denomina también como AUTOMATICAS, ya que son creadas y destruídas automaticamente por la llamada y el retorno de una función, respectivamente .

Estas variables se ubican en la pila dinámica (stack) de memoria ,destinandosele un espacio en la misma cuando se las define dentro de una función, y borrándose cuando la misma devuelve el control del programa, a quien la haya invocado.

Este método permite que, aunque se haya definido un gran número de variables en un programa, estas no ocupen memoria simultaneamente en el tiempo, y solo vayan incrementando el stack cuando se las necesita, para luego, una vez usadas desaparecer,

dejando al stack en su estado original .
El identificador ó nombre que se la haya dado a una variable es sólo relevante entonces, para la función que la haya definido, pudiendo existir entonces variables que tengan el mismo nombre, pero definidas en funciones distintas, sin que haya peligro alguno de confusión .
La ubicación de estas variables locales, se crea en el momento de correr el programa, por lo que no poseen una dirección prefijada, esto impide que el compilador las pueda inicializar previamente. Recuerdese entonces que, si no se las inicializa expresamente en el momento de su definición, su valor será indeterminado (basura) .

VARIABLES LOCALES ESTATICAS

Las variables locales vistas hasta ahora, nacen y mueren con cada llamada y finalización de una función, sin embargo muchas veces sería util que mantuvieran su valor, entre una y otra llamada a la función sin por ello perder su ámbito de existencia, es decir seguir siendo locales sólo a la función que las defina. En el siguiente ejemplo veremos que esto se consigue definiendo a la variable con el prefacio static.

VARIABLES DE REGISTRO

Otra posibilidad de almacenamiento de las variables locales es, que en vez de ser mantenidas en posiciones de la memoria de la computadora, se las guarde en registros internos del Microprocesador que conforma la CPU de la misma .
De esta manera el acceso a ellas es mucho más directo y rápido, aumentando la velocidad de ejecución del programa. Se suelen usar registros para almacenar a los contadores de los FOR, WHILE, etc.
Lamentablemente, en este caso no se puede imponer al compilador, este tipo de variable, ya que no tenemos control sobre los registros libres en un momento dado del programa, por lo tanto se SUGIERE, que de ser posible, ubique la variable del modo descripto. El prefacio en éste caso será :
register int var_reg ;
Hay que recalcar que esto es sólo válido para variables LOCALES, siendo imposible definir en un registro a una global. Por otra parte las variables de registro no son accesibles por dirección, como se verá más adelante .

VARIABLES EXTERNAS

Al DEFINIR una variable, como lo hemos estado haciendo hasta ahora, indicamos al compilador que reserve para la misma una determinada cantidad de memoria, (sea en el segmento de memoria de datos, si es global ó en el stack, si es local), pero debido a que en C es normal la compilación por separado de pequeños módulos, que componen el programa completo, puede darse el caso que una función escrita en un archivo dado, deba usar una variable global definida en otro archivo. Bastará para poder hacerlo, que se la DECLARE especificando que es EXTERNA a dicho módulo, lo que implica que está definida en otro lado .
Supongamos que nuestro programa está compuesto por sólo dos módulos: mod_prin.c y mod_sec.c los cuales se compilarán y enlazarán juntos, por medio del compilador y el linker, por ejemplo corriendo: bcc mod_prin.c mod_sec.c si usaramos el compilador de Borland .
Si en el primer módulo (mod_prin.c) aparece una variable global, definida como
double var1 = 5 ;
El segundo módulo, ubicado en un archivo distinto de aquel, podrá referenciarla mediante la declaración de la misma :
extern double var1 ;

Conceptos básicos

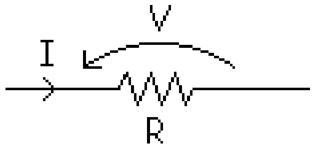
Para el correcto conocimiento de la electrónica es necesario saber algunas leyes y teoremas fundamentales como la Ley de Ohm, las Leyes de Kirchhoff, y otros teoremas de circuitos.

Ley de Ohm

Cuando una resistencia es atravesada por una corriente se cumple que:

$$V = I \cdot R$$

- Donde V es la tensión que se mide en voltios (V).
- Donde I es la intensidad de la corriente que atraviesa la resistencia, y que se mide en Amperios (A).
- Donde R es la resistencia que se mide en Ohmios (Ω).

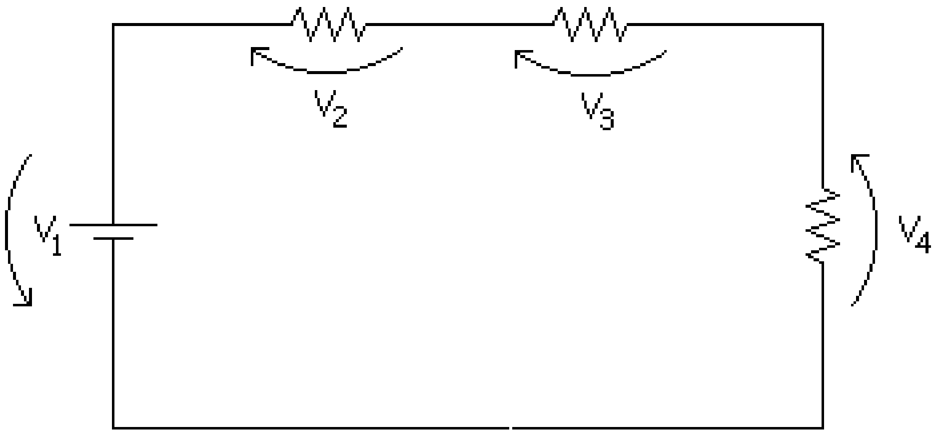


Leyes de Kirchhoff

Ley de Kirchhoff de tensiones

La suma de las caídas de tensiones de todos los componentes de una malla cerrada debe ser igual a cero.

$$\sum V = 0$$

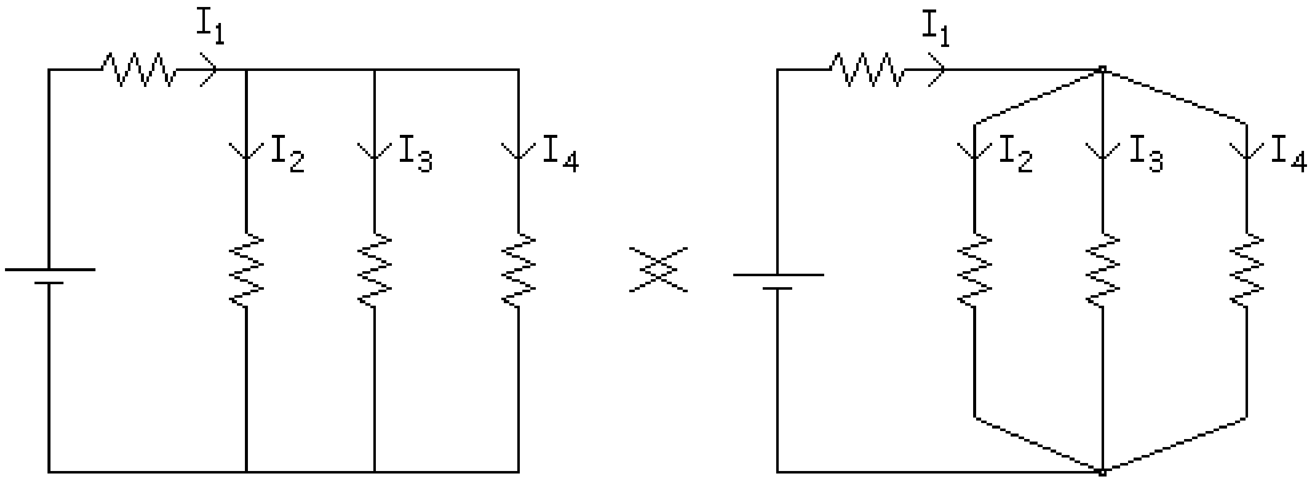


$$V_2 + V_3 + V_4 - V_1 = 0$$

Ley de Kirchhoff de corrientes

La suma de corrientes entrantes en un nodo es igual a la suma de corrientes salientes del nodo.

$$\sum I_{ENTRANTES} = \sum I_{SALIENTES}$$



$$I_1 = I_2 + I_3 + I_4$$

Resistencias

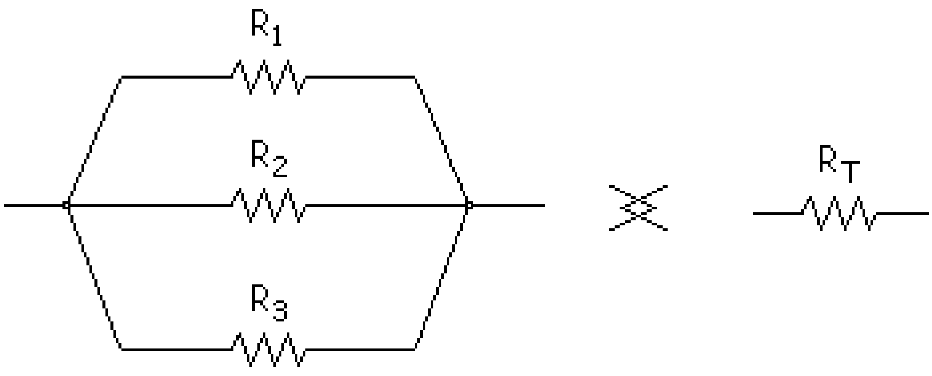
Resistencias en serie

Dos o más resistencias en serie (que les atraviesa la misma intensidad) es equivalente a una única resistencia cuyo valor es igual a la suma de las resistencias.

$$R_T = R_1 + R_2$$

Resistencias en paralelo

Cuando tenemos dos o más resistencias en paralelo (que soportan la misma tensión), pueden ser sustituidas por una resistencia equivalente, como se ve en el dibujo:



el valor de esa resistencia equivalente (R_T) lo conseguimos mediante esta expresión:

$$\frac{1}{R_T} = \sum \frac{1}{R_i}$$

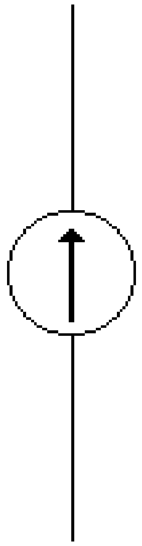
$$\frac{1}{R_T} = \frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_3}$$

Generadores

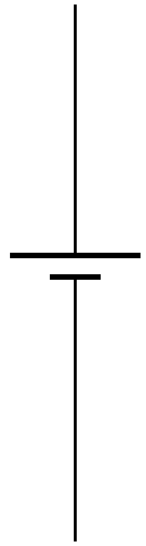
Generadores de Continua

Pueden ser tanto fuentes de corriente como de tensión, y su utilidad es suministrar corriente o tensión, respectivamente de forma continua.

Generador de corriente continua



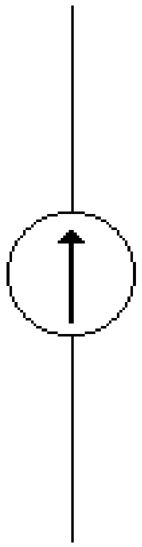
Generador de tensión continua



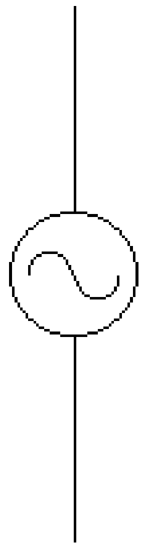
Generadores de Alterna

Pueden ser tanto fuentes de corriente como de tensión, y su utilidad es suministrar corrientes o tensiones, respectivamente de forma alterna (por ejemplo: de forma senoidal, de forma triangular, de forma cuadrada., etc....).

Generador de corriente alterna



Generador de tensión alterna

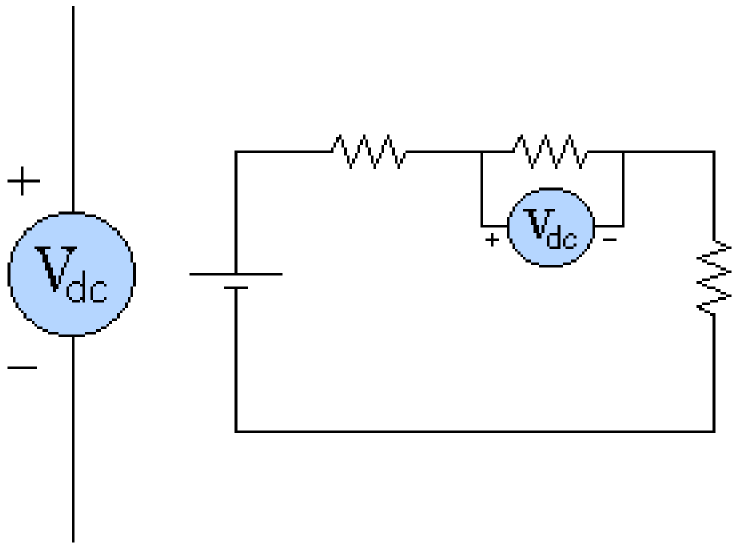


Aparatos de medición.

Voltímetro.

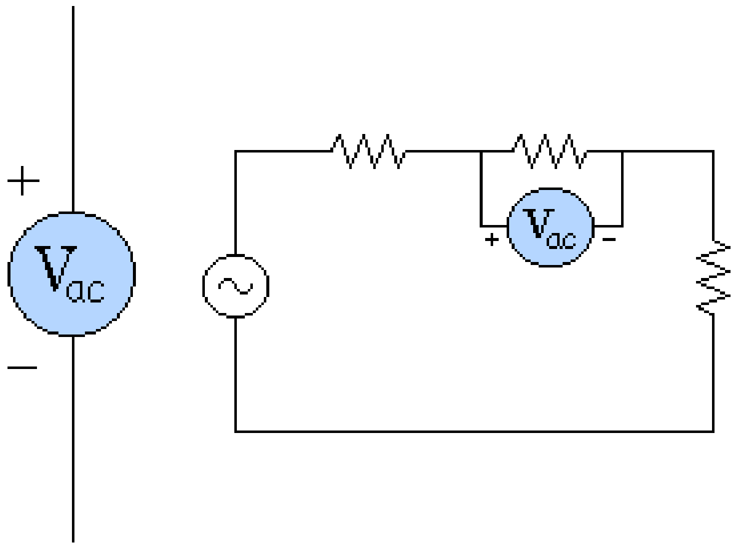
Aparato que mide tensiones eficaces tanto en continua como en alterna, y su colocación es de forma obligatoria en "paralelo" al componente sobre el cual se quiere medir su tensión.

Voltímetro de continua



dc = direct current (corriente directa, corriente de contínua)

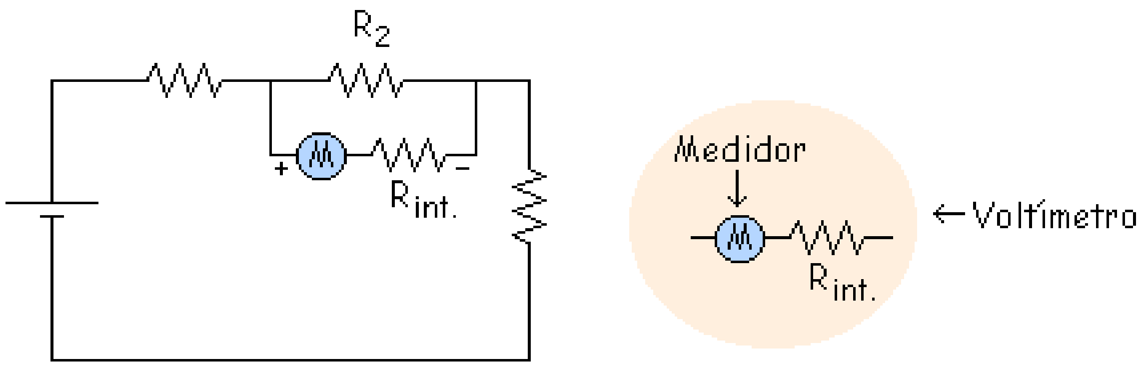
Voltímetro de alterna



ac = altern current (corriente alterna)

Errores al medir con voltímetros

Al medir con un voltímetro se comete un pequeño error porque dentro del voltímetro hay un resistencia interna ($R_{int.}$), que tiene un valor muy grande (se suele aproximar a infinito).

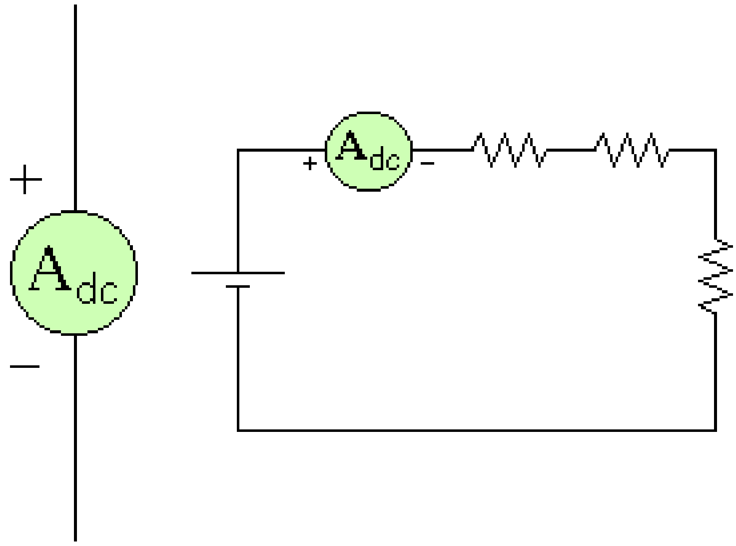


$R_2 // R_{int.} \simeq R_2$

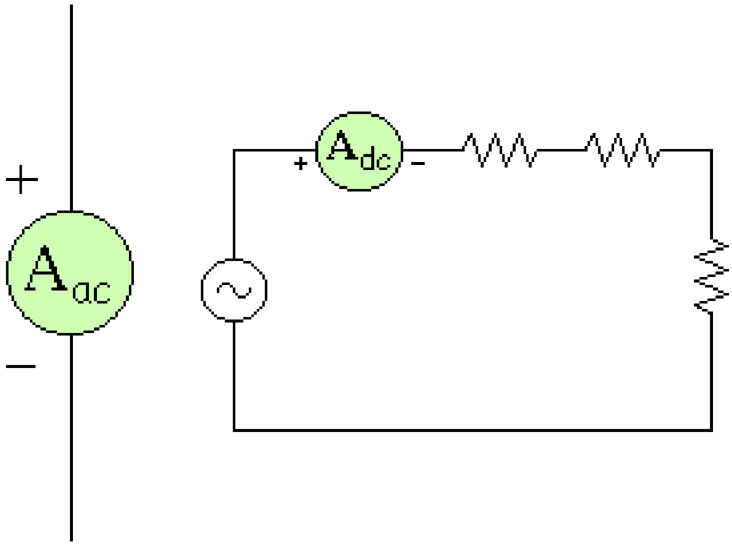
Amperímetro.

Aparato que mide el valor medio de la corriente, y su colocación es de forma obligatoria en "serie" con el componente del cual se quiere saber la corriente que le atraviesa.

Amperímetro de continua

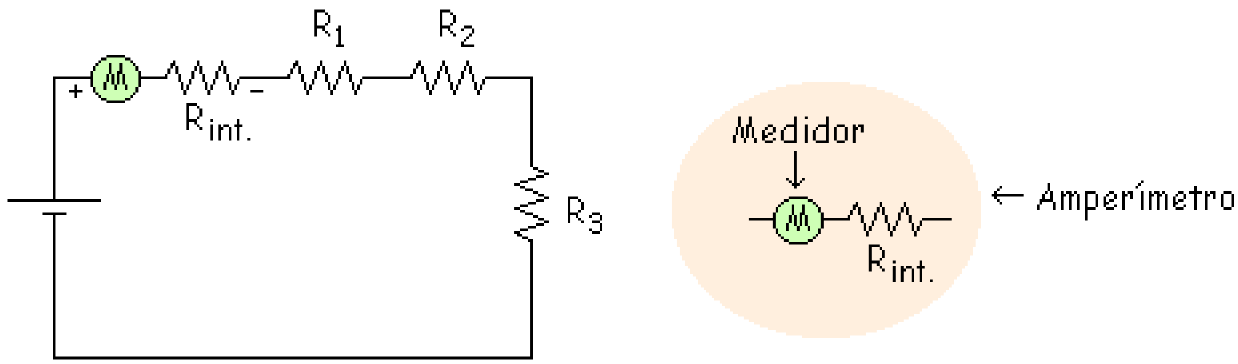


Amperímetro de alterna



Errores al medir con amperímetros

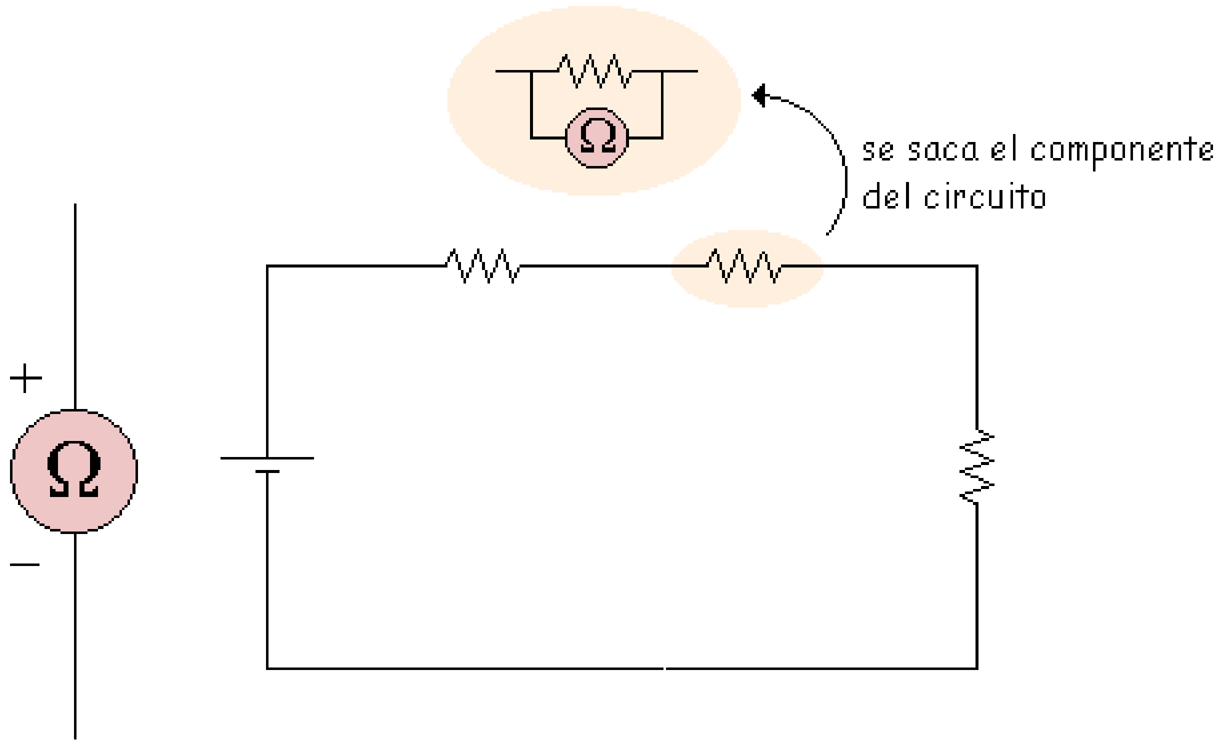
Como ocurre con el voltímetro, al medir con le amperímetro se comete un error debido a una resistencia interna ($R_{int.}$) de valor muy pequeño (se suele aproximar a cero).



$$R_{int.} + R_1 + R_2 + R_3 \simeq R_1 + R_2 + R_3$$

Óhmetro

Aparato que mide el valor de las resistencias, y que de forma obligatoria hay que colocar en paralelo al componente estando éste separado del circuito (sin que le atraviese ninguna intensidad). Mide resistencias en Ohmios (Ω).



Errores al medir con óhmetros

Como se ha visto anteriormente, todo aparato de medición comete un error que a veces se suele despreciar, con los óhmetros ocurre lo mismo, aunque se desprecie ese error hay que tener en cuenta que se suele hacer una pequeña aproximación.

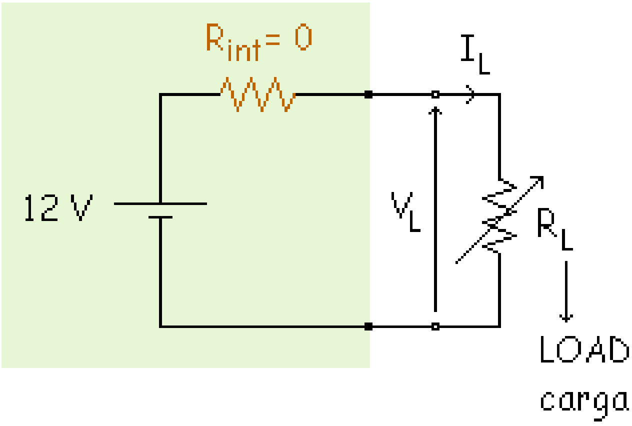
anterior/principal/siguiente

Fuentes de tensión

Los circuitos electrónicos deben poseer para su funcionamiento adecuado de al menos una fuente de energía eléctrica, que debe ser una fuente de tensión o de corriente.

Fuente de tensión ideal

Es una fuente de tensión que produce una tensión de salida constante, es una Fuente de Tensión con Resistencia interna cero. Toda la tensión va a la carga R_L.



$V_L = 12$ V. (cte)

R_L resistencia de carga
V_L tensión en la carga
I_L corriente en la carga

Ley de Ohm $V_L = R_L \cdot I_L$

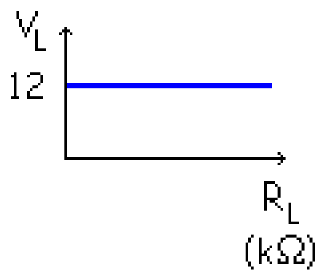
Si varía $R_L \Longrightarrow$ varía I_L

$$I_L \downarrow = \frac{V_L}{R_L \uparrow}$$

$R_L = 1\ k\Omega$ $\quad I_L = \frac{12}{1} = 12$ mA

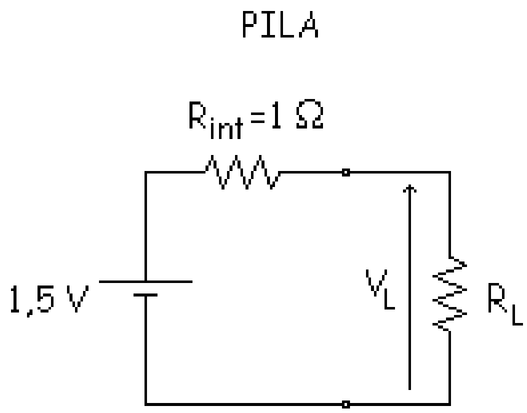
$R_L = 2\ k\Omega$ $\quad I_L = \frac{12}{2} = 6$ mA

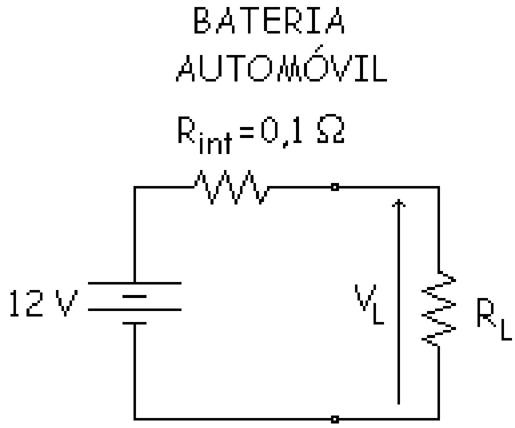
$R_L = 4\ k\Omega$ $\quad I_L = \frac{12}{4} = 3$ mA

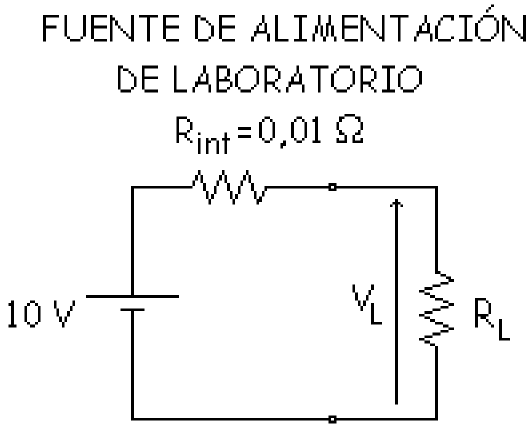


Fuente de tensión real

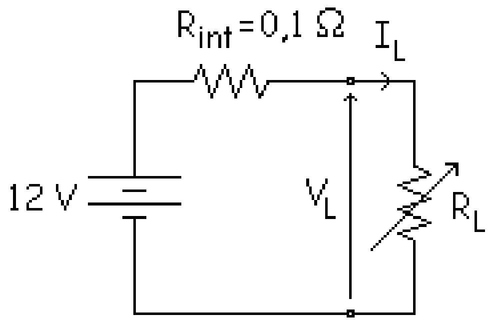
Algunos ejemplos de fuentes de tensión reales son:







Son las fuentes de tensión que tenemos en la realidad, como ya hemos dicho no existe una fuente ideal de tensión, ninguna fuente real de tensión puede producir una corriente infinita, ya que en toda fuente real tiene cierta resistencia interna.

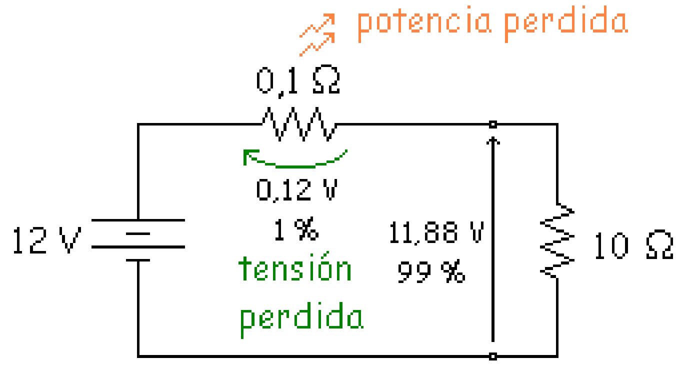


$$I_L = \frac{12}{0{,}1 + R_L}$$

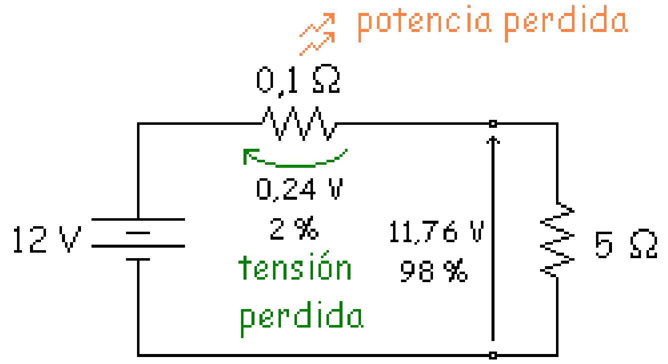
$$V_L = R_L \cdot \frac{12}{0{,}1 + R_L} \neq cte$$

Veamos que ocurre en 2 casos, cuando R_L vale 10 Ω y cuando vale 5 Ω.

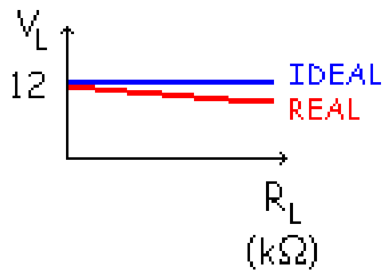
$R_L = 10\ k\Omega \qquad I_L = \dfrac{12}{0{,}1+10} = 11{,}88\ A \qquad V_L = 10 \cdot 11{,}88 = 11{,}88\ V$



$R_L = 5\ \Omega \qquad I_L = \dfrac{12}{0{,}1+5} = 2{,}353\ A \qquad V_L = 5 \cdot 2{,}353 = 11{,}76\ V$

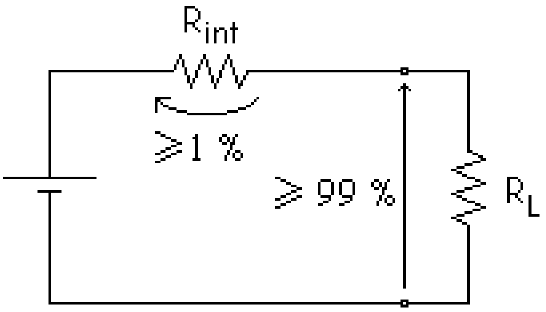


Ahora la tensión en la carga no es horizontal, esto es, no es ideal como en el caso anterior.



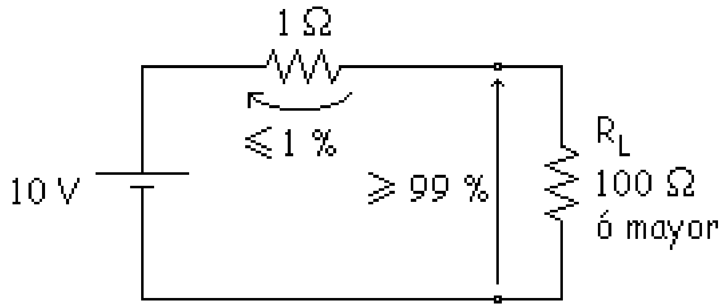
Fuente de tensión (aproximadamente) constante

Para que una fuente de tensión sea considerada como una "Fuente de tensión constante", se tiene que cumplir que la resistencia interna de la fuente (R_{int}) no este, esto es que sea despreciable. Para que despreciemos la R_{int} se tiene que cumplir:



$$R_L \geqslant 99\, R_{int} \approx \boxed{R_L \geqslant 100\, R_{int}} \quad ó \quad \boxed{R_{int} \leqslant \frac{R_L}{100}}$$

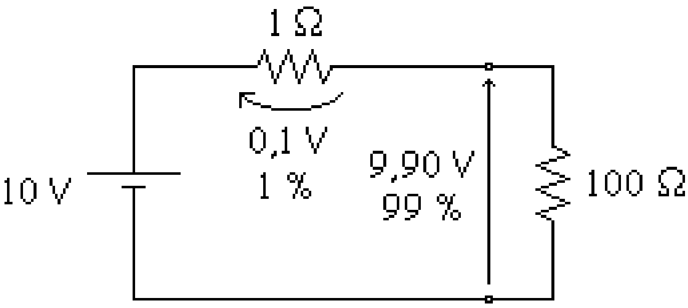
Solo se pierde el 1 % en el peor caso, por lo tanto se está aproximando a la fuente de tensión ideal.



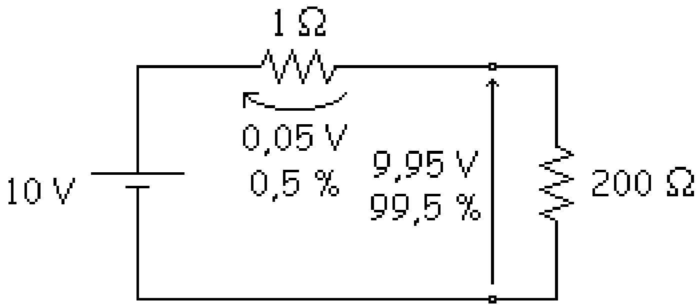
$$\text{Si} \quad R_L \geqslant 100\ \Omega \Longrightarrow \text{Fuente de tensión constante}$$

Veamos que ocurre en 2 valores diferentes de R_L.

$R_L = 100\ \Omega$



$R_L = 200\ \Omega$



$$V_L = 100 \cdot \frac{10}{1+100} = 9{,}90\ \text{V} \qquad V_L = 200 \cdot \frac{10}{1+200} = 9{,}95\ \text{V}$$

Resumen

- Fuente de tensión ideal es la que tiene una $R_{int.} = 0$ y produce en la salida una V_L = cte.
- Fuente de tensión real es la que tiene una determinada $R_{int.}$ En esta $R_{int.}$ hay una pérdida de tensión. El resto de tensión va a la carga que es la que se aprovecha.
- Fuente de tensión constante es la que tiene una $R_{int.} <= R_L/100$. La caída en la $R_{int.}$ es como mucho el 1 %, aproximadamente a la ideal, que es el 0 %.

Si tenemos que comparar dos fuentes de tensión, la mejor será la que tenga una $R_{int.}$ más pequeña (o sea la que más parecida a la ideal, que tiene una $R_{int.} = 0\ \Omega$).

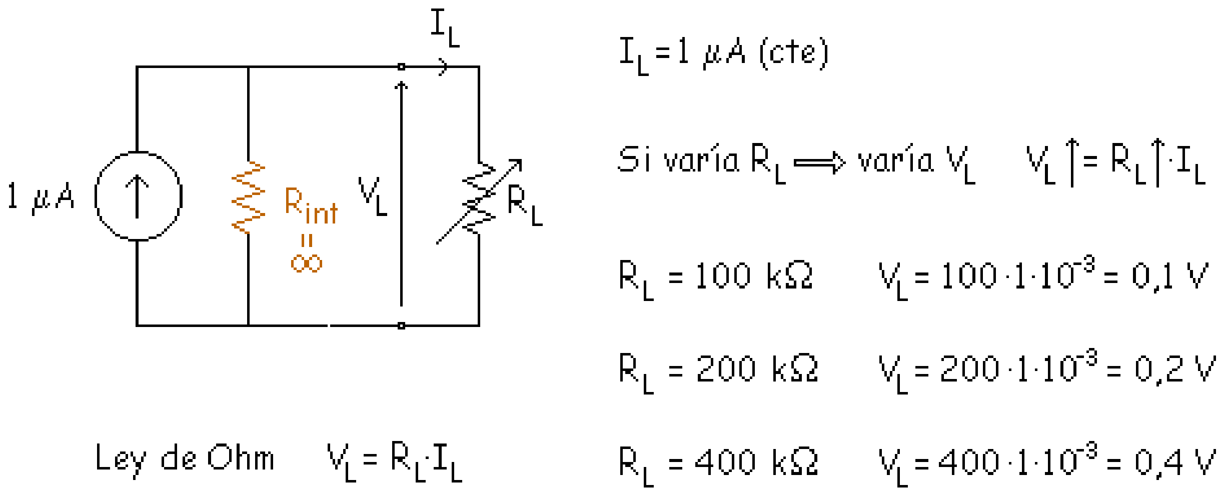
anterior/principal/siguiente

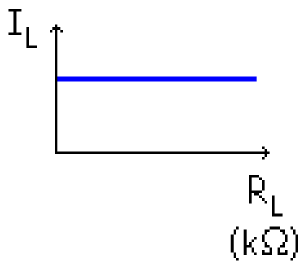
Fuentes de corriente

En el caso anterior de la fuente de tensión había una resistencia interna muy pequeña, pero una fuente de corriente es diferente, tiene una resistencia interna muy grande, así una fuente de corriente produce una corriente de salida que no depende del valor de la resistencia de carga.

Fuente de corriente ideal

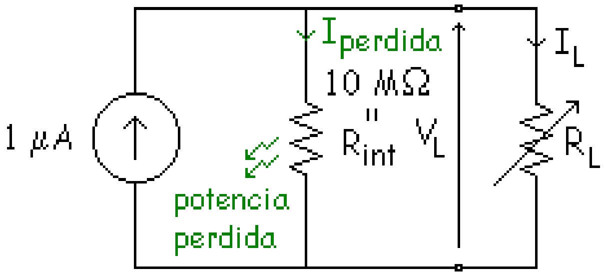
No existe, es ideal como en el anterior caso de la fuente de tensión ideal..





Fuente de corriente real

Son las fuentes que existen en la realidad.



$$V_L = 1\cdot(10//R_L)$$

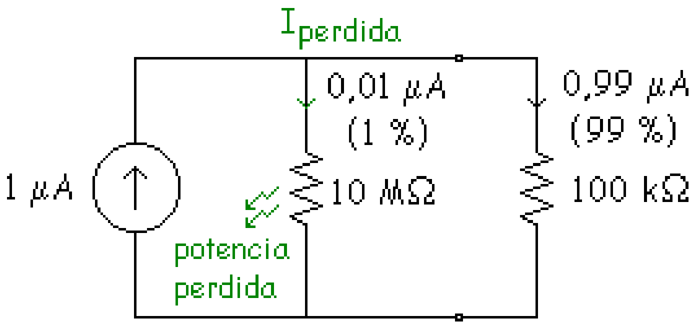
$$I_L = \frac{V_L}{R_L} = \frac{1\cdot(10//R_L)}{R_L} \neq cte$$

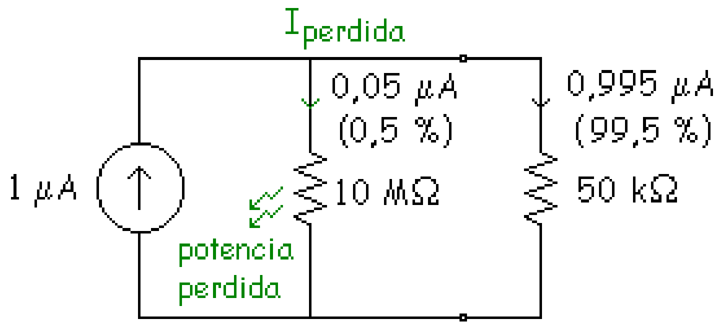
en carga $I_L < 1\ \mu A$

Veamos que ocurre con los diferentes valores de R_L.

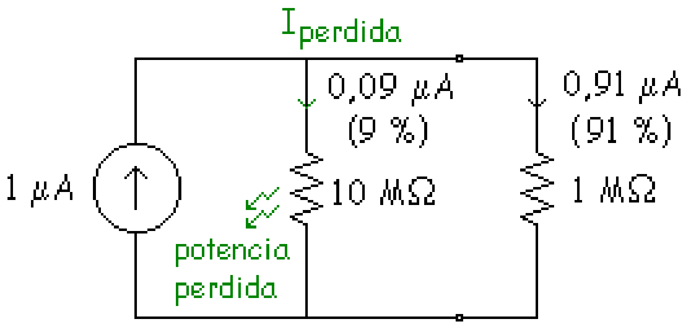
$R_L = 100\ k\Omega \qquad I_L = \dfrac{1\cdot(10//0{,}1)}{R_L} = 0{,}99\ \mu A$

$R_L = 50\ k\Omega \qquad I_L = \dfrac{1\cdot(10//0{,}05)}{R_L} = 0{,}995\ \mu A$

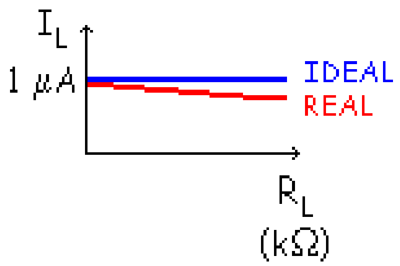




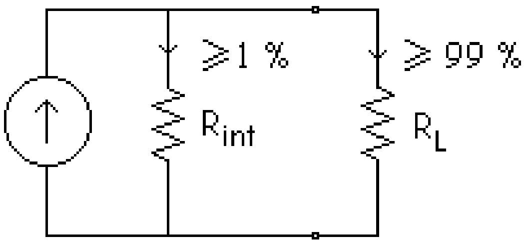
$R_L = 1\ M\Omega \qquad I_L = \dfrac{1\cdot(10//0{,}05)}{R_L} = 0{,}995\ \mu A$



Con esto vemos que una fuente de corriente funciona mejor cuando su resistencia interna es muy alta, mientras que una fuente de tensión funciona mejor cuando su resistencia interna es muy baja. La intensidad de carga tiene esta forma:

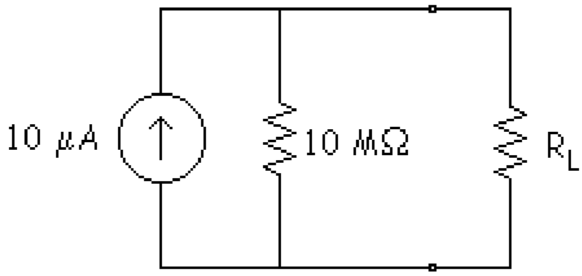


Fuente de corriente (aproximadamente) constante



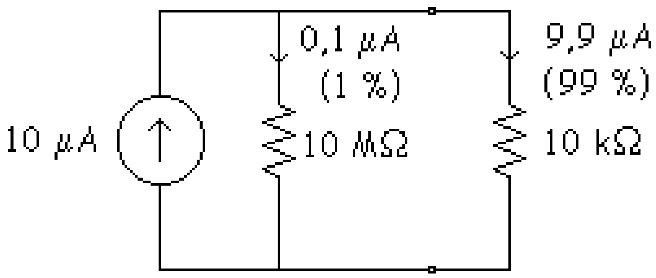
$$R_L \leqslant \frac{R_{int}}{99} \approx \boxed{R_L \leqslant \frac{R_{int}}{100}} \text{ ó } \boxed{R_{int} \geqslant 100\ R_L}$$

Solo se pierde el 1 % en el peor caso. Con esto nos aproximamos a la fuente de corriente ideal. Veamos 2 valores diferentes de R_L.



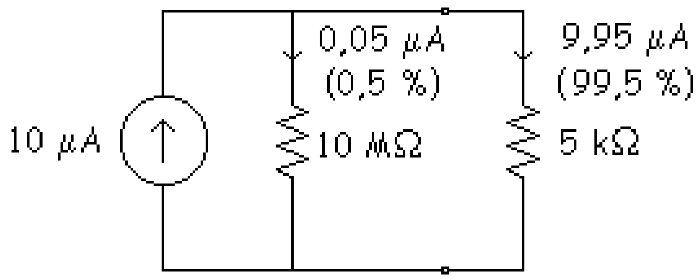
Si $R_L \leqslant 10\ k\Omega \Longrightarrow$ Fuente de corriente cte

$R_L = 10\ k\Omega$



$$I_L = \frac{10 \cdot (1 // 0{,}01)}{0{,}01} = 9{,}9\ \mu A$$

$R_L = 5\ k\Omega$



$$I_L = \frac{10 \cdot (1 // 0{,}005)}{0{,}005} = 9{,}95\ \mu A$$

Resumen

- Fuente de corriente ideal es la que tiene una R_{int} = 8 y produce en la salida una I_L = cte.

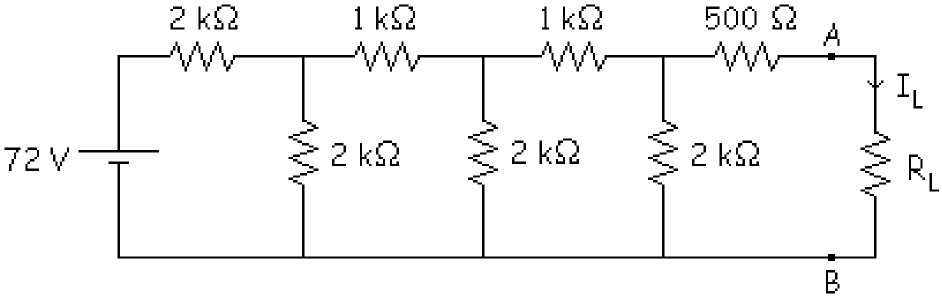
- Fuente de corriente real es la que tiene una determinada R_{int}. En esta hay pérdida de corriente. El resto de la corriente va a la carga que es la que se aprovecha.
- Fuente de corriente constante es la que tiene una $R_{int} >= 100R_L$. La corriente que se pierde por la R_{int} es como mucho el 1 %, aproximadamente a la ideal, que es el 0 %.

Si tenemos que comparar 2 fuentes de corriente, la mejor será la que tenga una R_{int} más grande (o sea la más parecida a la ideal, que tiene una R_{int} = 8).

anterior/principal/siguiente

Teorema de Thévenin

Vamos a dar dos teoremas (Thévenin y Norton) que nos van a servir para hacer más fácil (simplificar) la resolución de los circuitos.

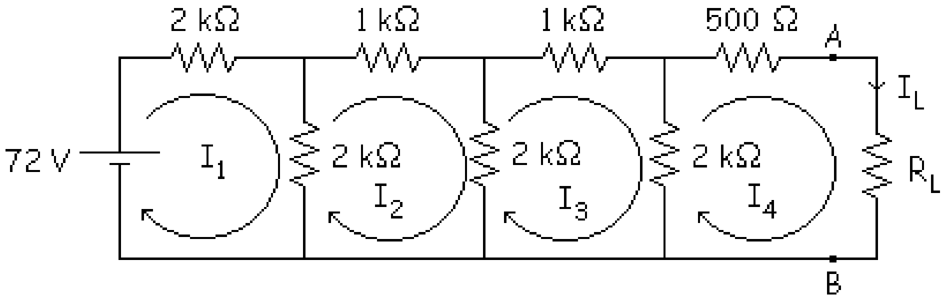


a) Calcular la I_L cuando R_L = 1,5 kΩ.

b) Calcular la I_L cuando R_L = 3 kΩ.

c) Calcular la I_L cuando R_L = 4,5 kΩ.

- Ley de Kirchhoff de tensiones.



a)

$$\left.\begin{aligned} &-72+2I_1+2\cdot(I_1-I_2)=0 \\ &2\cdot(I_2-I_1)+1I_2+2\cdot(I_2-I_3)=0 \\ &2\cdot(I_2-I_3)+1I_3+2\cdot(I_3-I_4)=0 \\ &2\cdot(I_4-I_3)+0{,}5I_4+1{,}5\,I_4=0 \end{aligned}\right\}\quad \begin{aligned} &I_1= \\ &I_2= \\ &I_3= \\ &I_4=I_L \end{aligned}$$

b)

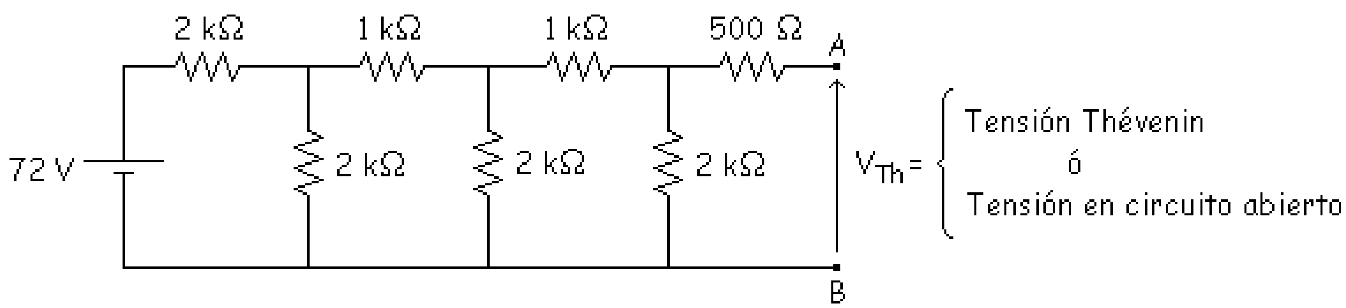
$$\left.\begin{aligned} &-72+2I_1+2\cdot(I_1-I_2)=0 \\ &2\cdot(I_2-I_1)+1I_2+2\cdot(I_2-I_3)=0 \\ &2\cdot(I_2-I_3)+1I_3+2\cdot(I_3-I_4)=0 \\ &2\cdot(I_4-I_3)+0{,}5I_4+3\,I_4=0 \end{aligned}\right\}\quad \begin{aligned} &I_1= \\ &I_2= \\ &I_3= \\ &I_4=I_L \end{aligned}$$

c)

$$\left.\begin{array}{l} -72+2I_1+2\cdot(I_1-I_2)=0 \\ 2\cdot(I_2-I_1)+1I_2+2\cdot(I_2-I_3)=0 \\ 2\cdot(I_2-I_3)+1I_3+2\cdot(I_3-I_4)=0 \\ 2\cdot(I_4-I_3)+0{,}5I_4+4{,}5I_4=0 \end{array}\right\} \begin{array}{l} I_1= \\ I_2= \\ I_3= \\ I_4=I_L \end{array}$$

- **Thévenin.**

1. Quitar la carga R_L.

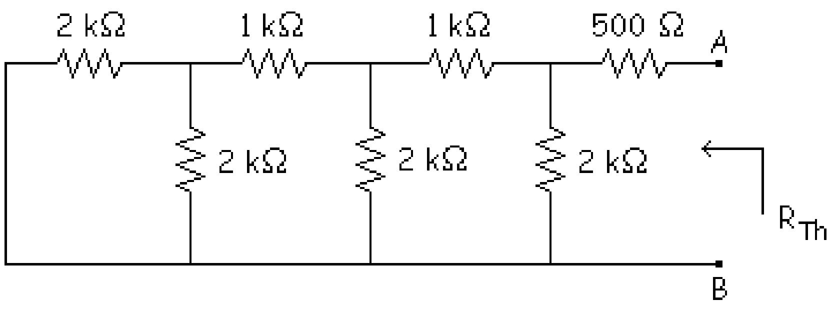


2. Hacemos mallas y calculamos V_{th}:

$$\left.\begin{array}{l} -72+2I_1+2I_2=0 \\ 2I_2-2I_1+I_2+2I_2-2I_3=0 \\ 2I_3-2I_2+I_3+2I_3=0 \end{array}\right\} \begin{array}{l} I_1= \\ I_2= \\ I_3= \end{array}$$

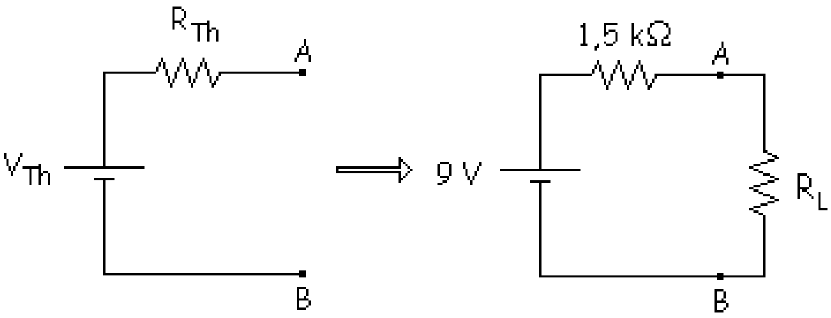
$$V_{Th}=2I_3$$

3. Cortocircuitar las fuentes de tensión independientes y abrir las fuentes de corriente independientes.



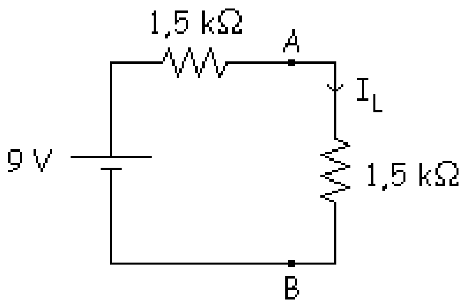
$$R_{Th}=\{\,[(2\,//\,2)-1\,]\,//\,1\,\}\,//\,2+0{,}5=1{,}5\ k\Omega$$

4. Unir la carga al circuito equivalente conseguido.



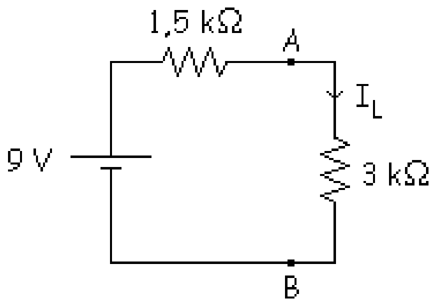
Ahora aplicando Thévenin es mucho más fácil resolver el problema que teníamos.

a)



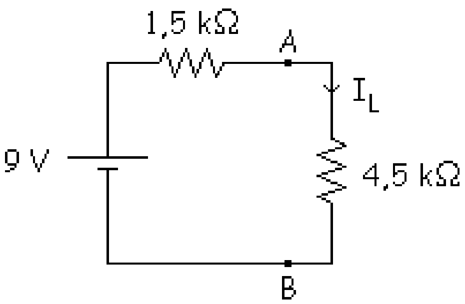
$$I_L = \frac{9}{1{,}5 + 1{,}5} = 3 \text{ mA}$$

b)



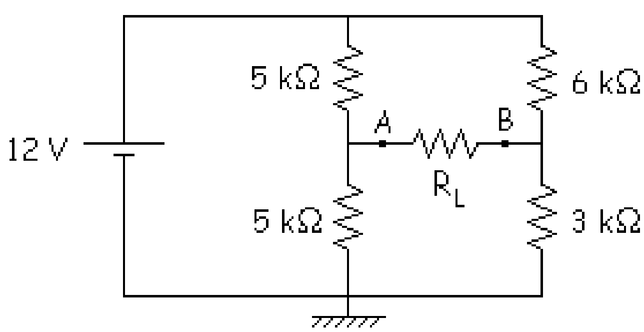
$$I_L = \frac{9}{1{,}5 + 3} = 2 \text{ mA}$$

c)

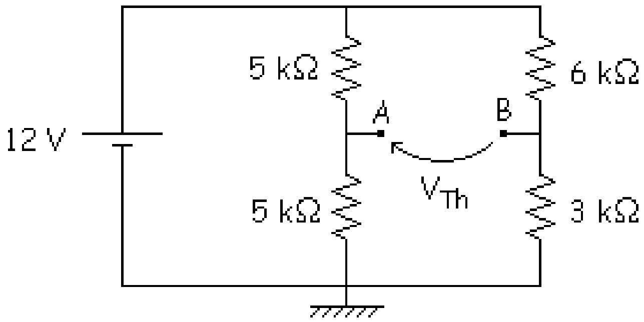


$$I_L = \frac{9}{1{,}5 + 4{,}5} = 1{,}5 \text{ mA}$$

Ejemplo: Calcular el equivalente de Thévenin del siguiente circuito:



1.



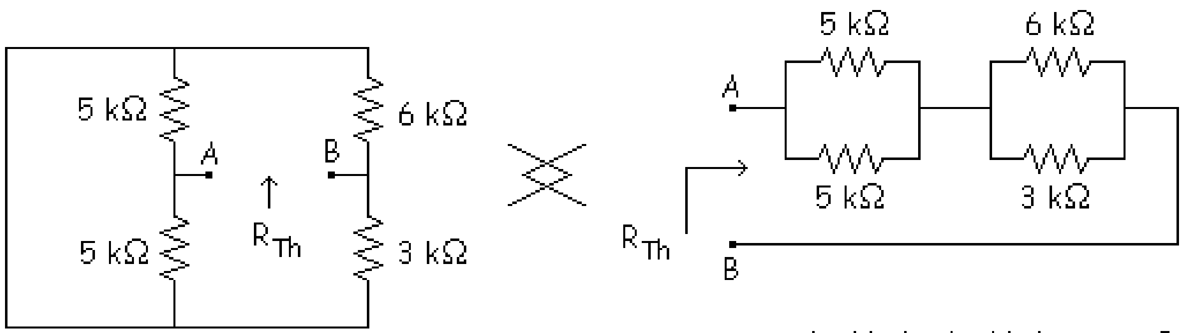
2.

$$V_A = I_A \cdot R \implies V_A = \frac{12}{5+5} \cdot 5 = 6\ V$$

$$V_B = I_B \cdot R \implies V_B = \frac{12}{6+3} \cdot 3 = 4\ V$$

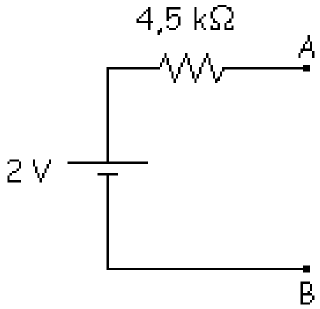
$$V_{Th} = V_{AB} = V_A - V_B = 6 + 4 = 2\ V$$

3.



$$R_{Th} = (5\ //\ 5) + (6\ //\ 3) = 4{,}5\ k\Omega$$

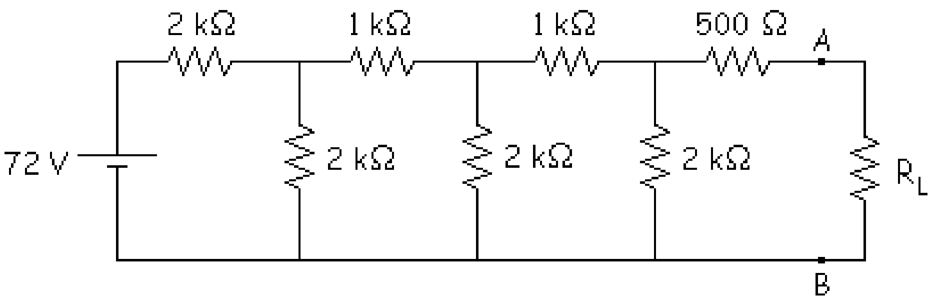
4.



anterior/principal/siguiente

Teorema de Norton

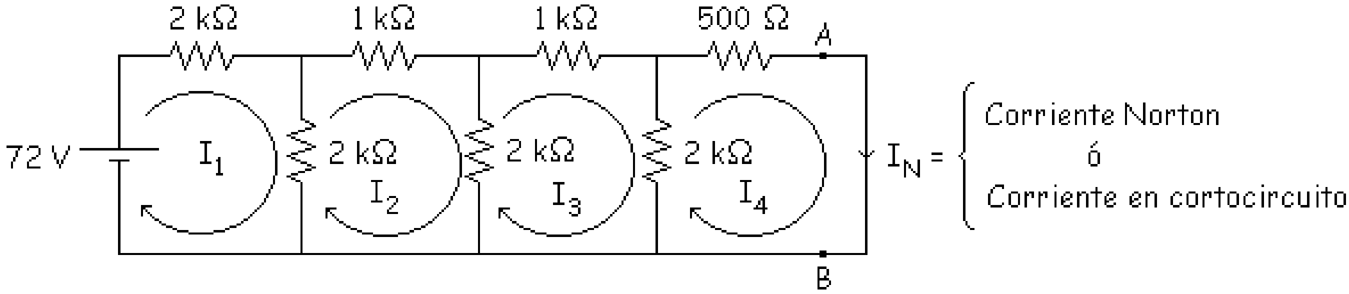
Este teorema esta muy relacionado con el Teorema de Thévenin. Resolveremos el problema anterior usando el teorema de Norton.



a) Calcular la I_L cuando R_L = 1,5 kΩ.
b) Calcular la I_L cuando R_L = 3 kΩ.
c) Calcular la I_L cuando R_L = 4,5 kΩ.

- **Norton.**

1. Quitar la carga R_L y poner un cortocircuito (R_L = 0).



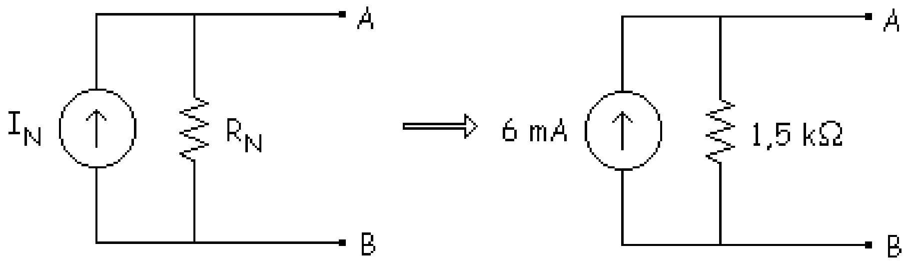
2. Hacemos mallas y calculamos V_{th}:

$$\left.\begin{array}{l} -72+2I_1+2\cdot(I_1-I_2)=0 \\ 2\cdot(I_2-I_1)+1I_2+2\cdot(I_2-I_3)=0 \\ 2\cdot(I_2-I_3)+1I_3+2\cdot(I_3-I_4)=0 \\ 2\cdot(I_4-I_3)+0{,}5I_4+1{,}5I_4=0 \end{array}\right\} \begin{array}{l} I_1= \\ I_2= \\ I_3= \\ I_4=I_N=6\ mA \end{array}$$

3. Cortocircuitar las fuentes de tensión independientes y abrir las fuentes de corriente independientes.

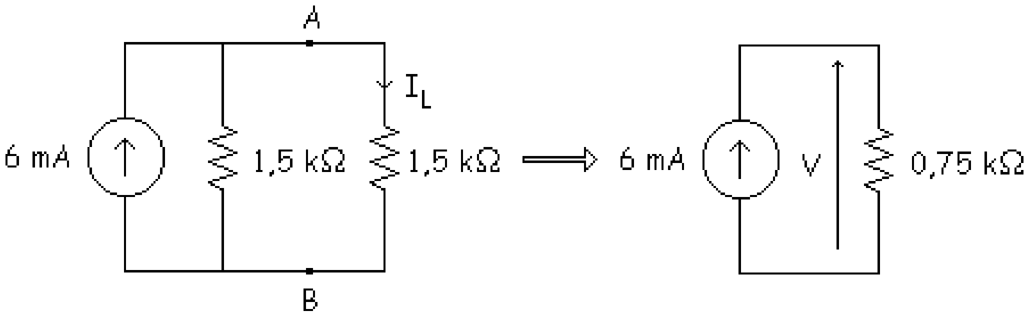
$R_N = R_{Th} = 1{,}5\ k\Omega$

4. Unir la carga al circuito equivalente conseguido.



Ahora aplicando Thévenin es mucho más fácil resolver el problema que teníamos.

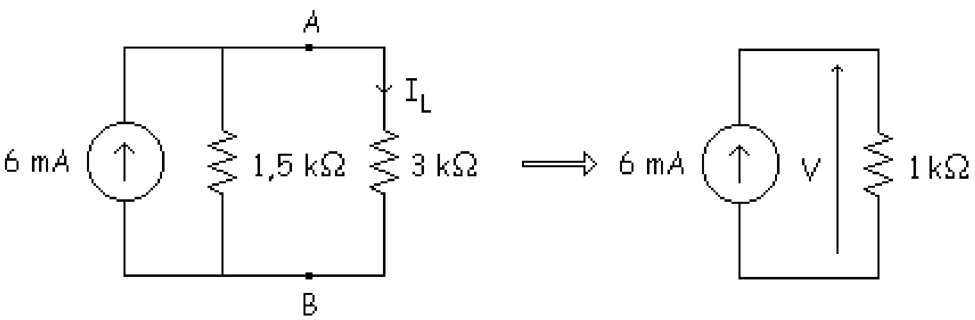
a)



$$V = 6 \cdot 0{,}75 = 4{,}5\ V$$

$$I_L = \frac{4{,}5}{1{,}5} = 3\ mA$$

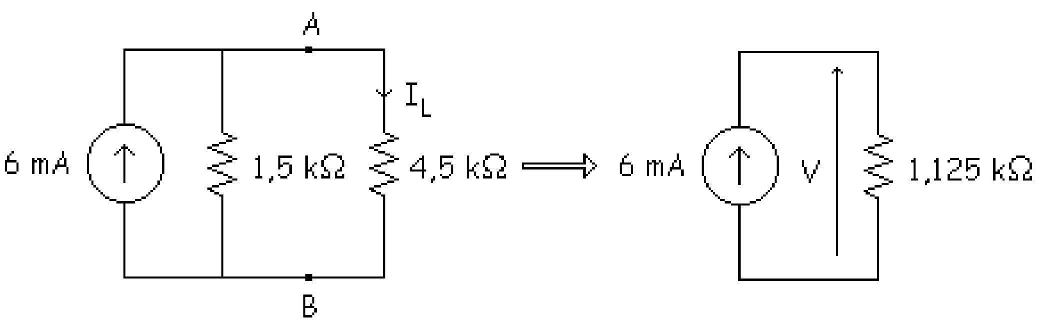
b)



$$V = 6 \cdot 1 = 6\ V$$

$$I_L = \frac{6}{3} = 2\ mA$$

c)

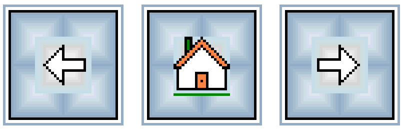


$$V = 6 \cdot 1{,}125 = 6{,}75\ V$$

$$I_L = \frac{6{,}75}{4{,}5} = 1{,}5\ mA$$

anterior/principal/siguiente

Paso de circuito Thévenin a circuito Norton y de circuito Norton a circuito Thévenin



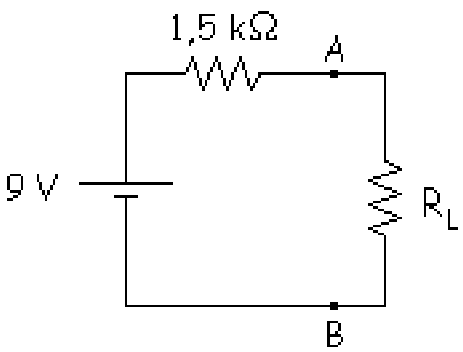
Paso de circuito Thévenin a circuito Norton

Paso de circuito Norton a circuito Thévenin

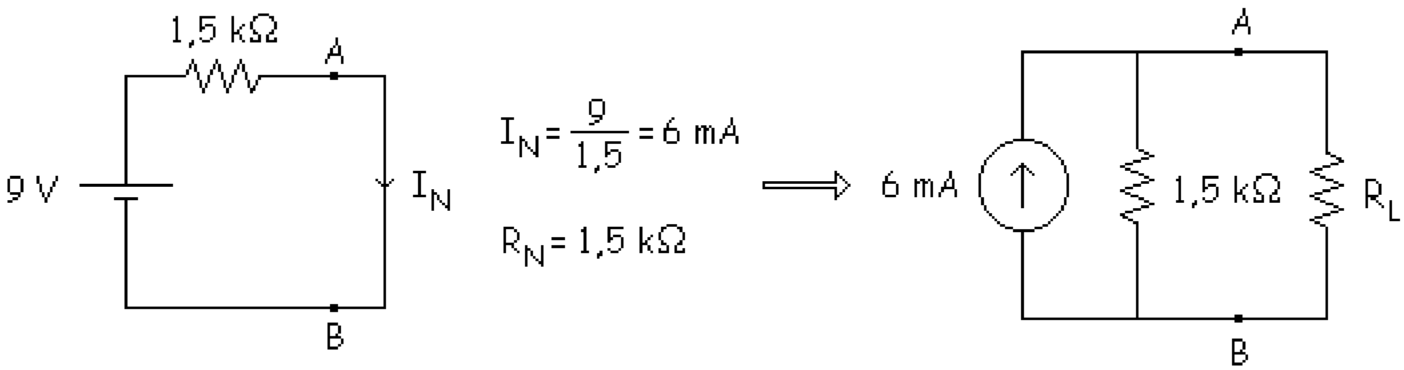
Como se ha dicho anteriormente los teoremas de Thénenin y Norton están relacionados, así se puede pasar de uno a otro.

Paso de circuito Thévenin a circuito Norton

Tenemos el circuito siguiente:

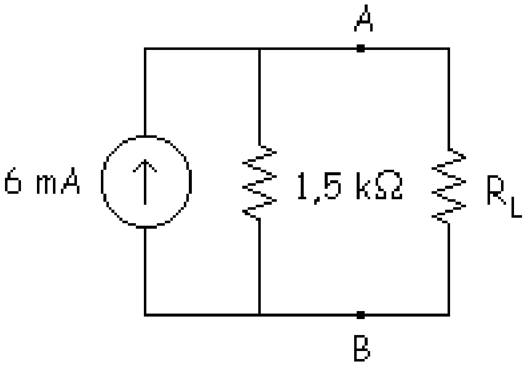


Cortocircuitamos la carga (R_L) y obtenemos el valor de la intensidad Norton, la R_N es la misma que la R_{Th}.

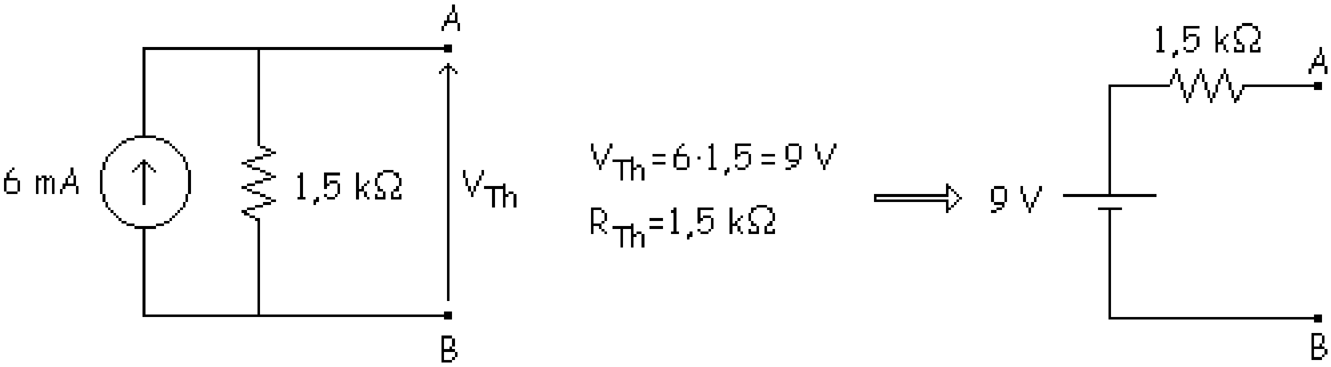


Paso de circuito Norton a circuito Thévenin

Tenemos este circuito:



Abrimos la carga (R_L) y calculamos la V_{Th}, la R_{Th} es la misma que la R_N.



anterior/**principal**/**siguiente**

Detección de averías

Cortocircuito

Circuito abierto

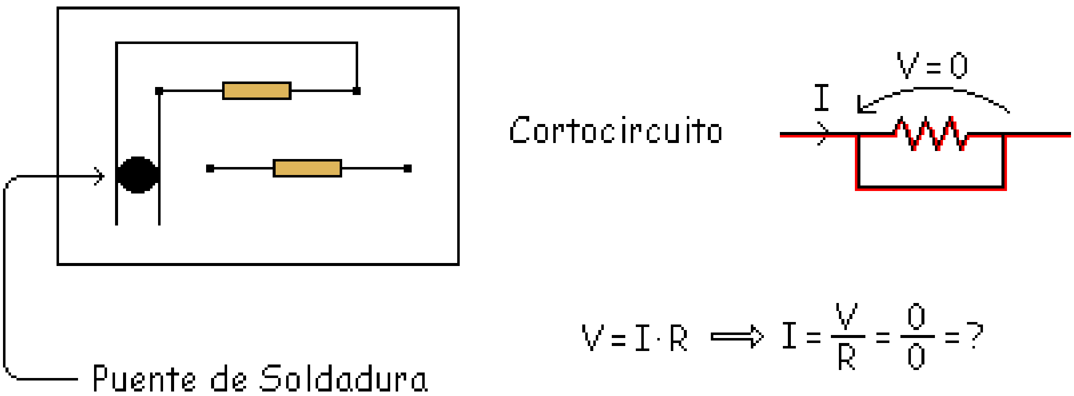
se trata de descubrir porque el circuito no funciona como debería. Los 2 tipos de averías más comunes son: dispositivo en cortocircuito y dispositivo en circuito abierto.

Cortocircuito

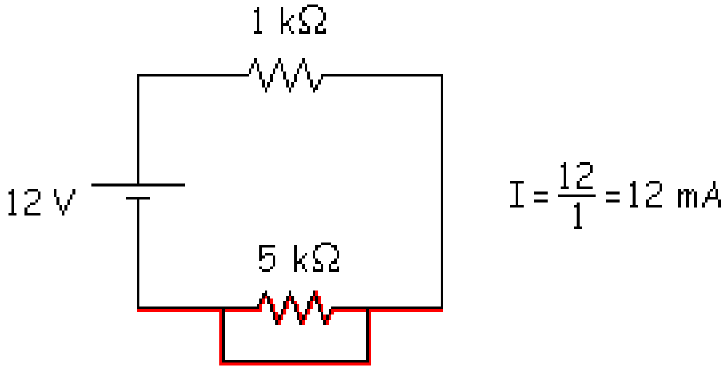
Sus características son:

- La tensión es cero en el dispositivo.
- La corriente es desconocida.

Una resistencia puede estar en cortocircuito si, por ejemplo, durante durante el horneado y soldadura de una tarjeta de circuito impreso, se cae una gota de soldadura y conecta 2 pistas cercanas, es un "Puente de Soldadura", esto es, cortocircuitar un dispositivo entre 2 pistas.



Hay que mirar en el resto del circuito para calcular la I.

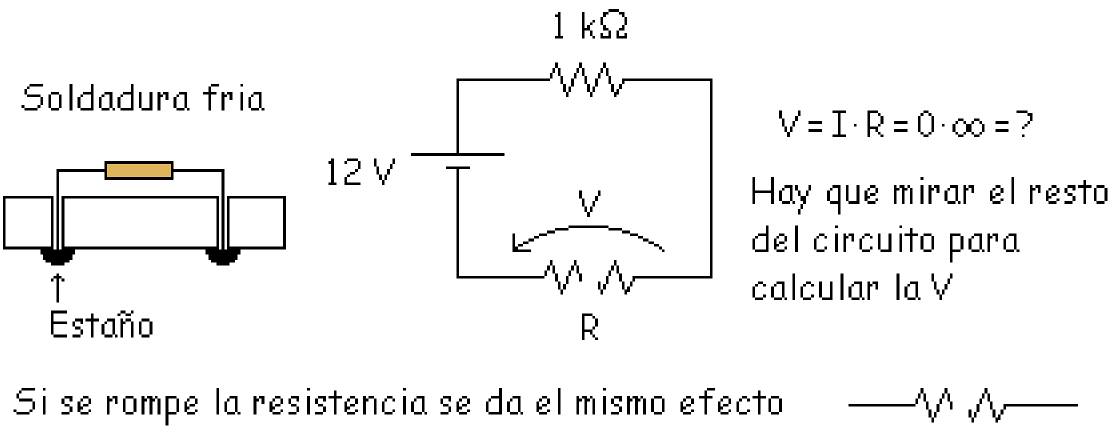


Circuito abierto

Se dan estas 2 características.

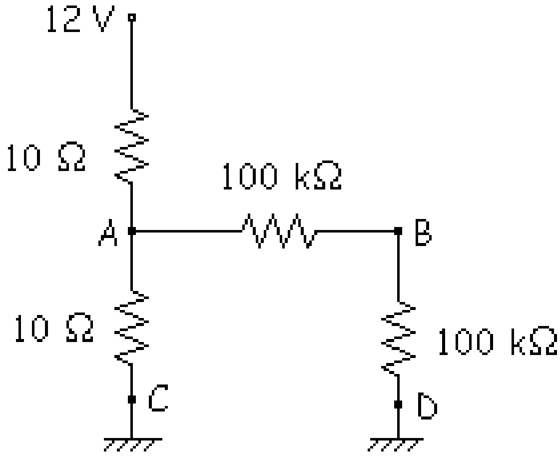
- La corriente es cero a través del dispositivo.
- La tensión es desconocida.

En circuitos impresos una mala soldadura significa la no conexión normalmente, esto es una "Unión de Soldadura Fría" y significa que el dispositivo está en circuito abierto.



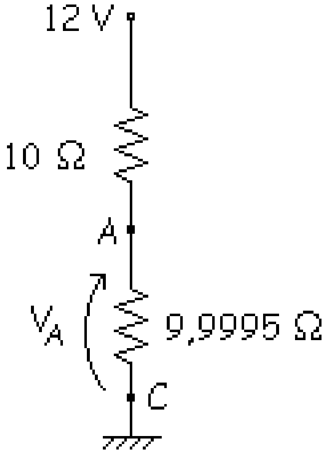
Las resistencias se convierten en circuitos abiertos cuando la potencia que disipan es excesiva.

Ejemplo:



Avería	V_A	V_B
No hay	+6 V	+3 V
R1 Cortocircuito	+12 V	+6 V
R1 Abierto	0 V	0 V
R2 Abierto	+12 V	+6 V

Primeramente no hay ninguna avería, hacemos el equivalente.



$$10\ \Omega\ //\ 200\ k\Omega = 9{,}9995\ \Omega$$

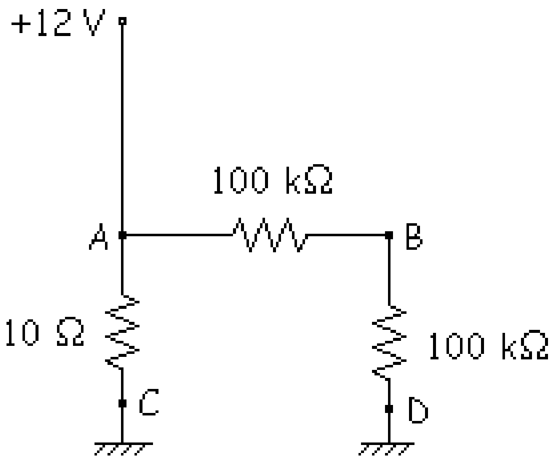
$$V_A = \frac{12}{10 + 9{,}9995} \cdot 9{,}9995 = 5{,}99985\ V$$

$$V_B = \frac{V_A}{2} = 2{,}999425\ V$$

De esa tensión V_A la mitad se disipa en la resistencia entre B y D de 100 kΩ y la otra mitad en la resistencia entre A y B de 100 kΩ.

Para detectar averías no hace falta hacer unos cálculos tan exactos, entonces tendríamos de forma aproximada V_A = 6 V y V_B = 3 V.

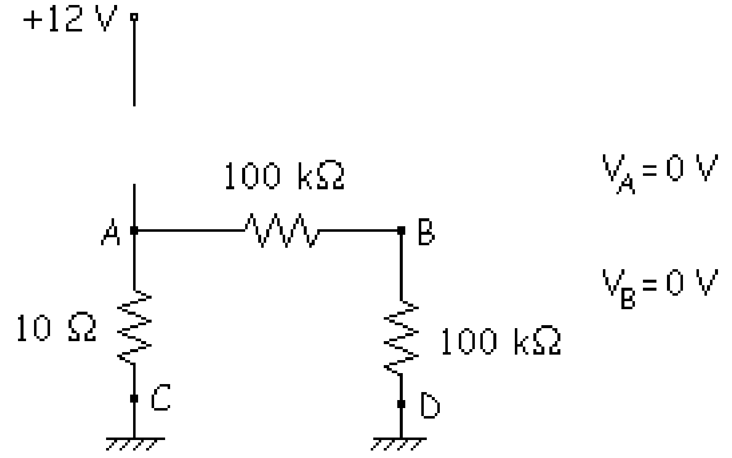
R1 en Cortocircuito:



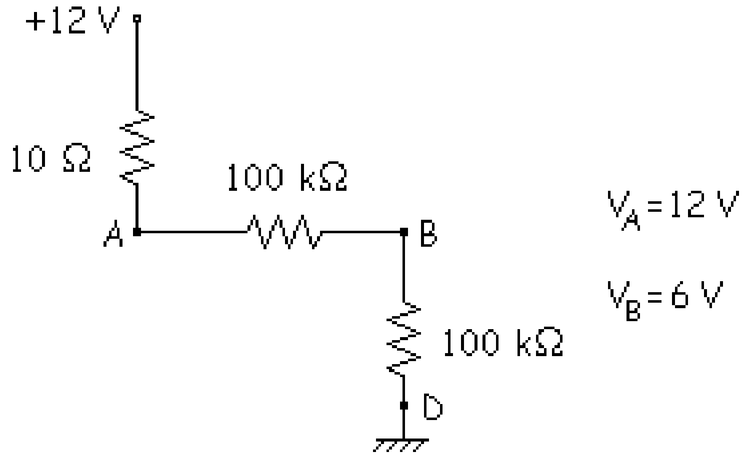
$V_A = 12\ V$

$V_B = 6\ V$

R1 en Circuito Abierto:



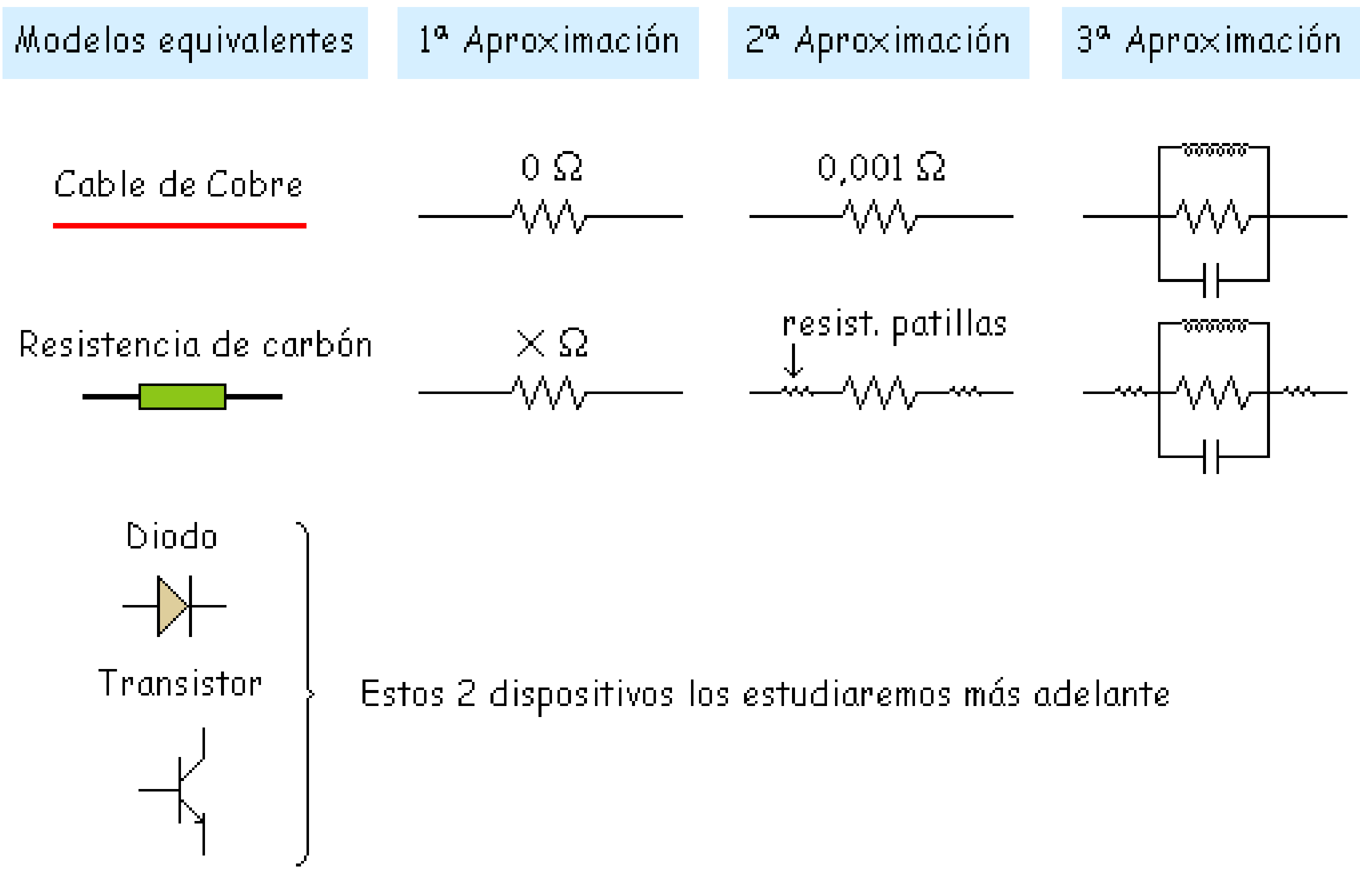
R2 en Circuito Abierto:



anterior/principal/siguiente

Aproximaciones

Para facilitar los cálculos se hacen aproximaciones, ya que hay ciertos valores que se pueden despreciar respecto a otros y que no influyen en gran medida en el resultado final, variándolo en un porcentaje muy pequeño respecto al resultado real. Las aproximaciones vistas hasta ahora son:



Mas adelante estudiaremos el diodo y el transistor y veremos que en estos 2 dispositivos también se usan 3 aproximaciones.

[anterior](#)/[principal](#)/[siguiente](#)

Problemas

[Problema 1.1](#)

[Problema 1.2](#)

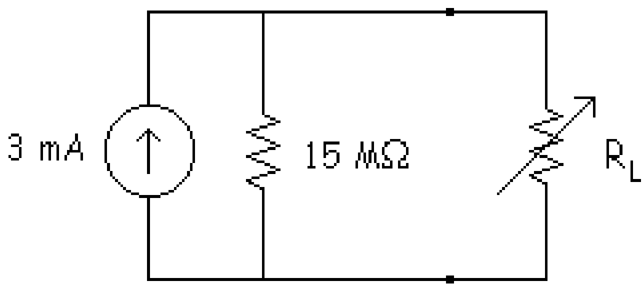
[Problema 1.3](#)

[Problema 1.4](#)

En este último apartado de este tema se resolverán algunos problemas relacionados con lo visto anteriormente.

Problema 1.1

En la figura se muestra una fuente de corriente de 2 mA con una resistencia de carga ajustable. Para que la fuente de corriente sea constante, ¿cuál el el máximo valor aceptable para la resistencia de carga?



Solución:

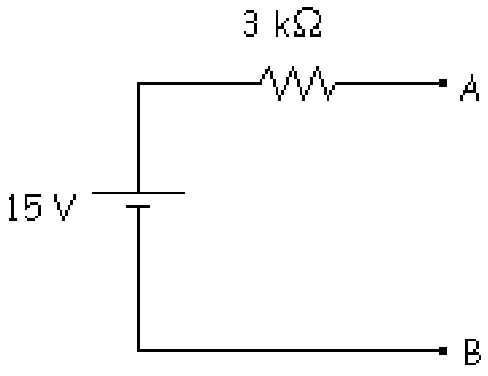
La fuente de corriente es constante cuando la resistencia de carga máxima permisible vale:

$$R_{L\,máx} = \frac{15\ \text{M}\Omega}{100} = 150\ \text{k}\Omega$$

La corriente por la carga será aproximadamente de 3 mA para cualquier resistencia de carga entre 0 y 150 kΩ. Mientras la resistencia de carga sea menor que 150 kΩ, podemos ignorar la resistencia interna de 15 MΩ y considerar que la fuente de corriente es ideal.

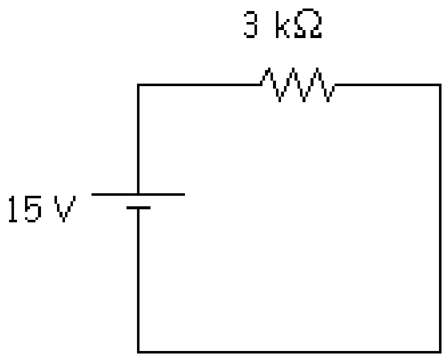
Problema 1.2

En la figura se muestra un circuito Thévenin. Conviértalo en un circuito Norton.



Solución:

En primer lugar, se cortocircuitarán los terminales de carga, como se muestra en la figura:



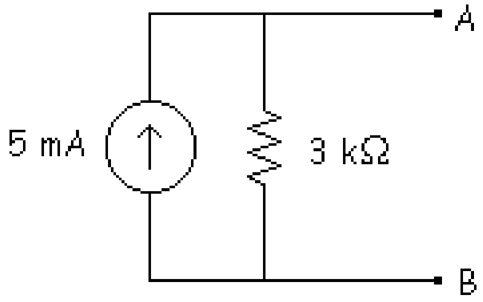
Con esto se calculará la corriente por la carga en este circuito, que es:

$$I_N = \frac{15\ V}{3\ k\Omega} = 5\ mA$$

Esta corriente de carga en cortocircuito es igual a la corriente de Norton. La resistencia Norton es igual a la resistencia Thévenin:

$$R_N = 3\ k\Omega$$

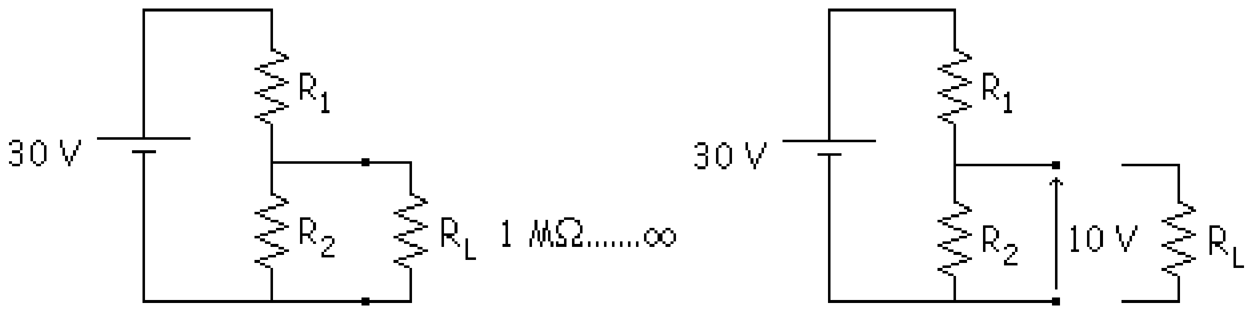
Ahora se dibuja el circuito Norton.



La corriente Norton es igual a la corriente con la carga en cortocircuito (5 mA) y la resistencia Norton es igual a la resistencia Thévenin (3 kΩ).

Problema 1.3

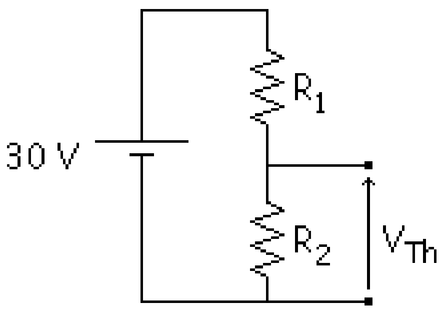
Diseñar un divisor de tensión para el circuito de la figura que genere una tensión fija de 10 V para todas las resistencias de carga mayores que 1 MW.



Solución:

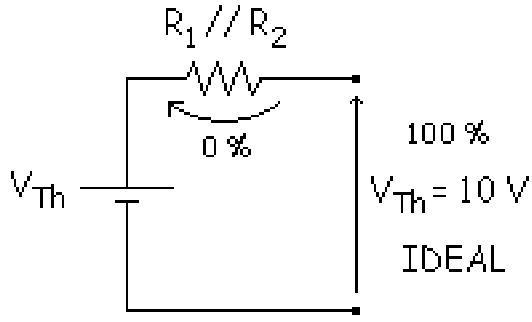
Se estudian los casos extremos para determinar los valores de las resistencias R1 y R2.

Caso extremo $R_L = \infty$

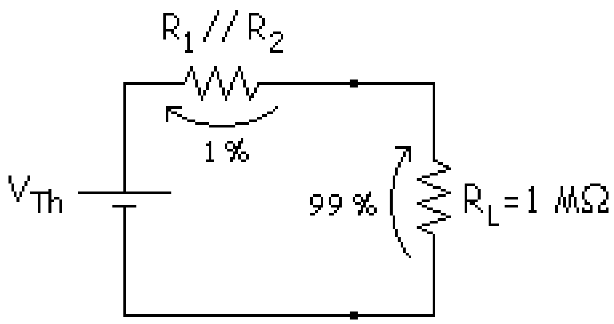


$$V_{Th} = \frac{30}{R_1 + R_2} \cdot R_2$$

$$R_{Th} = R_1 // R_2$$



Caso extremo $R_L = 1\ M\Omega$

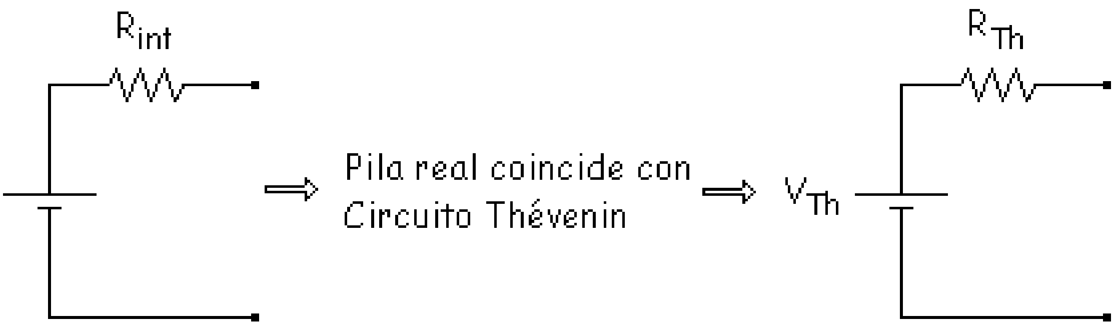


$$R_1 // R_2 = 10\ k\Omega \quad \left\{ \begin{array}{l} \dfrac{R_1 \cdot R_2}{R_1 + R_2} = 10 \\ \\ 10 = \dfrac{30}{R_1 + R_2} \cdot R_2 \end{array} \right\} \quad \begin{array}{l} R_1 = 30\ k\Omega \\ \\ R_2 = 15\ k\Omega \end{array}$$

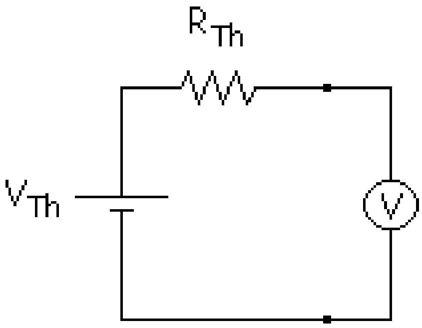
Problema 1.4

Sólo con una pila D, un polímetro y una caja con varias resistencias, describa un método mediante el cual, empleando una resistencia, halle la resistencia Thévenin de la pila.

Solución:

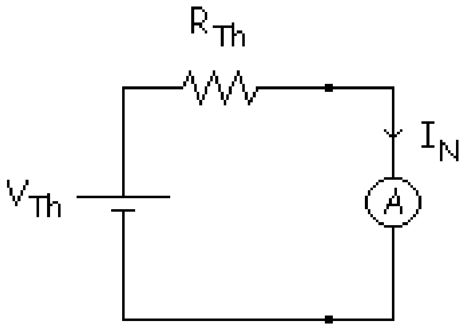


Voltímetro



$V_{Th} =$

Amperímetro



$I_N =$

Con estos 2 valores obtenemos el valor de la resistencia Thévenin.

$$V_{Th} = I_N \cdot R_{Th} \Longrightarrow R_{Th} = \frac{V_{Th}}{I_N}$$

Esta fórmula se suele utilizar para calcular Zi, Zo y Z vista desde dos puntos. Es una fórmula muy importante.

anterior/**principal**/**siguiente**

SEMICONDUCTORES

Antes de ver el funcionamiento de Diodos, Transistores y circuitos integrados, estudiaremos los materiales Semiconductores. Estos, que no son ni conductores ni aislantes, tienen electrones libres, pero lo que les caracteriza especialmente son los huecos.

En este tema, veremos los conceptos y propiedades más importantes de los Semiconductores.

Los objetivos de este tema son:

- Conocer las características de los semiconductores y conductores a nivel atómico.
- Ser capaz de describir la estructura de un cristal de Silicio.
- Saber cuales son y como se comportan los dos tipos de portadores y sus impurezas.
- Ser capaz de explicar las condiciones que se dan en la unión pn sin polarizar, polarizada en directa y polarizada en inversa.
- Conocer los dos tipos de corrientes de ruptura provocados por la aplicación sobre un diodo de gran voltaje en inversa.

anterior/**principal**/**siguiente**

Problemas

SEMICONDUCTORES

Antes de ver el funcionamiento de Diodos, Transistores y circuitos integrados, estudiaremos los materiales Semiconductores. Estos, que no son ni conductores ni aislantes, tienen electrones libres, pero lo que les caracteriza especialmente son los huecos.

En este tema, veremos los conceptos y propiedades más importantes de los Semiconductores.

Los objetivos de este tema son:

- Conocer las características de los semiconductores y conductores a nivel atómico.
- Ser capaz de describir la estructura de un cristal de Silicio.
- Saber cuales son y como se comportan los dos tipos de portadores y sus impurezas.
- Ser capaz de explicar las condiciones que se dan en la unión pn sin polarizar, polarizada en directa y polarizada en inversa.
- Conocer los dos tipos de corrientes de ruptura provocados por la aplicación sobre un diodo de gran voltaje en inversa.

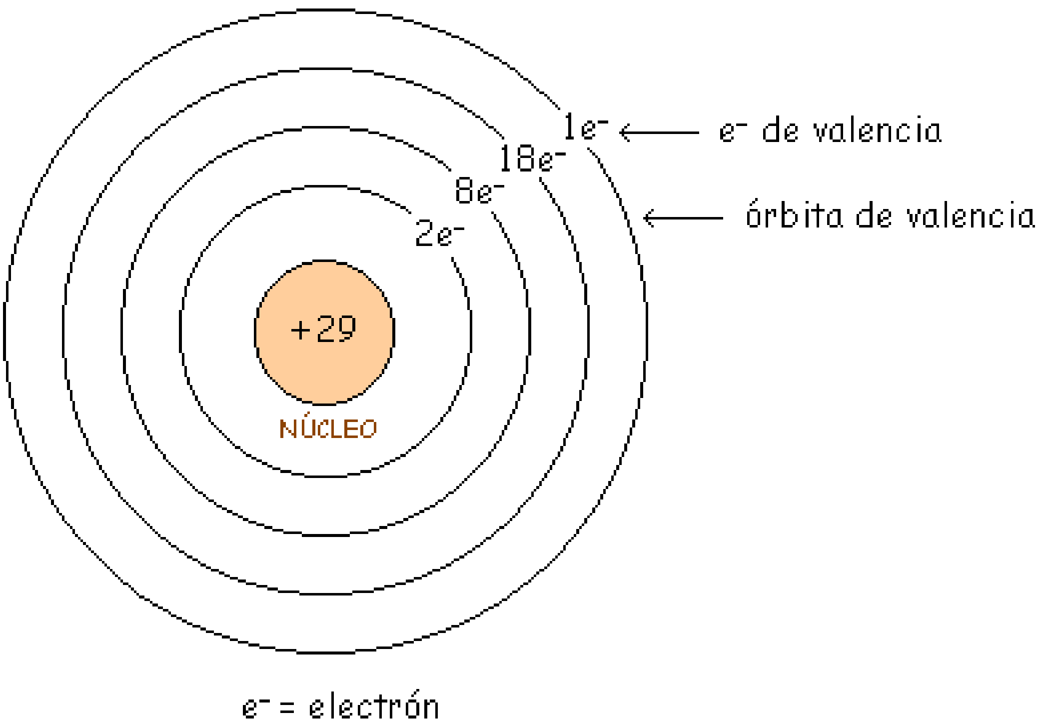
anterior/principal/siguiente

Conductores

Un conductor es un material que, en mayor o menor medida, conduce el calor y la electricidad. Son buenos conductores los metales y malos, el vidrio, la madera, la lana y el aire.

NOTA: Definimos la unidad de carga +1 como $+1{,}6\cdot 10^{-19}$ culombios. Así un electrón tiene una carga -1 equivalente a $-1{,}6\cdot 10^{-19}$ culombios.

El conductor más utilizado y el que ahora analizaremos es el Cobre (valencia 1), que es un buen conductor. Su estructura atómica la vemos en la siguiente figura.

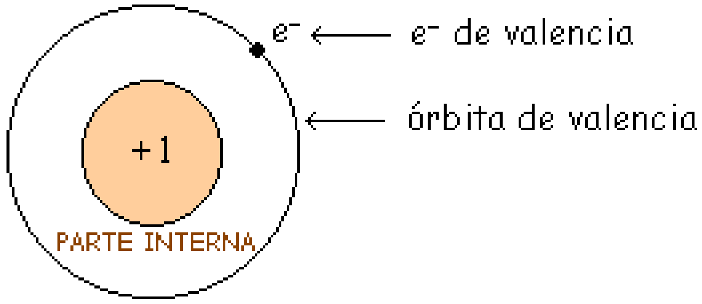


Su número atómico es 29. Esto significa que en el núcleo hay 29 protones (cargas positivas) y girando alrededor de él hay 29 electrones girando en diferentes órbitas.

En cada órbita caben $2n^2$ siendo n un número entero n = 1, 2, 3, ... Así en la primera órbita (n = 1) caben 212 = 2 electrones. En la segunda órbita $2\cdot 2^2 = 8$ electrones. En la tercera órbita $2\cdot 3^2 = 18$ electrones. Y la cuarta órbita solo tiene 1 electrón aunque en ella caben $2\cdot 4^2 = 32$ electrones.

Lo que interesa en electrónica es la órbita exterior, que es la que determina las propiedades del átomo. Como hay + 29 y - 28, queda con + 1.

Por ello vamos a agrupar el núcleo y las órbitas internas, y le llamaremos parte interna. En el átomo de cobre la parte interna es el núcleo (+ 29) y las tres primeras órbitas (- 28), con lo que nos queda la parte interna con una carga neta de +1.



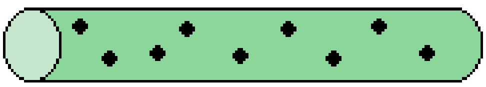
Como el electrón de valencia es atraído muy débilmente por la parte interna, una fuerza externa puede liberarlo fácilmente, por eso es un buen Conductor. Nos referiremos a ese electrón de valencia, como **electrón libre**.

Lo que define a un buen conductor es el hecho de tener un solo electrón en la órbita de valencia (valencia 1).

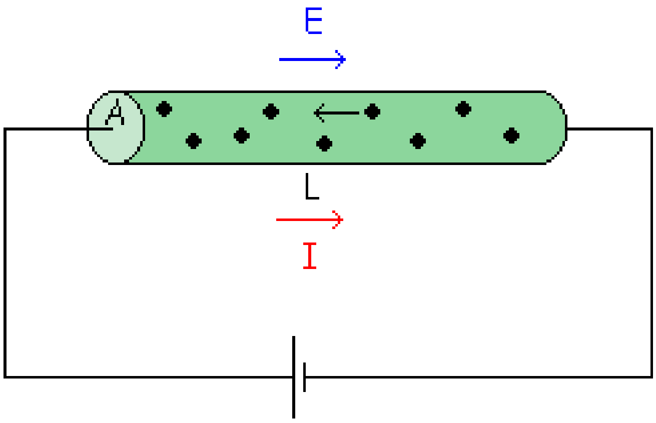
Así, tenemos que:

- A 0 ºK (-273 ºC) un metal no conduce.
- A Temperatura ambiente 300 ºK ya hay electrones libres debidos a la energía térmica.

- Si tenemos un campo eléctrico aplicado los electrones libres se mueven en todas direcciones. Como el movimiento es al azar, es posible que muchos electrones pasen por unidad de área en una determinada dirección y a la vez en la dirección opuesta. Por lo tanto la corriente media es cero.



- Veamos ahora como cambia la situación, si se aplica al metal un campo eléctrico.



Los electrones libres se mueven ahora en una dirección concreta. Y por lo tanto ya hay carga (en culombios) que cruza la sección del metal en un segundo, o sea ya existe una corriente.

Como ya conocemos, el electrón tiene una carga negativa (-1,619-19 culombios) y por tanto el convenio tomado para definir la corriente (contrario al movimiento de las cargas negativas) nos indica que la corriente toma el sentido indicado en la figura.

El electrón se mueve dentro de la red cristalina del metal con una velocidad media.

$$v = \mu \cdot E$$

v = velocidad media del electrón
μ = movilidad del electrón libre en el metal (mayor o menor facilidad para moverse dentro de la red cristalina)
E = intensidad del campo eléctrico

La resistencia que opone la barra de metal al paso de la corriente la podemos calcular de la siguiente forma:

$$\sigma = n \cdot \mu \cdot e$$

$$\rho = \frac{1}{\sigma}$$

$$R = \rho \cdot \frac{L}{A}$$

n = nº de electrones libres por m^3
e = carga del electrón $(-1{,}6 \times 10^{-19}$ Cul)
σ = conductividad del metal
ρ = resistividad del metal
R = resistencia del metal
L = longitud de la barra
A = sección de la barra

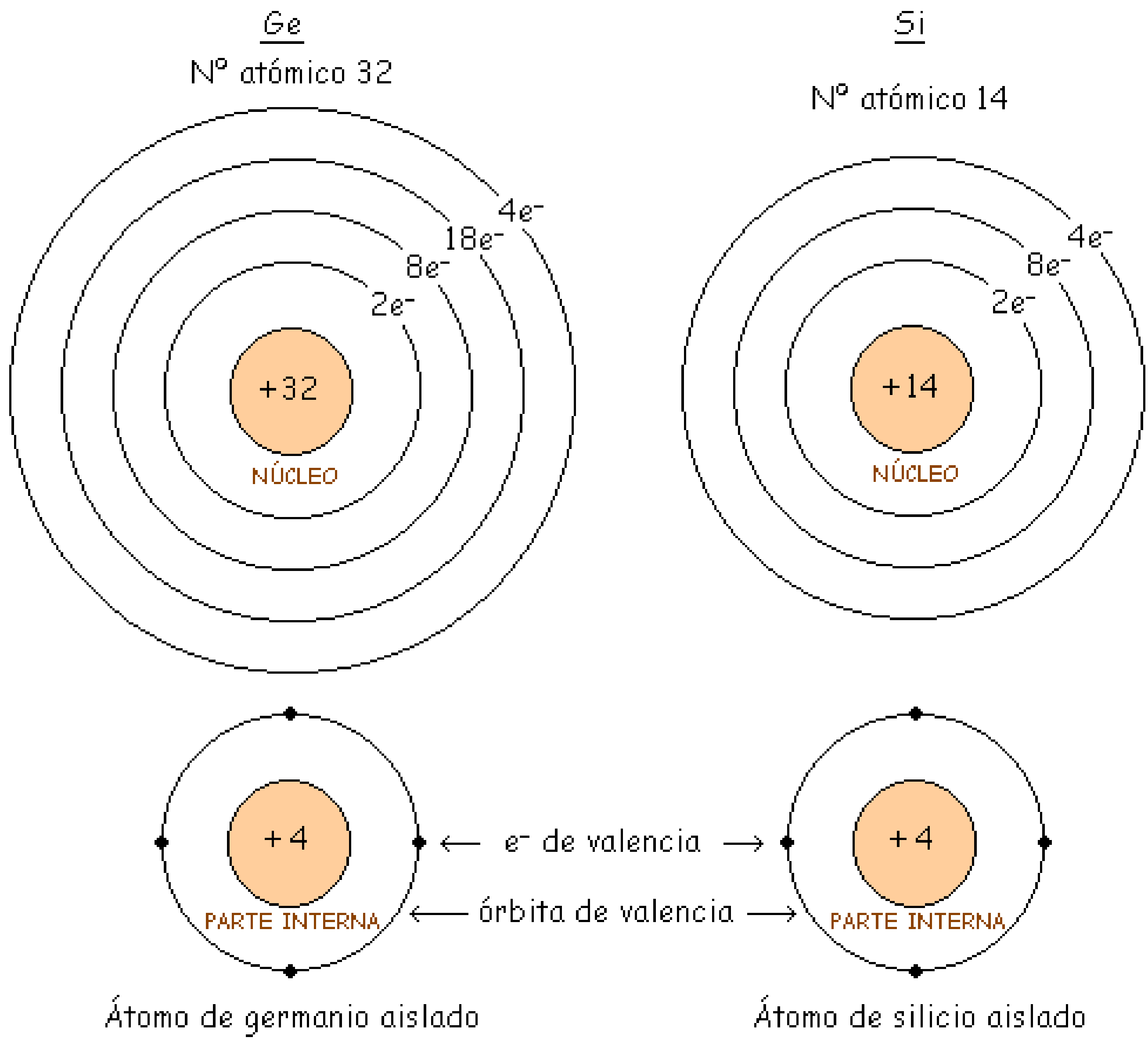
anterior/principal/siguiente

Semiconductores

Son elementos, como el germanio y el silicio, que a bajas temperaturas son aislantes. Pero a medida que se eleva la temperatura o bien por la adicción de determinadas impurezas resulta posible su conducción. Su importancia en electrónica es inmensa en la fabricación de transistores, circuitos integrados, etc...

Los semiconductores tienen valencia 4, esto es 4 electrones en órbita exterior ó de valencia. Los conductores tienen 1 electrón de valencia, los semiconductores 4 y los aislantes 8 electrones de valencia.

Los 2 semiconductores que veremos serán el Silicio y el Germanio:



Como vemos los semiconductores se caracterizan por tener una parte interna con carga + 4 y 4 electrones de valencia.

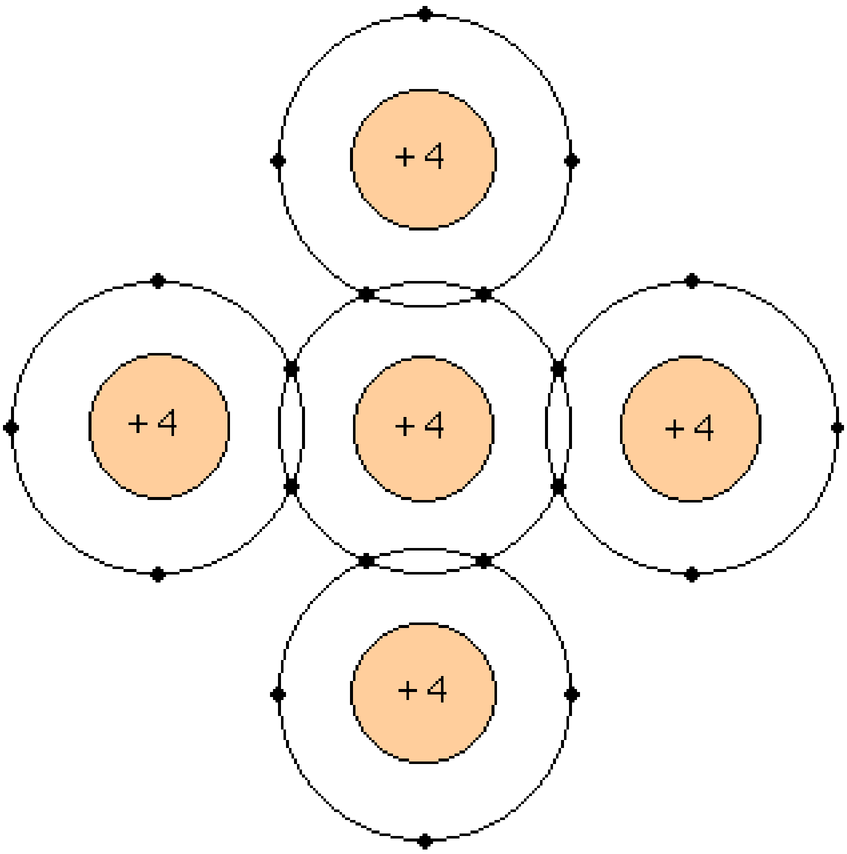
anterior/principal/siguiente

Cristales de silicio

[Simulación]

Al combinarse los átomos de Silicio para formar un sólido, lo hacen formando una estructura ordenada llamada cristal. Esto se debe a los "Enlaces Covalentes", que son las uniones entre átomos que se hacen compartiendo electrones adyacentes de tal forma que se crea un equilibrio de fuerzas que mantiene unidos los átomos de Silicio.

Vamos a representar un cristal de silicio de la siguiente forma:



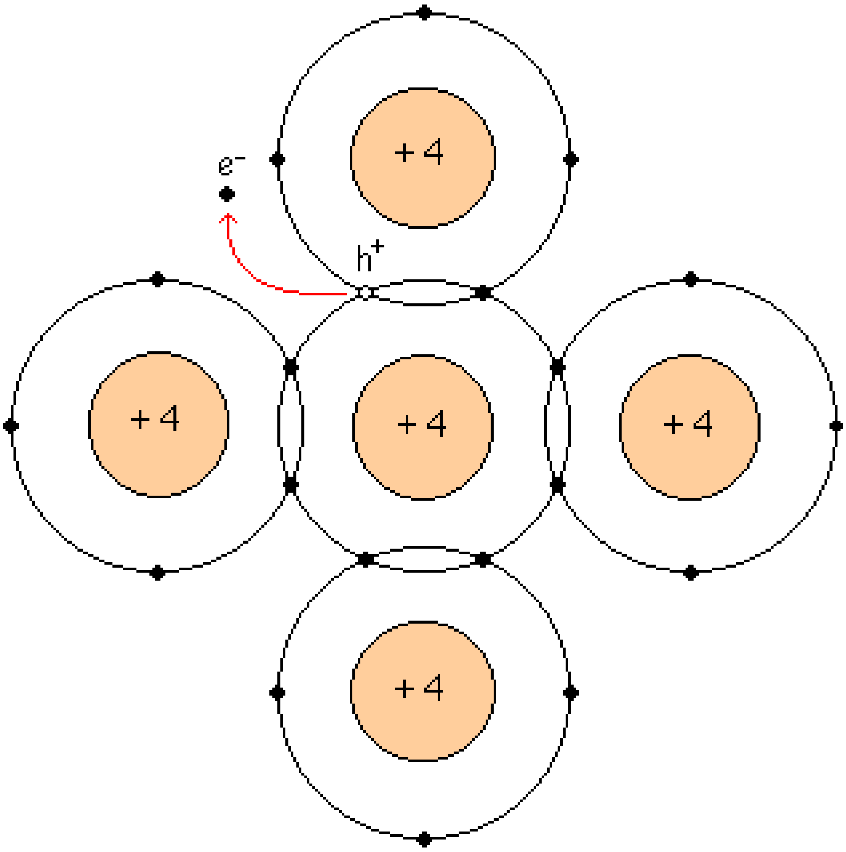
Cada átomo de silicio comparte sus 4 electrones de valencia con los átomos vecinos, de tal manera que tiene 8 electrones en la órbita de valencia, como se ve en la figura.

La fuerza del enlace covalente es tan grande porque son 8 los electrones que quedan (aunque sean compartidos) con cada átomo, gracias a esta característica los enlaces covalentes son de una gran solidez.

Los 8 electrones de valencia se llaman electrones ligados por estar fuertemente unidos en los átomos.

electrón ⇒ electrón libre
hueco ⇒ electrón ligado

El aumento de la temperatura hace que los átomos en un cristal de silicio vibren dentro de él, a mayor temperatura mayor será la vibración. Con lo que un electrón se puede liberar de su órbita, lo que deja un hueco, que a su vez atraerá otro electrón, etc...

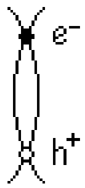


h^+ = hueco
e^- = electrón

A 0 ºK, todos los electrones son ligados. A 300 ºK o más, aparecen electrones libres.

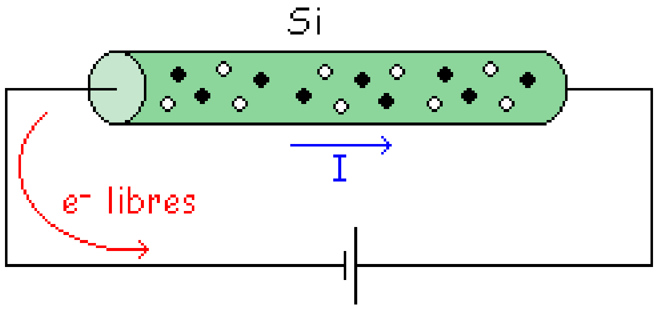
Tª↑ ⇒ más electrones libres

Esta unión de un electrón libre y un hueco se llama "recombinación", y el tiempo entre la creación y desaparición de un electrón libre se denomina "tiempo de vida".



Enlace covalente roto: Es cuando tenemos un hueco, esto es una generación de pares electrón libre-hueco.

Según un convenio ampliamente aceptado tomaremos la dirección de la corriente como contraria a la dirección de los electrones libres.



Simulación

En este applet podemos ver mediante una animación el comportamiento de los electrones en un cristal de silicio.

Los electrones libres (electrones) se mueven hacia la izquierda ocupando el lugar del hueco.
Carga del electrón libre = -1.6×10^{-19} Culombios.

Los electrones ligados (huecos) se mueven hacia la derecha.
Carga de electrón ligado = $+1.6\times10^{-19}$ Culombios.

Semiconductores: Conducen los electrones (electrones libres) y los huecos (electrones ligados).

Conductores: Conducen los electrones libres.

Resumiendo: Dentro de un cristal en todo momento ocurre esto:

- Por la energía térmica se están creando electrones libres y huecos.
- Se recombinan otros electrones libres y huecos.
- Quedan algunos electrones libres y huecos en un estado intermedio, en el que han sido creados y todavía no se han recombinado.

anterior/principal/siguiente

Semiconductores intrínsecos

Simulación

Es un semiconductor puro. A temperatura ambiente se comporta como un aislante porque solo tiene unos pocos electrones libres y huecos debidos a la energía térmica.

En un semiconductor intrínseco también hay flujos de electrones y huecos, aunque la corriente total resultante sea cero. Esto se debe a que por acción de la energía térmica se producen los electrones libres y los huecos por pares, por lo tanto hay tantos electrones libres como huecos con lo que la corriente total es cero.

La tensión aplicada en la figura forzará a los electrones libres a circular hacia la derecha (del terminal negativo de la pila al positivo) y a los huecos hacia la izquierda.

Simulación

En este applet podemos ver mediante una animación en que dirección se mueven los electrones y los huecos en un semiconductor intrínseco.

Cuando los electrones libres llegan la extremo derecho del cristal, entran al conductor externo (normalmente un hilo de cobre) y circulan hacia el terminal positivo de la batería. Por otro lado, los electrones libres en el terminal negativo de la batería fluirían hacia el extremos izquierdo del cristal. Así entran en el cristal y se recombinan con los huecos que llegan al extremo izquierdo del cristal. Se produce un flujo estable de electrones libres y huecos dentro del semiconductor.

anterior/principal/siguiente

Dopado de un semiconductor

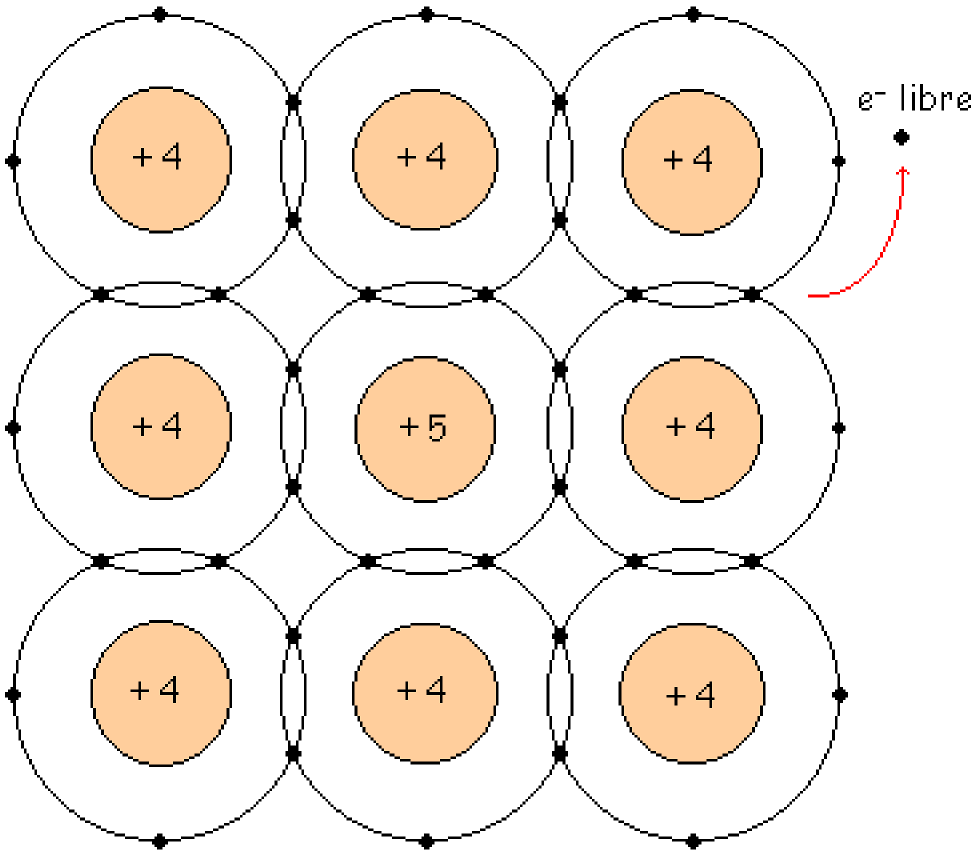
Caso 1

Caso 2

Para aumentar la conductividad (que sea más conductor) de un SC (Semiconductor), se le suele dopar o añadir átomos de impurezas a un SC intrínseco, un SC dopado es un SC extrínseco.

Caso 1

Impurezas de valencia 5 (Arsénico, Antimonio, Fósforo). Tenemos un cristal de Silicio dopado con átomos de valencia 5.



Los átomo de valencia 5 tienen un electrón de más, así con una temperatura no muy elevada (a temperatura ambiente por ejemplo), el 5º electrón se hace electrón libre. Esto es, como solo se pueden tener 8 electrones en la órbita de valencia, el átomo pentavalente suelta un electrón que será libre.

Siguen dándose las reacciones anteriores. Si metemos 1000 átomos de impurezas tendremos 1000 electrones más los que se hagan libres por generación térmica (muy pocos).

A estas impurezas se les llama "Impurezas Donadoras". El número de electrones libres se llama **n** (electrones libres/m^3).

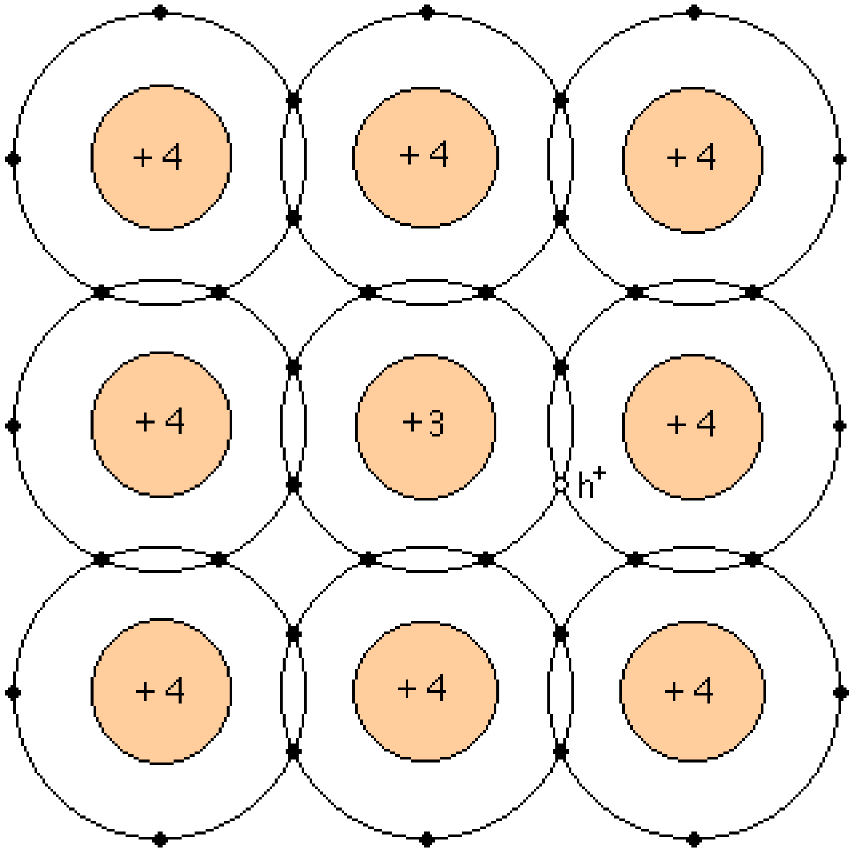
$$N_D = \frac{\text{átomos de impurezas Donadoras}}{m^3}$$

$$n = N_D + \text{electrones de generación térmica}$$

$$\boxed{n \simeq N_D}$$

Caso 2

Impurezas de valencia 3 (Aluminio, Boro, Galio). Tenemos un cristal de Silicio dopado con átomos de valencia 3.



Los átomo de valencia 3 tienen un electrón de menos, entonces como nos falta un electrón tenemos un hueco. Esto es, ese átomo trivalente tiene 7 electrones en la orbita de valencia. Al átomo de valencia 3 se le llama "átomo trivalente" o "Aceptor".

A estas impurezas se les llama "Impurezas Aceptoras". Hay tantos huecos como impurezas de valencia 3 y sigue habiendo huecos de generación térmica (muy pocos). El número de huecos se llama p (huecos/m^3).

$$N_A = \frac{\text{átomos de impurezas Aceptoras}}{m^3}$$

$$p = N_A + \text{huecos de generación térmica}$$

$$\boxed{p \simeq N_A}$$

anterior/**principal**/**siguiente**

Semiconductores extrínsecos

Semiconductor tipo n

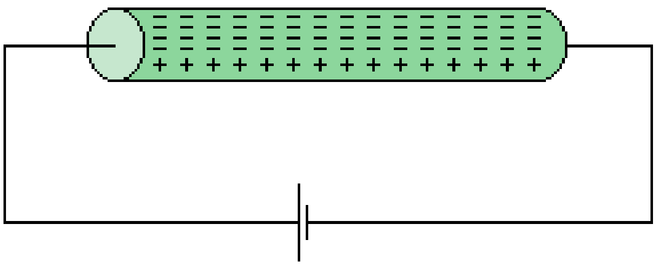
Semiconductor tipo p

Son los semiconductores que están dopados, esto es que tienen impurezas. Hay 2 tipos dependiendo de que tipo de impurezas tengan:

Semiconductor tipo n

Es el que está impurificado con impurezas "Donadoras", que son impurezas pentavalentes. Como los electrones superan a los huecos en un semiconductor tipo n, reciben el nombre de "portadores mayoritarios", mientras que a los huecos se les denomina "portadores minoritarios".

Al aplicar una tensión al semiconductor de la figura, los electrones libres dentro del semiconductor se mueven hacia la izquierda y los huecos lo hacen hacia la derecha. Cuando un hueco llega al extremo derecho del cristal, uno de los electrones del circuito externo entra al semiconductor y se recombina con el hueco.

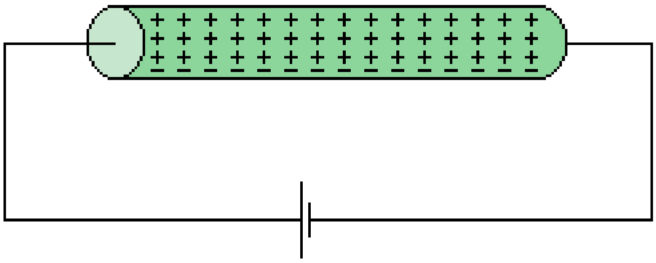


Los electrones libres de la figura circulan hacia el extremo izquierdo del cristal, donde entran al conductor y fluyen hacia el positivo de la batería.

Semiconductor tipo p

Es el que está impurificado con impurezas "Aceptoras", que son impurezas trivalentes. Como el número de huecos supera el número de electrones libres, los huecos son los portadores mayoritarios y los electrones libres son los minoritarios.

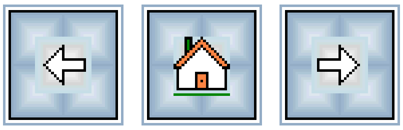
Al aplicarse una tensión, los electrones libres se mueven hacia la izquierda y los huecos lo hacen hacia la derecha. En la figura, los huecos que llegan al extremo derecho del cristal se recombinan con los electrones libres del circuito externo.



En el circuito hay también un flujo de portadores minoritarios. Los electrones libres dentro del semiconductor circulan de derecha a izquierda. Como hay muy pocos portadores minoritarios, su efecto es casi despreciable en este circuito.

anterior/principal/siguiente

El diodo no polarizado

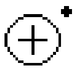


Zona de deplexión

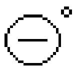
Barrera de potencial

Los semiconductores tipo p y tipo n separados no tienen mucha utilidad, pero si un cristal se dopa de tal forma que una mitad sea tipo n y la otra mitad de tipo p, esa unión pn tiene unas propiedades muy útiles y entre otras cosas forman los "Diodos".

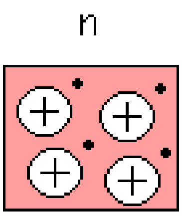
El átomo pentavalente en un cristal de silicio (Si) produce un electrón libre y se puede representar como un signo "+" encerrado en un circulo y con un punto relleno (que sería el electrón) al lado.



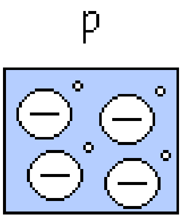
El átomo trivalente sería un signo "-" encerrado en un circulo y con un punto sin rellenar al lado (que simbolizaría un hueco).



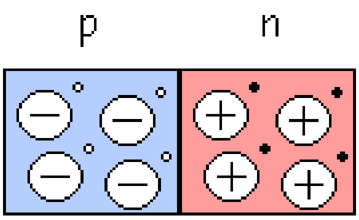
Entonces la representación de un SC tipo n sería:



Y la de un SC tipo p:



La unión de las regiones p y n será:

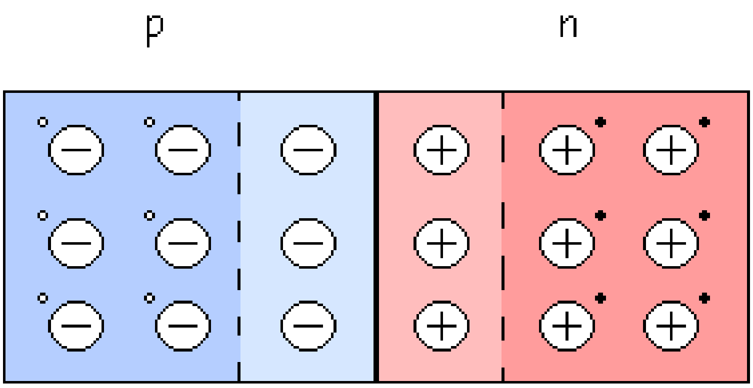


Al juntar las regiones tipo p y tipo n se crea un "Diodo de unión" o "Unión pn".

Zona de deplexión

Al haber una repulsión mutua, los electrones libres en el lado n se dispersan en cualquier dirección. Algunos electrones libres se difunden y atraviesan la unión, cuando un electrón libre entra en la región p se convierte en un portador minoritario y el electrón cae en un hueco, el hueco desaparece y el electrón libre se convierte en electrón de valencia. Cuando un electrón se difunde a través de la unión crea un par de iones, en el lado n con carga positiva y en el p con carga negativa.

Las parejas de iones positivo y negativo se llaman dipolos, al aumentar los dipolos la región cerca de la unión se vacía de portadores y se crea la llamada "Zona de deplexión".



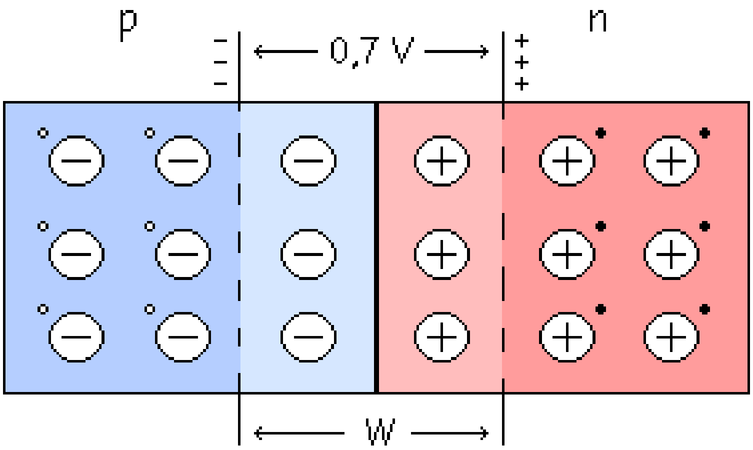
Barrera de potencial

Los dipolos tienen un campo eléctrico entre los iones positivo y negativo, y al entrar los electrones libres en la zona de deplexión, el campo eléctrico trata de devolverlos a la zona n. La intensidad del campo eléctrico aumenta con cada electrón que cruza hasta llegar al equilibrio.

El campo eléctrico entre los iones es equivalente a una diferencia de potencial llamada "Barrera de Potencial" que a 25 ºC vale:

- 0.3 V para diodos de Ge.
- 0.7 V para diodos de Si.

Polarizar: Poner una pila.
No polarizado: No tiene pila, circuito abierto o en vacío.
z.c.e.: Zona de Carga Espacial o zona de deplexión (W).

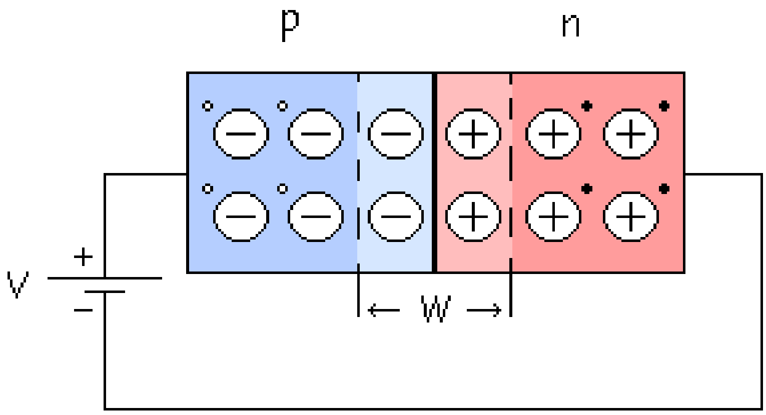


anterior/**principal**/**siguiente**

Polarización directa

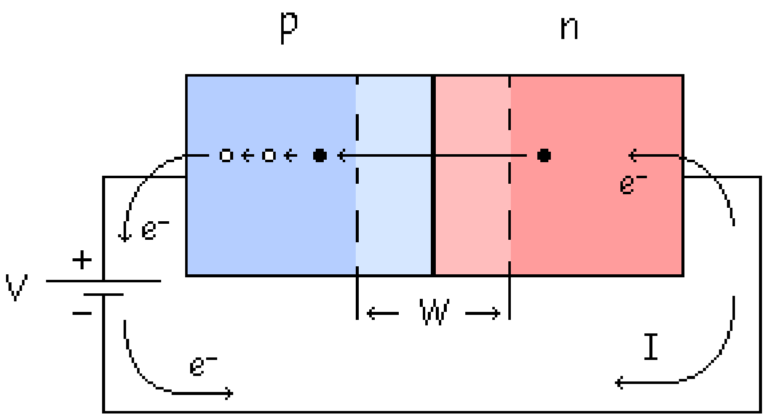
Si el terminal positivo de la fuente está conectado al material tipo **p** y el terminal negativo de la fuente está conectado al material tipo n, diremos que estamos en "Polarización Directa".

La conexión en polarización directa tendría esta forma:



En este caso tenemos una corriente que circula con facilidad, debido a que la fuente obliga a que los electrones libres y huecos fluyan hacia la unión. Al moverse los electrones libres hacia la unión, se crean iones positivos en el extremo derecho de la unión que atraerán a los electrones hacia el cristal desde el circuito externo.

Así los electrones libres pueden abandonar el terminal negativo de la fuente y fluir hacia el extremo derecho del cristal. El sentido de la corriente lo tomaremos siempre contrario al del electrón.



Lo que le sucede al electrón: Tras abandonar el terminal negativo de la fuente entra por el extremo derecho del cristal. Se desplaza a través de la zona n como electrón libre.

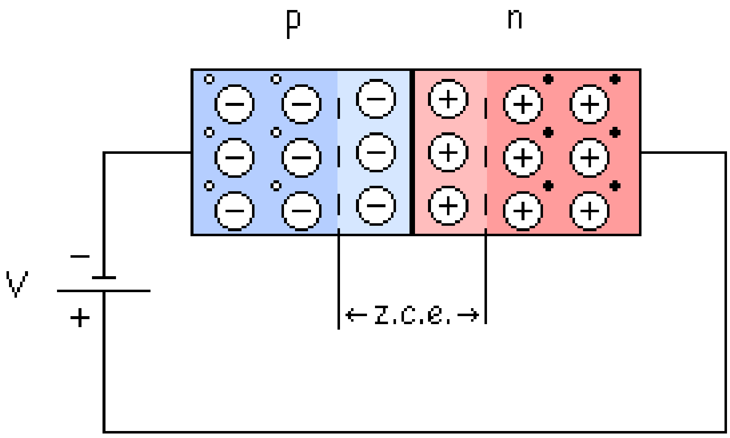
En la unión se recombina con un hueco y se convierte en electrón de valencia. Se desplaza a través de la zona p como electrón de valencia. Tras abandonar el extremo izquierdo del cristal fluye al terminal positivo de la fuente.

anterior/principal/siguiente

Polarización inversa

Se invierte la polaridad de la fuente de continua, el diodo se polariza en inversa, el terminal negativo de la batería conectado al lado p y el positivo al n, esta conexión se denomina "Polarización Inversa".

En la siguiente figura se muestra una conexión en inversa:

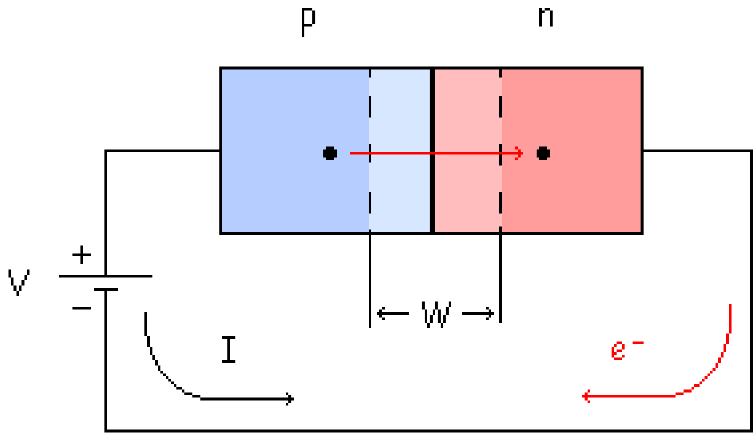


z.c.e. = W

El terminal negativo de la batería atrae a los huecos y el terminal positivo atrae a los electrones libres, así los huecos y los electrones libres se alejan de la unión y la **z.c.e.** se ensancha.

A mayor anchura de la z.c.e. mayor diferencia de potencial, la zona de deplexión deja de aumentar cuando su diferencia de potencial es igual a la tensión inversa aplicada (V), entonces los electrones y huecos dejan de alejarse de la unión.

A mayor la tensión inversa aplicada mayor será la z.c.e.



$$V \uparrow \Longrightarrow w \uparrow$$

Existe una pequeña corriente en polarización inversa, porque la energía térmica crea continuamente pares electrón-hueco, lo que hace que halla pequeñas concentraciones de portadores minoritarios a ambos lados, la mayor parte se recombina con los mayoritarios pero los que están en la z.c.e. pueden vivir lo suficiente para cruzar la unión y tenemos así una pequeña corriente.

La zona de deplexión empuja a los electrones hacia la derecha y el hueco a la izquierda, se crea así una la "Corriente Inversa de Saturación"(I_S) que depende de la temperatura.

$$T^{a} \uparrow \Longrightarrow I_S \uparrow$$

Además hay otra corriente "Corriente Superficial de Fugas" causada por las impurezas del cristal y las imperfecciones en su estructura interna. Esta corriente depende de la tensión de la pila (V ó V_P).

$$V \uparrow \Longrightarrow I_f \uparrow$$

Entonces la corriente en inversa (I ó I_R) será la suma de esas dos corrientes:

$$I = I_S + I_f$$

anterior/**principal**/**siguiente**

Ruptura

Efecto Avalancha

Efecto Zener

Los diodos admiten unos valores máximos en las tensiones que se les aplican, existe un límite para la tensión máxima en inversa con que se puede polarizar un diodo sin correr el riesgo de destruirlo.

Veamos un ejemplo:

	V_p	I_S	I_f	I_R
	−5 V	−10 nA	−10 nA	−10 nA
	−10 V	−10 nA	−20 nA	−30 nA
	−20 V	−10 nA	−40 nA	−50 nA
$V_{Ruptura}$ ⟹	−50 V	X	X	−100 mA

V_p = Tensión de la pila

I_R = Corriente inversa del diodo

A la tensión en la que la I_R aumenta de repente, se le llama "Tensión de Ruptura" ($V_{Ruptura}$). A partir de este valor I_R es muy grande y el diodo se estropea. En el diodo ha ocurrido el "Efecto Avalancha" o "Ruptura por Avalancha".

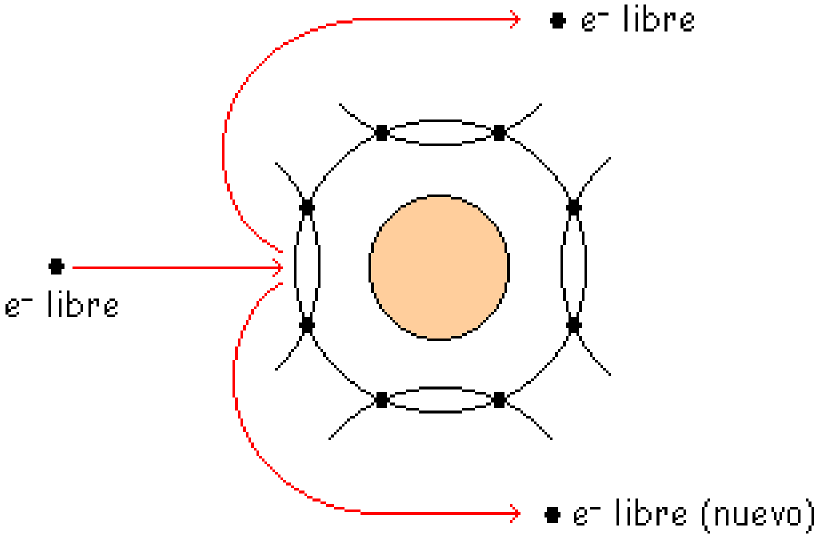
Efecto Avalancha = Ruptura por Avalancha = Multiplicación por Avalancha

Efecto Avalancha

Aumenta la tensión inversa y con ella la z.c.e..

$$V_{Inversa}\uparrow \Longrightarrow W\uparrow$$

Ocurre lo siguiente dentro del diodo:



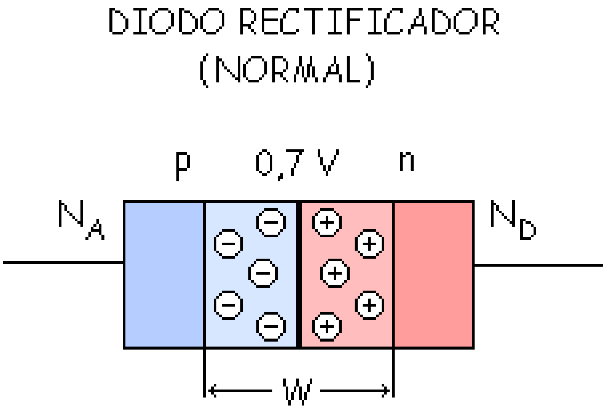
Justo en el límite antes de llegar a Ruptura, la pila va acelerando a los electrones. Y estos electrones pueden chocar con la red cristalina, con los enlaces covalentes. Choca el electrón y rebota, pero a $V_{Ruptura}$ la velocidad es muy grande y por ello la E_c es tan grande que al chocar cede energía al electrón ligado y lo convierte en libre. El electrón incidente sale con menos velocidad que antes del choque. O sea, de un electrón libre obtenemos dos electrones libres.

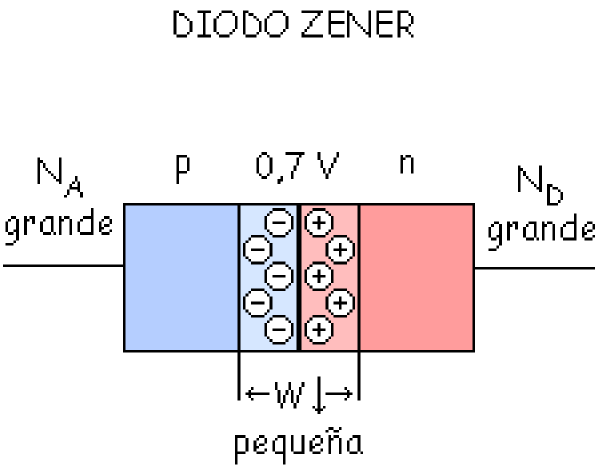
Estos 2 electrones se aceleran otra vez, pueden chocar contra otro electrón de un enlace covalente, ceden su energía... y se repite el proceso y se crea una Multiplicación por Avalancha.

Y ahora I_R ha aumentado muchísimo, tenemos una corriente negativa y muy grande (-100 mA). Con esta intensidad el diodo se estropea porque no está preparado para trabajar a esa I_R.

Efecto Zener

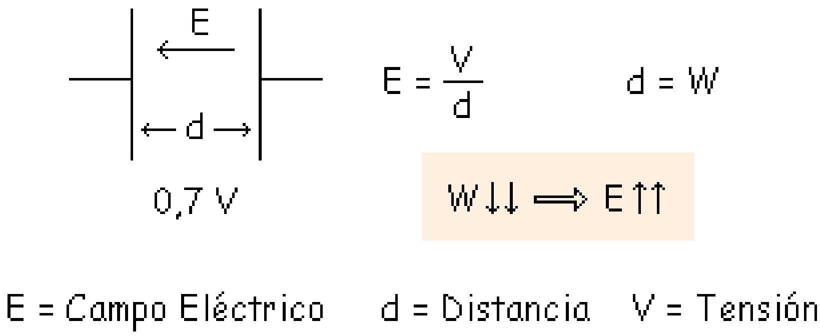
Este es otro efecto que puede estropear el diodo, y es muy parecido al anterior. Se suele dar en diodos muy impurificados, diodos con muchas impurezas.





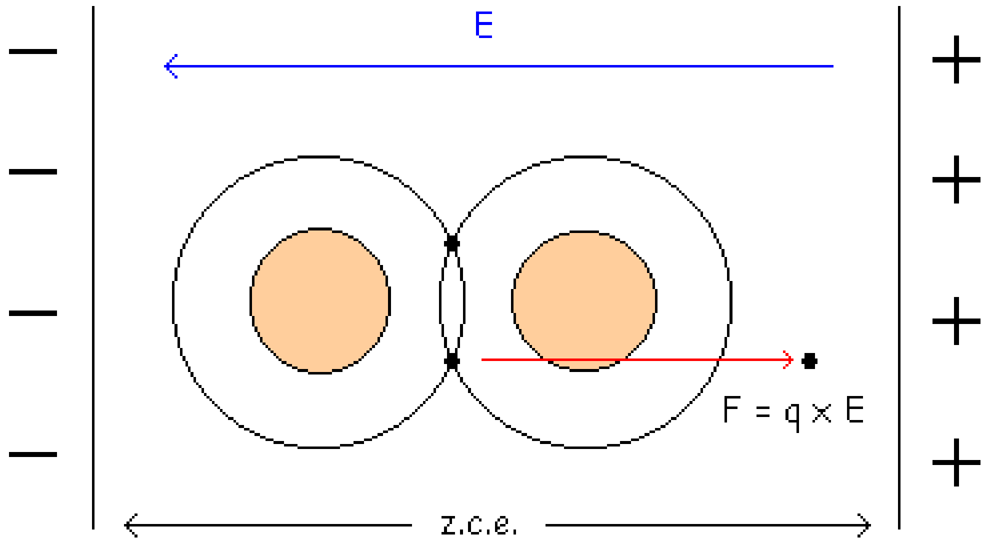
Al tener la z.c.e. muy pequeña y seguimos teniendo la misma tensión (0.7 V), tenemos muy juntos los átomos de impurezas teniendo así más carga en menos espacio.

En esta situación se crea un campo eléctrico muy intenso. Y el efecto es como la carga de un condensador.



Si se polariza en inversa se ensancha la z.c.e.

¿Qué ocurre en la z.c.e.?



A aumentado mucho E (Campo Eléctrico), por ejemplo para los 3 V llega a 300.000 V/cm y se da el "Efecto Zener": Ahora la F, fuerza debida al campo eléctrico, es capaz de arrancar el electrón y lo hace libre. Este campo eléctrico intenso arranca muchos electrones de esta forma dando lugar a una corriente grande que destruye el Diodo.

Veamos en que diodos se dan estos 2 efectos:

- Efecto Avalancha (Ruptura por avalancha)
  - **Diodo Rectificador** V_R = - 50 V (tensiones grandes).
  - **Diodo de Avalancha** V_R = - 6 V, - 7 V, - 8 V... A veces le llama Diodo Zener aunque no sea un Zener en si.
- Efecto Zener (Ruptura Zener)
  - **Diodo Zener** V_R = - 4 V, - 3 V, - 2 V... A veces puede ocurrir este efecto en otro tipo de diodos que no sean Zener, pero tienen que estar muy impurificados. Los Diodos Zener están especialmente preparados para no estropearse.

Entre - 4 V y - 6 V se pueden dar los 2 fenómenos a la vez (Avalancha y Zener).

anterior/principal/siguiente

Niveles y bandas de energía

Bandas de Energía en un Semiconductor Intrínseco

Bandas de Energía en un Semiconductor tipo n

Bandas de Energía en un Semiconductor tipo p

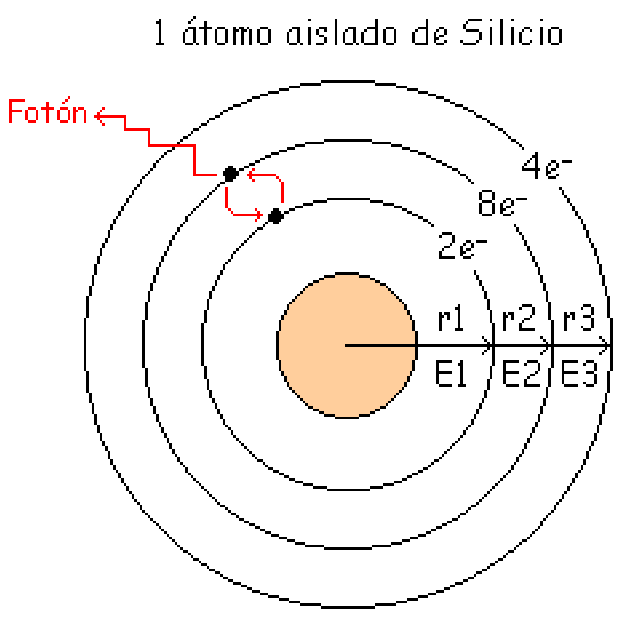
Las ideas y los conceptos vistos anteriormente los analizaremos ahora desde un punto de vista energético.

Hablar de Radios y de Energías es lo mismo. Cuanto mayor sea el radio mayor será también la energía.

Existen diversas maneras de darle energía a un electrón, por:

- Energía Térmica.
- Energía Luminosa (fotón E = h x f).
- Campo Eléctrico.
- etc...

Si se le da energía a un electrón para que pase de E1 a E2, este electrón puede pasar de una orbita a otra.



Ese electrón vuelve enseguida, al volver tiene que ceder o soltar la energía. Puede hacerlo de 2 formas:

- Al volver sale un fotón de luz:

$$E2 - E1 = h \times f$$

Una aplicación de esta característica se ve en los Diodos Led, que dependiendo de las energías tendrán diferentes colores, y también pueden soltar fotones invisibles a frecuencias en las que la vista no puede captarlas.

- También se suelta energía en forma de calor, energía térmica (calentamiento del diodo).

Las energías las representaremos gráficamente de esta manera:

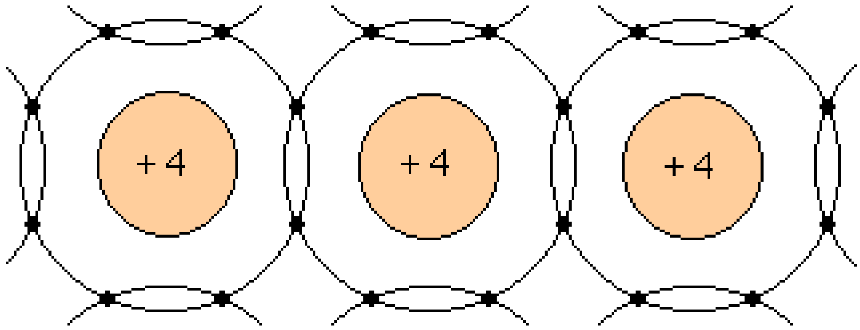
Hasta ahora hemos visto un átomo aislado, pero en un cristal tenemos que aplicar el "Principio de Exclusión de Pauli":

"*En un sistema electrónico no puede haber 2 electrones con los mismos números cuánticos*".

Esto es, que no puede haber 2 electrones con la misma energía.

Bandas de Energía en un Semiconductor Intrínseco

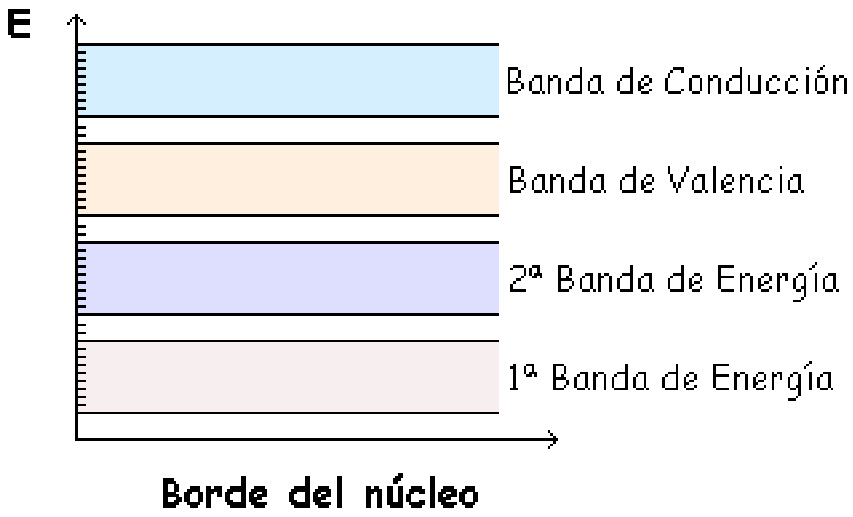
Anteriormente hemos visto que los semiconductores intrínsecos eran aquellos que no tenían impurezas, esto es, todos son átomos de Si.



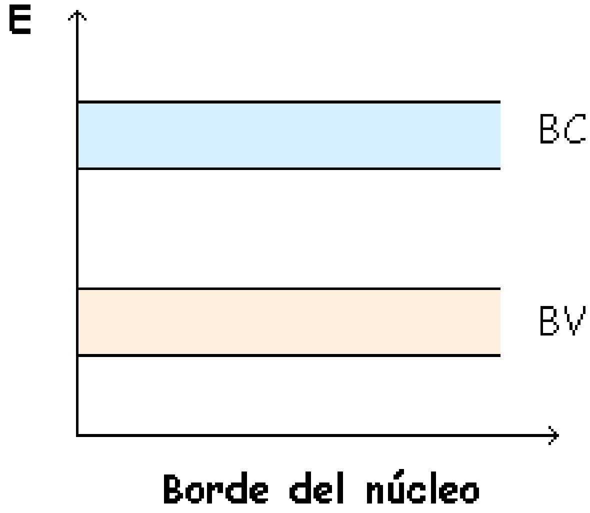
Al aplicar el principio de exclusión de Pauli el electrón de energía E1 de un átomo y el electrón de energía E1 del átomo vecino se han de separar en energía. Como hay una gran cantidad de átomos aparecen muchos niveles energéticos con una separación muy pequeña, formando la 1ª Banda de Energía.

Los electrones de energía E2 se separan en energía formando la 2ª Banda de Energía.

Y así sucesivamente con el resto de energías se van creando Bandas de Energía (grupos de niveles energéticos). El resultado es el siguiente:

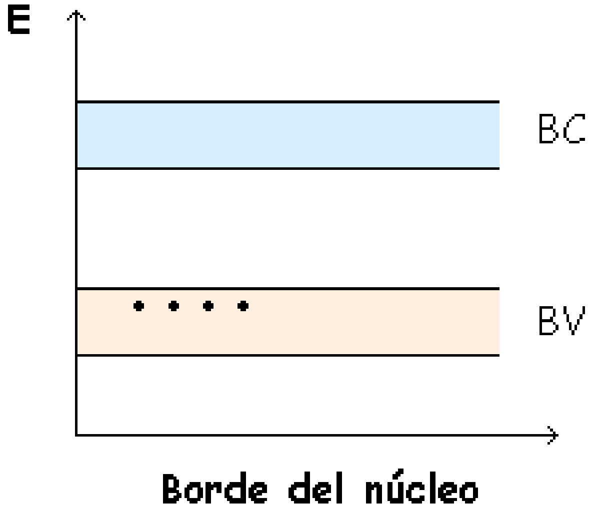


Como es difícil sacar un electrón de las bandas inferiores, no nos interesan las 2 bandas inferiores, no las tendremos en cuenta, así tendríamos:



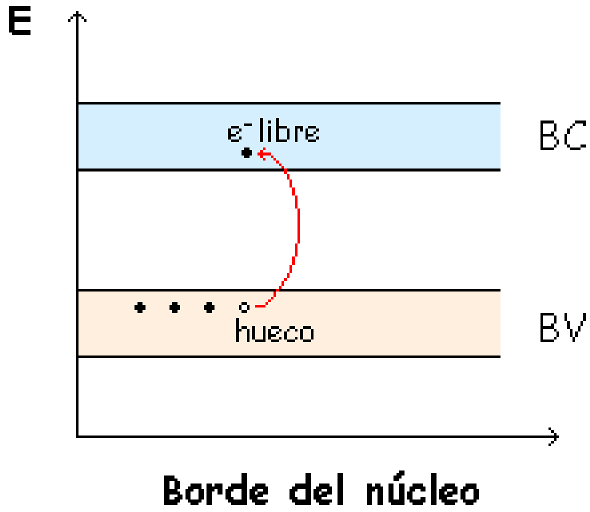
Estas 2 bandas son las creadas por los 4 electrones de la última órbita del átomo.

A 0 ºK los 4 electrones de cada átomo están en la Banda de Valencia (cada uno en un radio o energía permitido).



BC = Banda de Conducción
BV = Banda de Valencia

A 300 ºK (27 ºC, temperatura ambiente) o a mayor temperatura, algún electrón puede conseguir suficiente energía como para pasar a la Banda de Conducción, dejando así un hueco en la Banda de Valencia.



Recordar que a esto le llamábamos Generación Térmica de Pares electrón libre-hueco. Cuanto más aumente la temperatura, más electrones suben debido a la generación térmica.

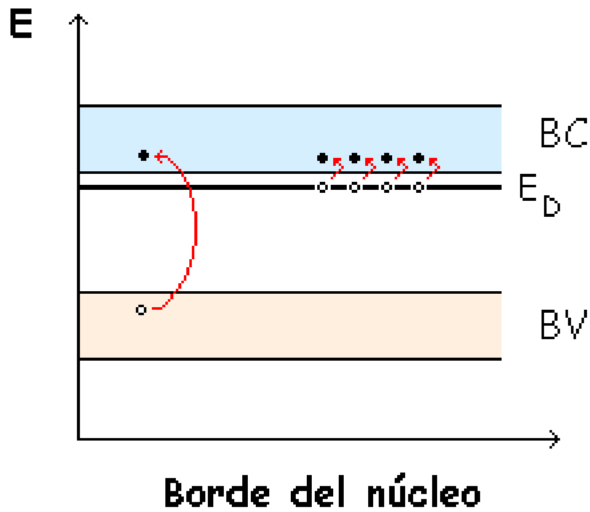
Por eso un semiconductor a 0 ºK no conduce y si aumenta la temperatura conduce más. Ahora veremos que es lo que ocurre con los semiconductores con impurezas.

Bandas de Energía en un Semiconductor tipo n

Tenemos muy pocos átomos de impurezas (+5) en comparación con los átomos normales de Silicio (+4).

Como se impurifica muy poco, los átomos de +5 están muy alejados y no se influyen entre si, pudiendo tener electrones de átomos diferentes la misma energía y por lo tanto están todos al mismo nivel. Esa energía que tienen se llama "Energía del átomo Donador" (E_D).

En cuanto se le de una pequeña energía los electrones suben a la BC y se convierten en libres.

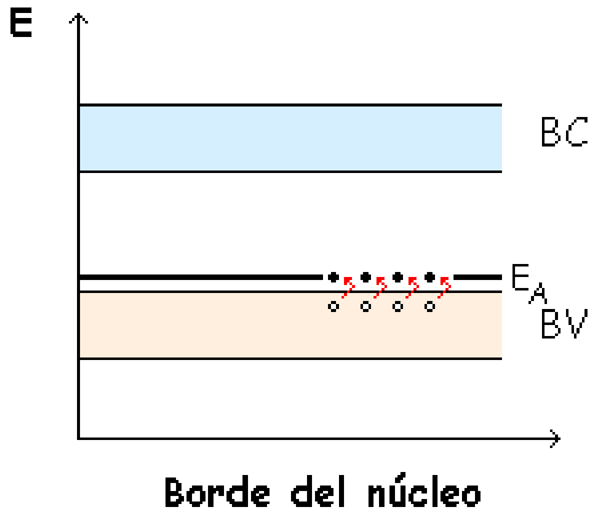


También se da la generación térmica (generación de pares hueco-electrón), pero lo que más ocurre es debido a las impurezas y muy poco por generación térmica, por lo que despreciaremos esta última.

Bandas de Energía en un Semiconductor tipo p

En este caso las impurezas son átomos de +3, y como en el caso anterior hay muy pocos y están muy alejados por lo que los electrones de átomos diferentes están al mismo nivel energético. Esa energía es la "Energía del átomo Aceptor" (E_A).

A 300 ºK o más, el electrón cercano a E_A sube desde la BV y deja un hueco en la BV mientras que la E_A se llena de electrones. Se sigue dando generación térmica también, pero como antes es despreciable.



anterior/principal/siguiente

La barrera de energía

Antes de la Difusión

Empieza la Difusión y la Recombinación

Equilibrio

Polarización Directa

Polarización Inversa

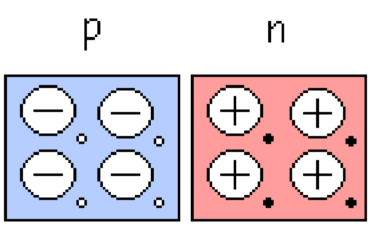
Para poder comprender como funcionan los dispositivos semiconductores, es necesario conocer el modo en que los niveles de energía controlan la acción de una unión pn.

Ahora se verá como se forma la barrera de potencial de 0.7 V en el diodo. Veremos 5 puntos:

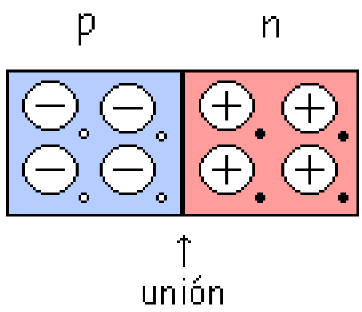
- Antes de la difusión
- Empieza la difusión y la recombinación
- Equilibrio
- Polarización Directa
- Polarización Inversa

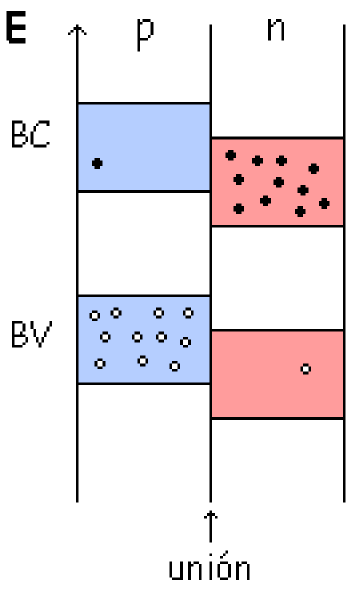
Antes de la Difusión

Zona p y n antes de unirse:



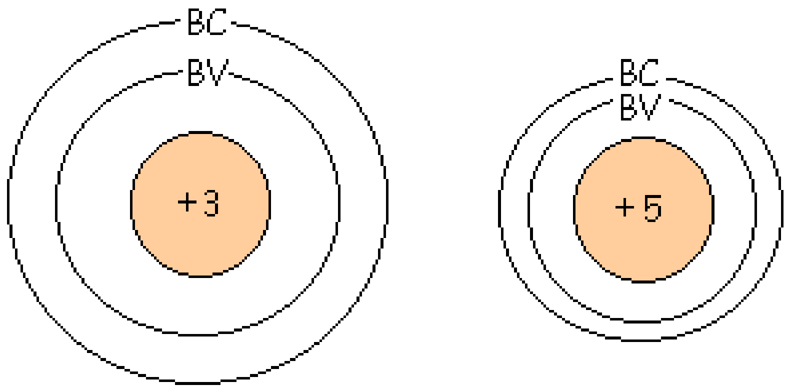
Instante inicial en que se juntan. Instante cero, todavía no ha habido difusión:





¿ Porqué están más altas una bandas en p que en n ?

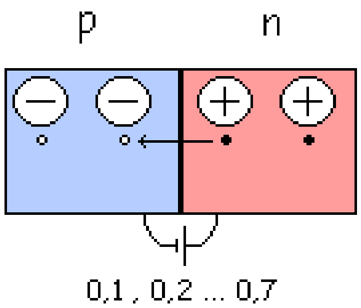
Las órbitas de la zona p son más pequeñas y por lo tanto los radios también son más pequeños. Como se ha dicho anteriormente, hablar de radios es equivalente a hablar de energías, entonces las energías también son más pequeñas.



Esto es porque +5 atrae más fuertemente que +3. A mayor carga atrae con más fuerza, disminuye así el radio, con lo que la energía es menor.

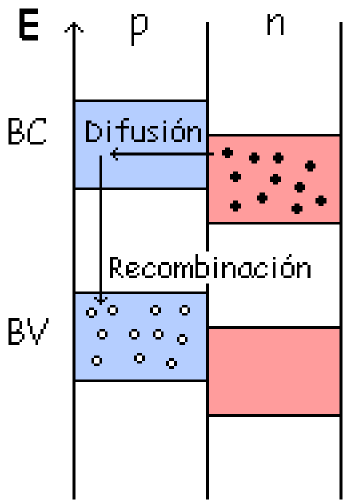
Empieza la Difusión y la Recombinación

Los electrones pasan de derecha a izquierda y se recombinan con los huecos.



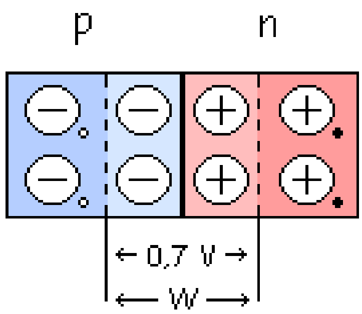
Cruzan y se recombinan los que están al lado de la unión. Se empieza a crear una diferencia de potencial entre una parte y otra, esta diferencia de potencial aumenta hasta que se establezca el equilibrio (Si a 0.7 V, Ge a 0.3 V).

En las Bandas de Energía ocurre lo siguiente: Un electrón va de n a p y luego en p baja de BC a BV.



Al recombinarse, la energía que hay desde el nivel que tenía al que está el hueco al que ha saltado la tiene que saltar y la suelta en forma de calor (un diodo se suele calentar) o también en forma de radiación que puede ser visible (Led) o no.

Esto continúa hasta que se llega a 0.7 V y se llega al equilibrio.



Y se ha creado una diferencia de potencial o anchura de banda (W).

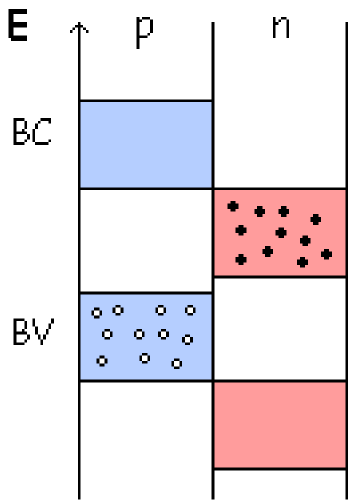
Hasta ahora resumiendo lo que ha ocurrido es:

- Difusión.
- Recombinación.
- Se ha formado una z.c.e. (ó deplexión).

Además de eso las bandas de energía se han desplazado (hasta llegar al equilibrio).

Equilibrio

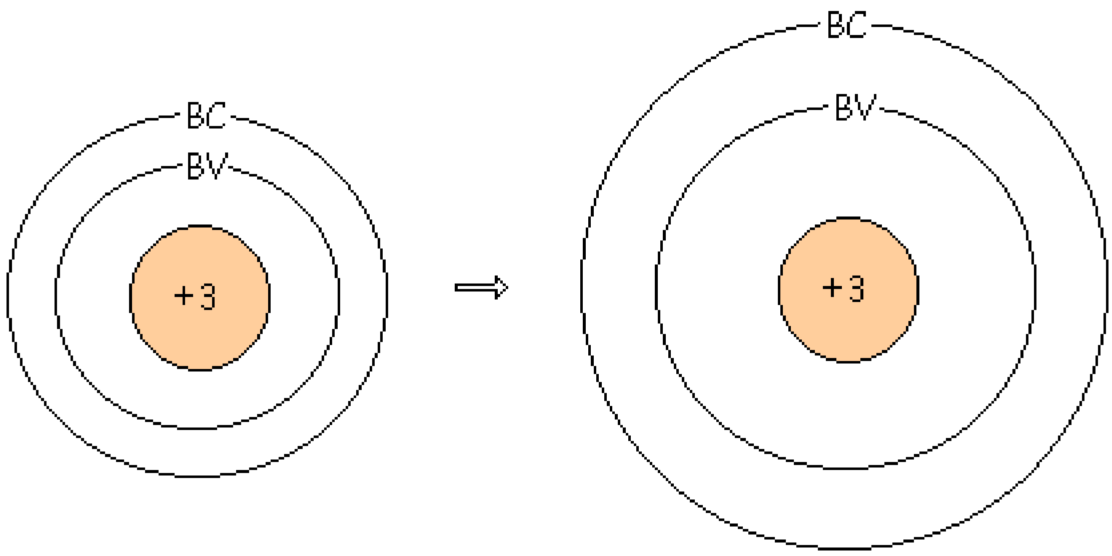
Al llegar a 0.7 V las bandas se han desplazado. Han subido hasta que el nivel inferior de p este al mismo nivel que el nivel superior de n.



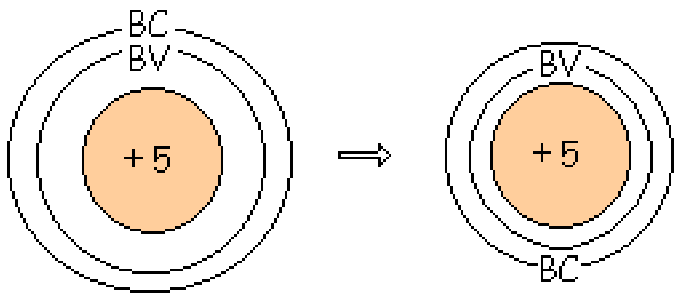
Y se mantendrán en esa posición a no ser que se rompa el equilibrio. En este equilibrio no puede difundirse ningún electrón, no hay difusión ni recombinación si no se rompe el equilibrio.

Veamos porque se han desplazado:

Los átomos de valencia +3 tienen en la última órbita 7 electrones y 1 hueco. Las órbitas se ensanchan por el hueco y esto hace que aumenten los radios de la BV y BC. Aumenta radio lo que implica que aumenta la energía, hasta llegar a la situación antes explicada.



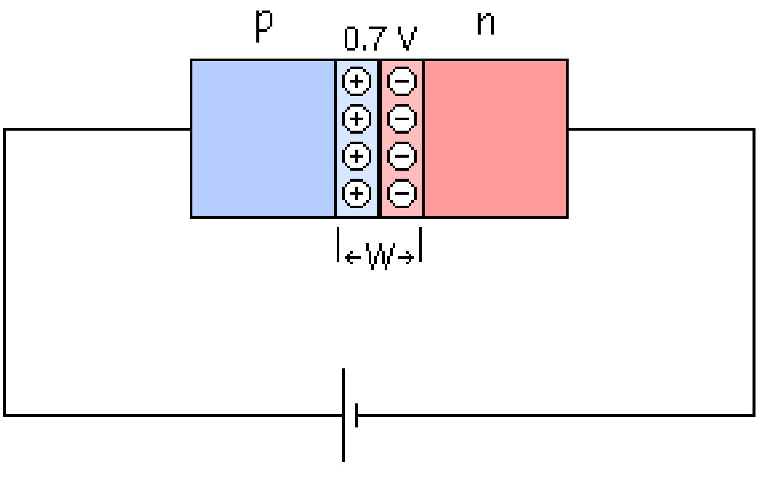
Con los átomos +5 ocurre lo contrario, disminuye el radio con lo que disminuye la energía.



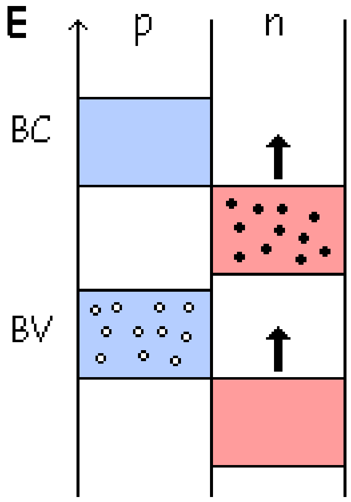
Lo que ha ocurrido es que ya no hay radios coincidentes entre los átomos de valencia +3 y los de valencia +5, por eso se crea el equilibrio.

Polarización Directa

Ahora romperemos el equilibrio poniendo una pila.



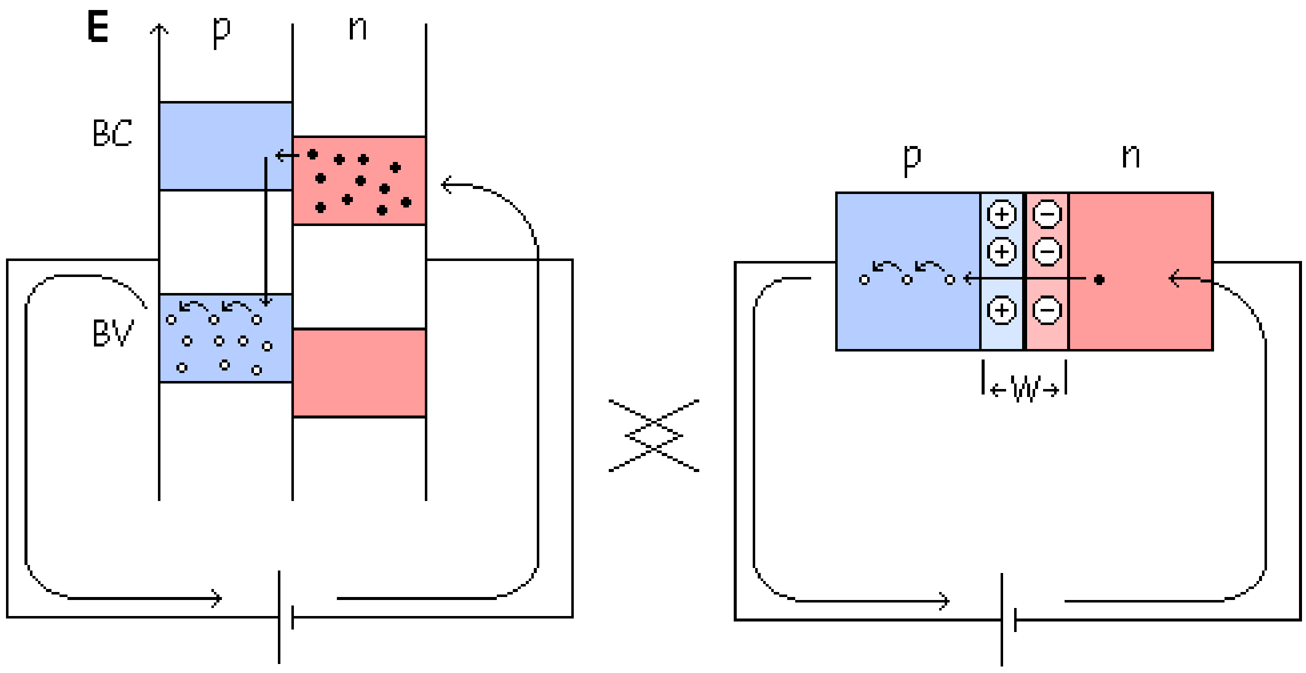
La pila es una "Energía Externa" que hace subir los niveles de la zona n. Esta pila en directa elevará el nivel de energía de la zona n.



Suben las bandas de energía de la zona n y coinciden algunas con la de la zona p, y ya puede haber difusión y recombinación.

Entonces pasan los electrones, se recombinan, etc...Ahora la pila les obliga.

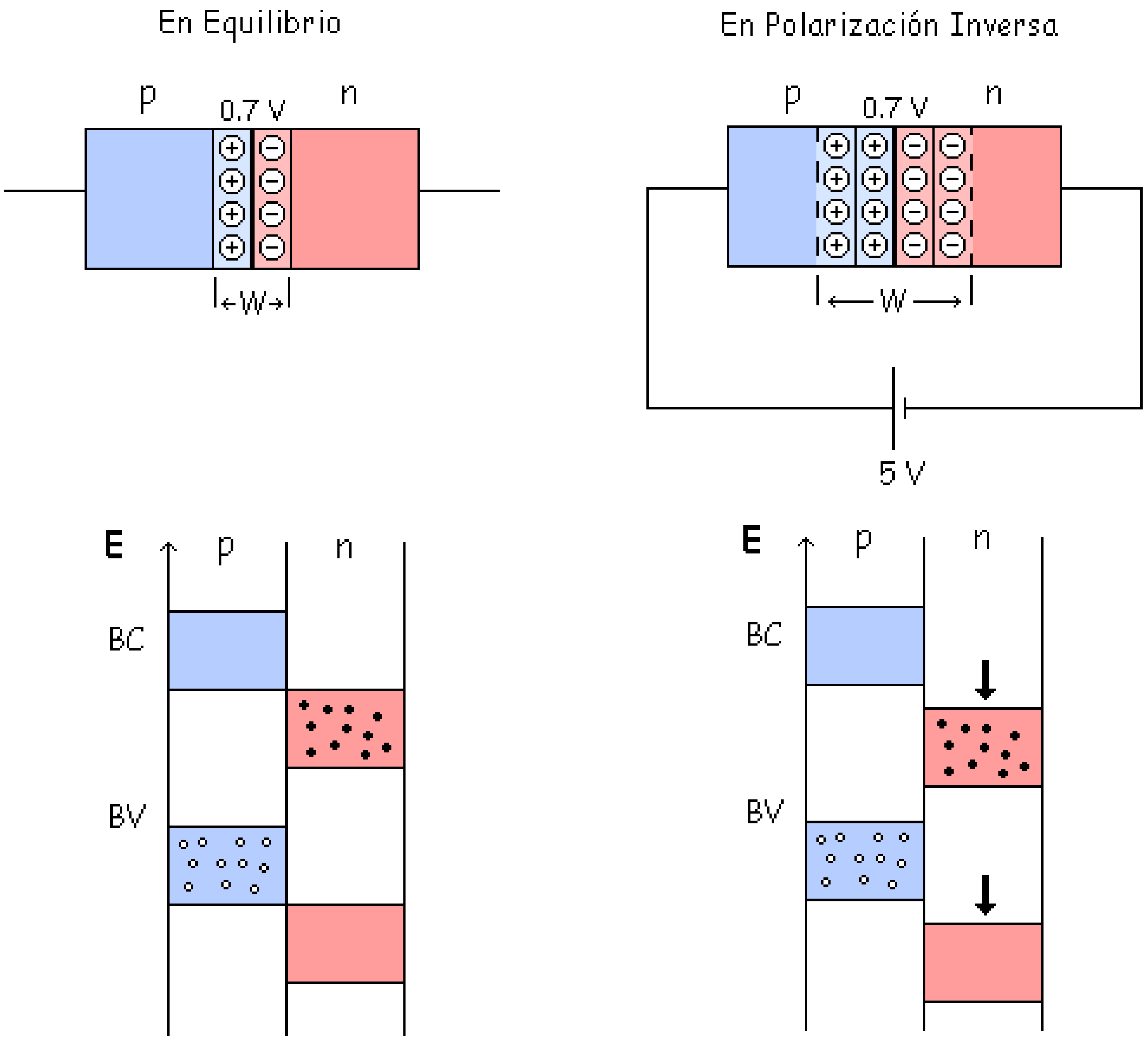
El electrón cruza la W y va pegando saltos de hueco en hueco formando una malla cerrada.



Algunos electrones puede que antes de cruzar bajen y se recombinen con el hueco, pero hay muchos más que se comportan de la otra manera.

Polarización Inversa

Otra forma de romper el equilibrio es con la Polarización Inversa, que se da poniendo la pila al revés que en el caso anterior.



Al poner la pila de esa forma aumenta el W porque la pila atrae a los huecos y los electrones.

Y se ensancha la W hasta igualarse la barrera de potencial al valor de la pila externa. En este ejemplo se llegará al nuevo equilibrio al llegar esa barrera de potencial al valor de 5 V.

Las bandas de energía de la zona n bajan respecto a la zona p, y no hay corriente.

anterior/principal/siguiente

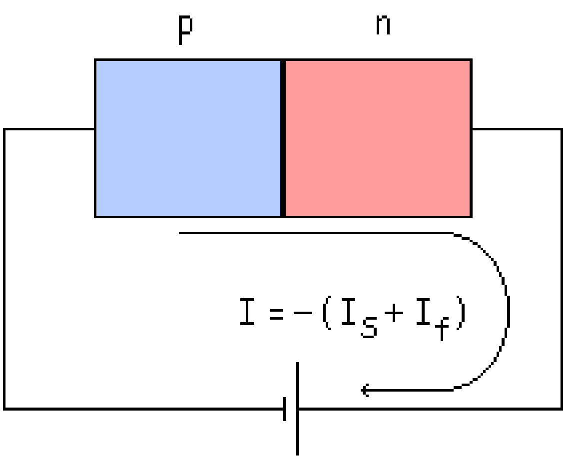
Corrientes en un diodo en polarización inversa

En polarización inversa es más difícil la conducción, porque el electrón libre tiene que subir una barrera de potencial muy grande de n a p al ser mayor el valor de W. Entonces no hay conducción de electrones libres o huecos, no hay corriente.

En esta situación tenemos que tener en cuenta la generación térmica de pares electrón-hueco. Los pocos electrones generados térmicamente pierden energía y bajan de p a n, es la "Corriente Inversa de Saturación" (I_S) que es muy pequeña.

Esa corriente tiene un sentido, siempre se toma la corriente de p a n. Entonces sería negativa en este caso.

Además de esta corriente tenemos otra corriente debida a las fugas, que se denomina "Corriente de Fugas" (I_f).



También ocurre un fenómeno antes de llegar a ese valor, antes de establecerse el valor de I_S.

Mientras van saliendo huecos y electrones, entre el instante inicial y el equilibrio final, hay instantes intermedios. Se crea un transitorio durante el cual en un intervalo breve de tiempo hay una "Corriente Transitoria".

$I_{transitoria}$ puede llegar a tener un valor muy grande.

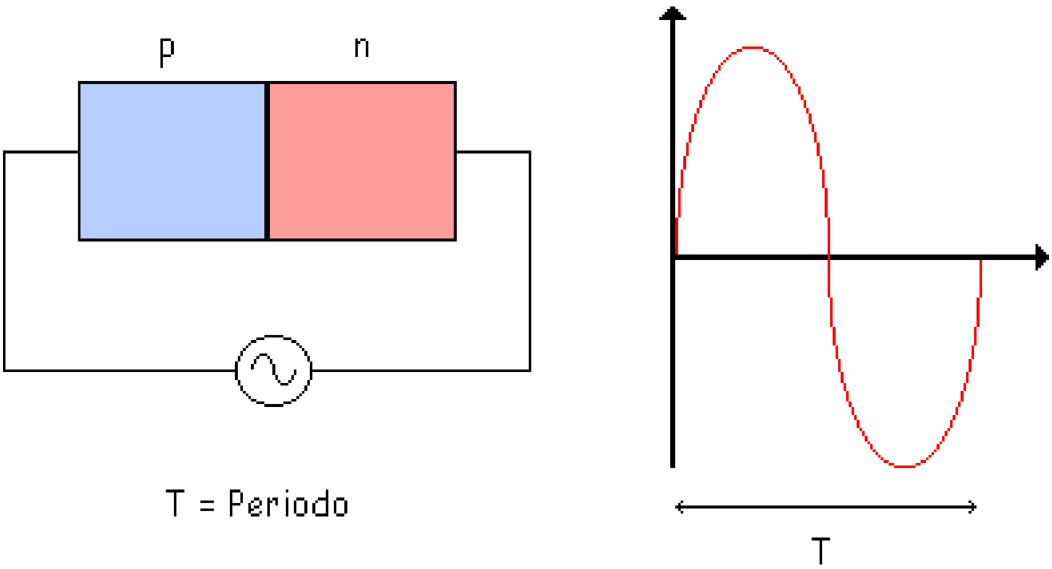
$I_{transitoria}$ = - Grande

Pero dura muy poco, unos nanosegundos. Su duración depende de la resistencia y la capacidad que haya en la malla, así tenemos una "Constante de Tiempo":

$$\tau = R \cdot C$$

Esta constante de tiempo define lo rápido o lenta que es esa malla. Conviene que τ sea pequeña. Suele ser del orden de decenas de nanosegundos.

Si en vez de poner una pila de continua, conectamos el diodo a una onda alterna:



Al tener una onda senoidal el valor de la tensión se está variando continuamente, es como una pila variable, por ello siempre se moverá con retraso debido a esa τ. Por lo tanto, la frecuencia de esa onda senoidal es importante, por ejemplo para una frecuencia de 10 MH:

$$f = 10\ MHz$$

$$T = \frac{1}{f} = \frac{1}{10 \cdot 10^6} = 100\ nseg$$

$$T = 100\ nseg$$

τ (decenas de nseg) ha de ser pequeño respecto a T. Entonces para frecuencias menores o iguales a 10MHz el circuito funcionaría bastante bien.

$$f \leq 10\ MHz$$

La malla tiene que ser suficientemente rápida respecto a la frecuencia de la senoidal.

Tenemos que la I_f (Intensidad debida a fugas) es proporcional a la tensión, mientras que la I_S depende de la temperatura (I_S aumenta 7 % por cada ºC).

anterior/principal/siguiente

INTRODUCCIÓN

En el presente manual se hace entrega de diversos problemas matemáticos con el objetivo de contribuir al desarrollo de los aprendizajes de estudiantes de primer año promoción 2019, pertenecientes a las carreras de Ingeniería civil, Ingeniería civil industrial, Ingeniería civil eléctrica, Ingeniería civil informática, Ingeniería comercial, Contador auditor y Pedagogía en Educación Medía en Matemática en los ramos de Álgebra, Cálculo e Introducción al Análisis, por ende se detalla con precisión cada procedimiento inmerso en la resolución de un problema y los contenidos que el educando debe saber para dar solución a las interrogantes planteadas.

¿POR QUÉ ESTUDIAR EL ÁLGEBRA?

Porque es una rama de la matemática que desarrolla habilidades transcendentales en los estudiantes, permitiendo que éstos modelen algebraicamente distintas situaciones de la vida cotidiana, con el objetivo de encontrar soluciones que mejoren la calidad de vida de una sociedad. Por otro lado, desarrolla la creatividad, el pensamiento crítico y reflexivo, promueve la innovación, desarrolla el razonamiento deductivo y amplía la perspectiva del estudiante hacia su entorno que lo rodea, impulsándolo a indagar en más conocimiento, para comprender los acontecimientos físicos que ocurren en el mundo.

Software de Apoyo

¿Has usado alguna vez una herramienta que te ayude a resolver tus ejercicios?
En internet existe una gran cantidad de opciones y una de ellas es:

GeoGebra

Escanéa el código QR para visitar el sitio web

¡Te invito a descubrir sus prestaciones! Y claro, a buscar otras opciones hasta que elijas la que más te acomode.

CONCEPTOS FUNDAMENTALES DEL ÁLGEBRA

En Aritmética las cantidades se representan por números que representan valores determinados, por ejemplo el número 5 representa el valor de cinco y si queremos expresar otra cantidad deberemos escoger un número distinto de 5.
A diferencia de la Aritmética, en Álgebra las cantidades se estudian de forma general, utilizando cantidades representadas por símbolos que no solo son los números si no que otros como las letras del alfabeto que pueden representar cualquier valor único.
Una letra representa cualquier valor porque dicho símbolo asumirá el valor que nosotros le asignemos, y es único porque dentro de un mismo problema esa letra no puede representar otro valor distinto al que le hemos asignado.
La gama de problemas que pueden resolverse a través del álgebra es extensa y para enfrentar un problema se requiere transformar la situación a expresiones algebraicas que contienen símbolos y signos.

Símbolos

Números
Se emplean para representar cantidades conocidas y determinadas por su propio valor.
Letras
Se emplean para representar todo tipo de cantidades, ya sea conocidas o desconocidas.
Cantidades conocidas
Suelen estar representadas por las primeras letras del alfabeto (a, b, c, ...).
Cantidades desconocidas
Suelen estar representadas por las últimas letras del alfabeto (..., w, x, y, z).

TIP
Una misma letra puede representar distintas cantidades solo si se escribe de forma diferente por ejemplo agregando súper índices ($a' \neq a''$) o sub índices ($a_1 \neq a_2$).

Signos
Los signos se agrupan en tres clases:

Signos de operación:
Las operaciones son las mismas que conocemos de la aritmética, ellas y sus respectivos símbolos son:
Adición o Suma: $+$
Sustracción o Resta: $-$
Multiplicación o producto: $\cdot$ ó x
División: $\div$ ó $/$
Potencias: a^n donde a es la base y n es el exponente de la potencia.
Raíces: $\sqrt[n]{a}$ donde a es la base y n es el exponente de la raíz, el signo $\sqrt{}$ es el radical.
Signos de relación
Se emplean para indicar la relación que hay entre dos cantidades
$=$: Igualdad
$>$: Mayor estricto
$\geq$: Mayor ó igual que
$<$: Menor estricto
$\leq$: Menor ó igual que

Signos de agrupación

Se refiere a la variedad de paréntesis ( ),[ ],{ }, entre otros, y se emplean para indicar que la operación entre ellos debe efectuarse en primer lugar.

Ahora que ya hemos definido los símbolos y signos del álgebra podemos definir un término algebraico y luego una expresión algebraica y sus respectivas clasificaciones.

Término algebraico

Un término algebraico consta de un símbolo o varios símbolos que no están separados entre sí por los signos de adición o sustracción.

Ejemplo:

$$\left.\begin{array}{c} a \\ -3b \\ 2xy^2 \\ \dfrac{4a}{3x} \end{array}\right\} \textit{Son términos algebraicos}$$

Un término algebraico se compone de su signo, coeficiente, su parte literal y su grado.

- El signo de un término algebraico es positivo o negativo, cuando es positivo generalmente se omite escribir el signo.
- El coeficiente de un término algebraico es la cantidad numérica.
- La parte literal de un término algebraico corresponde a sus letras incluyendo sus exponentes.
- El grado de un término algebraico puede ser absoluto (suma de los exponentes de todos los factores de su parte literal) o bien relativo, es decir, con respecto a cada factor literal.

Ejemplo:

Para el término algebraico $2xy^2$ determine sus componentes:
Signo: Positivo
Coeficiente: 2
Parte literal: xy^2
Grado Absoluto: $1+2=3$
Grado relativo a x: 1
Grado relativo a y: 2

Expresión algebraica

Una expresión algebraica es un término algebraico (monomio) o bien la suma o resta de dos o más de ellos (polinomio).

Dependiendo de la cantidad de términos los polinomios reciben algunos nombres especiales, así si un polinomio se compone de la suma o resta de dos monomios entonces se le llama binomio, si se compone de tres términos sumados y/o restados entonces se le llama trinomio. Si se compone de la suma y/o resta de cuatro o más términos entonces se le llama genéricamente polinomio.

Ejemplo:

$$2a \quad \textit{es un monomio}$$

$$2xy^2 + z \quad \textit{es un binomio}$$

$$\frac{4a}{3x} - 2y + w \quad \textit{es un trinomio}$$

$$2xy^2 + z + 2b + 1 \quad \textit{es un polinomio}$$

El grado absoluto de un polinomio corresponde al grado de su término de mayor grado, mientras que el grado relativo a una letra es el mayor exponente de la letra en el polinomio.

Ejemplo:
Sea el polinomio $ax^4-5x^3+x^2-3x$ determine su grado absoluto y relativo a sus factores literales
Grado absoluto: 5
Grado relativo a x: 4
Grado relativo a a: 1

Términos semejantes

Dos o más términos son semejantes cuando tienen igual parte literal (incluyendo sus exponentes).

Reducción de términos semejantes

La reducción de términos semejantes es una operación que tiene por objetivo convertir en un solo término dos o más términos semejantes.

Ejemplo:
Reconocer Términos Semejantes:

Término algebraico 1	Término algebraico 2	¿Término 1 y término 2 son semejantes?
$2a$	a	Si
$-2x$	$7x$	Si
$-5a^3 b^2$	$10a^3 b^2$	Si
$3ax^3$	$3ax^4$	No

Ejemplos

Reducir términos semejantes

a) $2a+5a=7a$

b) $-2a+5a=3a$

c) $\frac{1}{2}ab+\frac{2}{3}ab+b=\frac{7}{6}ab+b$

d) $-m-3m-6m+5m=-5m$

e) $3a^{(x-2)}+5a^{(x-2)}-a^{(x-3)}+2a^{(x-3)}+yz^2+yz=8a^{(x-2)}+a^{(x-3)}+yz^2+yz$

EJERCICIOS PROPUESTOS

Reduce, si es posible, los siguientes polinomios:

a) $(a^2-3a)-(3a+a^2)$

b) $x^3+y^2-(3x^3-2y^2)+(y^2-x^3)-(4y^2-6x^3)$

c) $a^2-\frac{a^2}{2}+\frac{a^2}{3}$

d) $7a-\left(\frac{3a}{4}-\frac{1}{2}(a-2)+\frac{a}{2}\right)$

Propiedades de los Reales

A partir del estudio de las operaciones en los conjuntos $\mathbb{Z}$ y $\mathbb{Q}$, concluimos que las sustracciones se pueden considerar como adiciones y las divisiones como multiplicaciones y, por lo tanto, + y · son las operaciones relevantes en dichos conjuntos. Ahora, si dotamos al conjunto R de estas mismas operaciones, formamos una estructura algebraica que, en este caso, recibe el nombre de Cuerpo. Este nombre indica que la estructura $(\mathbb{R}, +, \cdot)$ cumple varias propiedades que son las que nos permiten usar el álgebra como una herramienta eficaz.

TIP

Considerando la existencia de los números negativos (conjunto $\mathbb{Z}$) podemos representar una sustracción como una adición.

Ejemplo

$5-7=-2$ Se puede reescribir $5+(-7)=-2$
$8-4=4$ Se puede reescribir $8+(-4)=4$

TIP

En cuanto a la división, esta puede considerarse una multiplicación gracias al conjunto de los números Racionales ($\mathbb{Q}$) y también con el uso de las potencias.

Ejemplo $\frac{5}{2}$ Puede reescribirse $5 \cdot \frac{1}{2}$ ó bien $5 \cdot 2^{(-1)}$

A continuación indicaremos simbólicamente estas propiedades:

Propiedades de la adición en $\mathbb{R}$

1. Clausura: Si $a \in \mathbb{R}$ y $b \in \mathbb{R}$, entonces $(a+b) \in \mathbb{R}$
2. Asociatividad: Si $a, b, c \in \mathbb{R}$, entonces $a+(b+c)=(a+b)+c$
3. Conmutatividad: Si $a, b \in \mathbb{R}$, entonces $a+b=b+a$
4. Elemento neutro aditivo: Si $a \in \mathbb{R}$, entonces existe un único elemento neutro aditivo, el 0, tal que $a+0=a$
5. Elemento inverso aditivo u opuesto. Para todo elemento $a \in \mathbb{R}$, existe un elemento $-a \in \mathbb{R}$, tal que $a+(-a)=0$

Propiedades de la multiplicación en $\mathbb{R}$.

6. Clausura: Si $a \in \mathbb{R}$ y $b \in \mathbb{R}$, entonces $ab \in \mathbb{R}$

7. Asociatividad: Si $a, b, c \in \mathbb{R}$, entonces $a(bc) = (ab)c$

8. Conmutatividad: Si $a, b \in \mathbb{R}$, entonces $ab = ba$

9. Elemento neutro multiplicativo: Si $a \in \mathbb{R}$, entonces existe un único elemento neutro multiplicativo, el 1, tal que $a \cdot 1 = a$

10. Elemento inverso multiplicativo o recíproco: Para cada elemento $a^{-1} \in \mathbb{R}$, menos para el 0, existe un elemento $a \cdot a^{-1} \in \mathbb{R}$, tal que . El elemento a^{-1} también se suele escribir $\frac{1}{a}$.

Para la adición y la multiplicación en $\mathbb{R}$.

11. Distributividad :Si $a, b, c \in \mathbb{R}$, entonces $a(b + c) = ab + ac$

Operatoria con polinomios

Adición (y sustracción) de polinomios

La adición y sustracción o suma y resta respectivamente de monomios a sido ejemplificada en la reducción de términos semejantes y su operatoria es extensible a polimonios como se ejemplifica en los siguientes ejercicios resueltos.

Ejemplos

Sean los siguientes polinomios

Polinomio 1: $a^3b-b^4+ab^3+5a^2+b^2$

Polinomio 2: $-2a^2 b2+4ab^3+2b^4$

a) Sumar los polinomios:

$(a^3 b-b^4+ab^3+5a^2 b^2)+(-2a^2 b^2+4ab^3+2b^4)$

$=a^3 b-b^4+ab^3+5a^2 b^2-2a^2 b^2+4ab^3+2b^4$

$=a^3 b+5a^2 b^2-2a^2 b^2+4ab^3+ab^3+2b^4-b^4$

$=a^3 b+3a^2 b^2+5ab^3+b^4$

b) Restar del Polinomio 1 el polinomio 2:

$a^3b-b^4+ab^3+5a^2 b^2-(-2a^2 b^2+4ab^3+2b^4)$

$=a^3 b-b^4+ab^3+5a^2 b^2+2a^2 b^2-4ab^3-2b^4$

$=a^3 b+5a^2 b^2+2a^2 b^2-4ab^3+ab^3-2b^4-b^4$

$=a^3 b+7a^2 b^2-3ab^3-3b^4$

Multiplicación

Para hallar el producto de dos polinomios usamos las propiedades distributivas, ley de los signos, las leyes de los exponentes y la reducción de términos semejantes, como se muestra en el ejemplo que sigue.

Ejemplo

Sean los siguientes polinomios

Polinomio 1: $x^3 +3x-1$
Polinomio 2: $2x^2 -4x+5$

Calculamos el producto aplicando las propiedades distributivas y luego efectuamos el producto de monomios y la reducción de términos semejantes expresando el resultado final, que es otro polinomio, con sus términos ordenados de mayor a menor grado.

$$
\begin{aligned}
&=(x^3 +3x-1)(2x^2 -4x+5)\\
&=(x^3 +3x-1)(2x^2)+(x^3 +3x-1)(-4x)+(x^3 +3x-1)(5)\\
&=(x^3)(2x^2)+(3x)(2x^2)+(-1)(2x^2)+(x^3)(-4x)+(3x)(-4x)\\
&\qquad +(-1)(-4x)+(x^3)(5)+(3x)(5)+(-1)(5)\\
&=2x^5 +6x^3 -2x^2-4x^4 -12x^2 +4x+5x^3 +15x-5\\
&=2x^5 -4x^4 +11x^3 -14x^2+19x-5
\end{aligned}
$$

Productos notables

Ciertos productos de binomios se presentan con tanta frecuencia que debe aprender a reconocerlos, estos son multiplicaciones de expresiones algebraicas fácilmente reconocibles y que para determinar su desarrollo basta con aplicar una fórmula general conocida.

Cuadrado de binomio

Corresponde a la expresión $(x + y)^2$ ó $(x - y)^2$ que representa el producto $(x + y)(x + y)$ ó $(x - y)(x - y)$ respectivamente.
Determinemos la fórmula general de su desarrollo.

$$
\begin{aligned}
(x + y)^2 &= (x + y)(x + y) && \text{aplicando distributividad}\\
&= (x + y)x + (x + y)y && \text{distribuyendo otra vez}\\
&= xx + xy + xy + yy && \text{y reduciendo}\\
&= x^2 + 2xy + y^2 && \text{que es la fórmula buscada.}
\end{aligned}
$$

Para el binomio $(x - y)^2$, su fórmula de desarrollo es $x^2- 2xy + y^2$.
Luego, en general podemos anotar:

$$\boxed{(x \pm y)^2 = x^2 \pm 2xy + y^2}$$

Ejemplos

a) $(3a + 4)^2 = (3a)^2 + 2\cdot 3a\cdot 4 + 4^2$
$= 9a^2 + 24a + 16$

b) $(a - 3b)^2 = a^2 - 2\text{-}a\text{-}3b + (3b)^2$
$= a^2 - 6ab + 9b^2$

Suma por su diferencia

Corresponde al producto de dos binomios con los mismos términos, pero en un caso se suman y en el otro se restan.

Es decir:

$$(x + y)(x - y)$$

Para determinar la fórmula de desarrollo de este producto, aplicamos sucesivamente la propiedad distributiva.

$$\begin{aligned}(x + y)(x - y) &= (x + y)x - (x + y)y \\ &= xx + yx - (xy + yy) \\ &= x^2 + yx - xy - y^2 \\ &= x^2 + 0 - y^2\end{aligned}$$

Luego su resultado es:

$$\boxed{(x+y)(x-y) = x^2 - y^2}$$

Ejemplos

a) $(3a + 2b)(3a - 2b) = (3a)^2 - (2b)^2$
$= 9a^2 - 4b^2$

b) $(a^2 + 8)(a^2 - 8) = (a2)^2 - 8^2$
$= a^4 - 64$

Producto de binomios con un término común

Corresponde a la multiplicación de dos binomios donde uno de los términos se repite en ambos. Su forma general es:

$$(x + a)(x + b)$$

Determinemos la fórmula de desarrollo empleando el mismo procedimiento de los casos anteriores.

$$\begin{aligned}(x + a)(x + b) &= (x + a)x + (x + a)b \\ &= xx + ax + xb + ab\end{aligned}$$

Y reduciendo términos semejantes se llega a:

$$\boxed{(x+a)(x+b)=x^2+(a+b)x+ab}$$

Ejemplos

a) $(x+4)(x+9)= x^2+(4+9)x+4\cdot 9$
$\quad = x^2+13x+36$

b) $(2x+5)(2x-2) = (2x)\ 2+(5-2)\cdot 2x+5\cdot(-2)$
$\quad = 4x^2+3\cdot 2x+(-10)$
$\quad = 4x^2+6x-10$

c) $(x^2+3)(x^2-4) = (x^2)\ 2+(3-4)x^2+3\cdot(-4)$
$\quad = x^4+(-x^2)+(-12)$
$\quad = x^4-x^2-12$

Cubo de binomio

Corresponde a la expresión $(x+y)^3$ ó $(x-y)^3$ que representa el producto $(x+y)(x+y)(x+y)$ ó $(x-y)(x-y)(x-y)$ respectivamente.

Encontremos su forma general:

$(x+y)^3 = (x+y)^2(x+y)$
$\quad = (x^2+2xy+y^2)(x+y)$
$\quad = (x^2+2xy+y^2)x+(x^2+2xy+y^2)y$
$\quad = x^3+2x^2y+xy^2+x2y+2xy^2+y^3$
$\quad = x^3+3x^2y+3xy^2+y^3$

Para el binomio $(x-y)^3$, su fórmula de desarrollo es $x^3-3x^2y+3xy^2-y^3$. Luego, en general podemos anotar

$$\boxed{(x\pm y)^3=x^3\pm 3x^2y+3xy^2\pm y^3}$$

Ejemplos

a) $(a+2)^3 = a^3+3a^2\cdot 2+3a\cdot 2^2+2^3$
$\quad = a^3+6a^2+12a+8$

b) $(b-2)^3 = b^3-3b^2\cdot 2+3b\cdot 2^2-2^3$
$\quad = b^3-6b^2+12b-8$

c) $(2x-3y)^3 = (2x)^3+3\cdot(2x)^2(3y)+3\cdot(2x)(3y)^2+(3y)^3$
$\quad = 8x^3+36x^2y+54xy^2+27y^3$

Otros productos notables son:

Cuadrado de un trinomio

$$(a+b+c)^2 = a^2+b^2+c^2+2ab+2ac+2bc$$

$$(a-b-c)^2 = a^2+b^2+c^2-2ab-2ac+2bc$$

Suma y resta de cubos

$$(a+b)(a^2-ab+b^2) = a^3+b^3$$

$$(a-b)(a^2+ab+b^2) = a^3-b^3$$

Te invitamos a ver:

EJERCICIOS PROPUESTOS

1. Verifica los siguientes desarrollos, ¿A qué producto notable corresponden?

a) $(p+2b)^2 = p^2+2\cdot p\cdot 2b+(2b)^2 = p^2+4pb+4b^2$

b) $(3m+4n)^2 = (3m)^2+2\cdot 3m\cdot 4n+(4n)^2 = 9m^2+24mn+16n^2$

c) $(5x-y)^2 = (5x)^2+2\cdot 5x\cdot y+(y)^2 = 25x^2+10xy+y^2$

d) $(a^2-3)(a^2+3) = a^4-9$

Desarrolla los siguientes productos notables, ¿A cuál producto notable corresponden?

a) $(3a + x)(3a - x)$

b) $(x + 1)(1 - x)$

c) $(a + 4)(a - 4)$

d) $(x + x2)(x - x2)$

e) $\left(\frac{1}{4}+\frac{x}{3}\right)\left(\frac{1}{4}-\frac{x}{3}\right)$

f) $\left(x+\frac{1}{2}\right)\left(x+\frac{1}{3}\right)$

Factorización

Factorizar un número es expresarlo como una multiplicación de dos o más factores. Por ejemplo, factorizar el 24 es expresarlo como $3\cdot8$, $2\cdot4\cdot3$ ó $12\cdot2$. Ahora bien, factorizar un término algebraico es expresar dicho término como una multiplicación entre diversos coeficientes y/o factores literales.

Ejemplo

$$-3x^2y^3 = -1\cdot3\cdot x\cdot x\cdot y\cdot y\cdot y$$

Sea la expresión $4x^2 - 6x$. Esta expresión no se puede reducir ya que los monomios que la componen no poseen términos semejantes, sin embargo si puede transformarse en una multiplicación factorizándola. Para eso, factoricemos previamente cada término.

$$4x^2 - 6x = 2\cdot2\cdot x\cdot x - 2\cdot3\cdot x$$

Ahora, notamos que se repiten un 2 y una x en los términos de este binomio, luego podemos escribir:

$$4x^2 - 6x = 2x\cdot2x - 2x\cdot3$$

Ahora, sacamos el factor común fuera del binomio de manera que quede multiplicando a los dos términos ya sin el factor común:

$$4x^2 - 6x = 2x\cdot(2x - 3)$$

Ejemplos

a) Factorizar:

$$\begin{aligned} 5a^2b - 15ab + 20a^2b^2 &= 5ab\cdot a - 5ab\cdot3 + 5ab\cdot4ab \\ &= 5ab\cdot(a - 3 + 4ab) \end{aligned}$$

b) Factorizar:

$$\begin{aligned} \frac{3x}{2} + \frac{5y}{6} &= \frac{1}{2}\cdot3x + \frac{1}{2}\cdot\frac{5y}{3} \\ &= \frac{1}{2}\left(3x + \frac{5y}{3}\right) \end{aligned}$$

c) Factorizar: sacar factor común 1

$$\begin{aligned} ax^2 - b &= -1\cdot(-ax2) - 1\cdot b \\ &= -1(-ax2 + b) \\ &= -(b - ax2) \end{aligned}$$

Tip

Sacar factor común **–1** es un procedimiento que permite cambiar los signos de una expresión algebraica.

EJERCICIOS PROPUESTOS

Factorizar las siguientes expresiones

1. $\frac{\pi r^2}{2} + 2\pi r$
2. $-2x^3 - 4x^2 - 6x$
3. $3x + 2y - 5z$

Simplificación de expresiones algebraicas

Ahora aplicaremos este mismo concepto a una fracción cuyo numerador y denominador son expresiones algebraicas.
Sea la fracción $\frac{3x^2}{6x}$. Esta fracción puede simplificarse por $3x$, porque es un divisor común del numerador y del denominador.

Dividiendo nos queda:

$$\frac{3x^2}{6x} = \frac{3x \cdot x}{3x \cdot 2} = \frac{3x}{3x} \cdot \frac{x}{2} = 1 \cdot \frac{x}{2} = \frac{x}{2}$$

Es decir, factorizamos el numerador y el denominador de manera que tengamos el mismo factor. Luego se expresa este factor común como una fracción equivalente a 1 y, como el 1 es el elemento neutro multiplicativo, el factor común "se va" o "desaparece" de la fracción original.

Ejemplos

Simplificar:

a) $\frac{3a^2bc}{12ac} = \frac{3ac \cdot ab}{3ac \cdot 4} = \frac{3ac}{3ac} \cdot \frac{ab}{4} = 1 \cdot \frac{ab}{4} = \frac{ab}{4}$

b) $\frac{5x^3}{20} = \frac{5 \cdot x^3}{5 \cdot 4} = \frac{x^3}{4}$

c) $\frac{9ab^2}{3} = \frac{3 \cdot 3ab^2}{3} = 3ab^2$

EJERCICIOS PROPUESTOS

Simplificar::

a) $\dfrac{9a}{18a^2b}$

b) $\dfrac{-3x^2y}{12x}$

c) $\dfrac{-2a^2}{6a^3}$

d) $\dfrac{5ab-2a}{4a}$

e) $\dfrac{5x^2-20x}{15x^3}$

Lógica

Constantemente usamos el razonamiento lógico para realizar cualquier actividad y este se emplea en matemática para demostrar teoremas, y en ciencias que usan la matemática se utiliza para sacar conclusiones, para verificar la estructura de algoritmos, etc.
La lógica proporciona reglas y técnicas para determinar si es o no válido un argumento dado.

Proposición

Es una expresión con sentido en algún lenguaje que afirma o niega algo y que nos proporciona información.
Las proposiciones en lógica se denotan por símbolos como p, q r, etc. Dichas proposiciones pueden ser compuestas usando los conectivos lógicos:

Ejemplo

Son proposiciones las siguientes afirmaciones:
- El pizarrón es blanco
- El plumón es negro

Las proposiciones del ejemplo pueden ser Verdaderas o Falsas, no aceptan ambigüedades.

Ejemplo

No son proposiciones los siguientes enunciados
- El interruptor
- $2x+1=4$
- ¿Qué hora es?

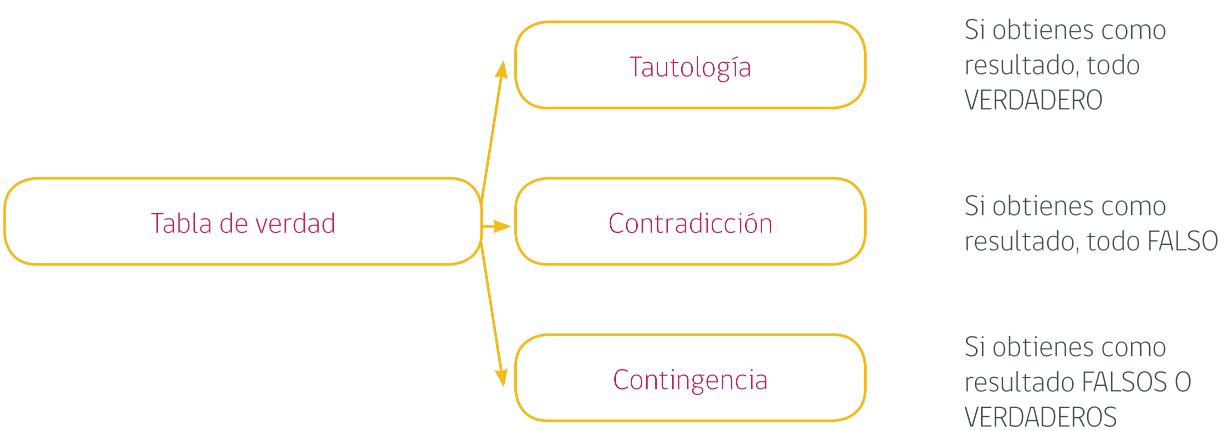
Ya que no pueden ser afirmadas o negadas.

Valor de Verdad

Es una función que define una proposición. El valor de verdad puede ser Verdadero (V) o Falso (F).
Tabla de Verdad
En la figura se detallan los posibles resultados que se pueden obtener a partir de la resolución de una tabla de la verdad.

Tabla de Verdad

En la figura se detallan los posibles resultados que se pueden obtener a partir de la resolución de una tabla de la verdad.



Ejemplo

Resultados de una tabla de verdad

a) Tautología

p	q	$(p \wedge q)$	$(p \wedge q) \Rightarrow p$
V	V	V	V
V	F	F	V
F	F	F	V
F	F	F	V

b) Contradicción

p	q	$(p \vee q)$	$\sim (p \vee q)$	$(p \vee q) \Leftrightarrow \sim (p \vee q)$
V	V	V	F	F
V	F	V	F	F
F	V	V	F	F
F	F	F	V	F

c) Contingencia

p	q	$(p \Leftrightarrow q)$	$\vee$	$[(p \Rightarrow q) \wedge (q \Rightarrow p)]$		
V	V	V	V	V	V	V
V	F	F	F	F	F	V
F	V	F	F	V	F	F
F	F	V	V	V	V	V

Notación de operadores en Lógica:

$\wedge$, se denomina conjunción, se lee y
$\vee$, lo llamaremos disyunción, se lee o.
$\underline{\vee}$, lo llamaremos disyunción excluyente, se lee o.... o...
$\sim$, se denomina negación, se lee no.
$\rightarrow$, se denomina condicional, se lee si..., entonces...
$\leftrightarrow$, se llama bicondicional, se lee si, y sólo si

Resultados de los operadores lógicos

Ilustración 2: Resultados de los operadores lógicos

p	q	$p \wedge q$	$p \vee q$	$p \underline{\vee} q$	$p \rightarrow q$	$p \leftrightarrow q$
V	V	V	V	F	V	V
V	F	F	V	V	F	F
F	V	F	V	V	V	F
F	F	F	F	F	V	V

Tip

- Debes tener en consideración el o excluyente $\underline{\vee}$, ya que como es un operador no trivial, aparece en ejercicios de evaluaciones.
- Los operadores $\rightarrow$ y $\leftrightarrow$, dominan sobre los operadores $\vee$; $\wedge$.
- La cantidad de filas de la tabla de verdad se calculan a través del algoritmo 2^n, donde el 2 corresponde a las opciones verdaderas o falsas y n es la cantidad de proposiciones que se utilizarán en un ejercicio determinado.

Ejemplo

Resolver mediante una tabla de verdad: $p \wedge q \rightarrow q$

En este caso se realiza primero la operación $q \rightarrow q$ y luego a este resultado aplicarle la operación "$p\wedge$". Resolvamos, en este caso el algoritmo quedará de la siguiente forma: $2^2=4$, ya que son dos proposiciones, por ende, son 4 filas como se muestra a continuación.

p	q	$q \rightarrow q$	$p \wedge q \rightarrow q$
V	V	V	V
V	F	V	V
F	V	V	F
F	F	V	F

CONTINGENCIA

Uso de paréntesis

El uso de paréntesis es un símbolo que forma parte de la lógica secuencial, el uso de ellos es lógico, sin los paréntesis las fórmulas o expresiones lógicas pueden carecer de sentido.

Ejemplo

Notar que las expresiones son claramente distintas

$$p \rightarrow (q \vee r)$$
$$(p \rightarrow q) \vee r$$

Ejemplo

Desarrolle la tabla de verdad de la siguiente proposición

$$\sim(p \wedge \sim q)$$

Primero completamos las dos primeras columnas correspondientes a las proposiciones p y q, sabemos que la cantidad de filas es igual a 4 dada la fórmula 2^n en donde n es la cantidad de proposiciones (p y q) presentes en la proposición compuesta.

p	q
V	V
V	F
F	V
F	F

Luego agregamos la columna de la negación de q

p	q	$\sim q$
V	V	F
V	F	V
F	V	F
F	F	V

Posteriormente incluimos una columna para el paréntesis

p	q	$\sim q$	$(p \wedge \sim q)$
V	V	F	F
V	F	V	V
F	V	F	F
F	F	V	F

Y por último aplicamos la negación del paréntesis, en la columna correspondiente

p	q	$\sim q$	$(p \wedge \sim q)$	$\sim (p \wedge \sim q)$
V	V	F	F	V
V	F	V	V	F
F	V	F	F	V
F	F	V	F	V

La última columna contiene la solución, que es una contingencia.

Tip

Estas fórmulas se utilizan para demostrar si se cumple una equivalencia de proposiciones. Se parte aplicando las propiedades a un lado de la equivalencia y se debe llegar a la proposición expuesta al otro lado de la equivalencia.

Ejemplo

Determine si $[p \rightarrow (p \wedge q)] \leftrightarrow [(\sim q \rightarrow \sim p) \veebar (p \vee q)]$ es una equivalencia lógica.

p	q	$\sim q$	$\sim p$	$p \wedge q$	$p \rightarrow (p \wedge q)$	$\sim q \rightarrow \sim p$	$p \vee q$	$[(\sim q \rightarrow \sim p) \veebar (p \vee q)]$
V	V	F	F	V	V	V	V	F
V	F	V	F	F	F	F	V	V
F	V	F	V	F	V	V	V	F
F	F	V	V	F	V	V	F	V

$[p \rightarrow (p \wedge q)] \leftrightarrow [(\sim q \rightarrow \sim p) \veebar (p \vee q)]$

F
F
V
V

$\longrightarrow$ **Contingencia**

Como el bicondicional no es una tautología, la proposición compuesta $[p\rightarrow(p\wedge q)]\leftrightarrow[(\sim q\rightarrow\sim p)\veebar(p\vee q)]$, no es una equivalencia lógica.

Ejemplo

Sean las siguientes proposiciones:

$p:\ 3x+3y=9$

$q:\ 5x+y=7$

$r:\ 5y+x=11 \quad \text{con } x=1, y\neq 2, y\in\mathbb{R}$

Determine el valor de verdad de la siguiente proposición compuesta

$[(p\vee q)\wedge\sim r]\rightarrow\sim q$

Primero determinamos el valor de verdad de p, q, r.

$p=q=r=F$

Luego reemplazamos el valor de verdad de las proposiciones simples en la proposición compuesta:

$[(p\vee q)\wedge\sim r]\rightarrow\sim q$

$[(F\vee F)\wedge\sim F]\rightarrow\sim F$

$[(F\vee F)\wedge V]\rightarrow V$

$[(F)\wedge V]\rightarrow V$

$[F]\rightarrow V$

V

EJERCICIOS PROPUESTOS

1-Considerando las mismas proposiciones p, q, r del ejemplo anterior determine el valor de verdad de:

a) $\sim[(\sim q\rightarrow\sim p)\vee\sim r]$

b) $[\sim p\vee\sim r]\rightarrow\sim q$

2. Desarrolle las siguientes tablas de verdad

a) $(p\wedge\sim q)\rightarrow p$

b) $(\sim p\vee q)\rightarrow q$

c) $\sim(p\rightarrow q)\wedge p$

d) $(p\vee\sim q)\vee(\sim p\wedge\sim q)$

e) $\sim(p\rightarrow r)\wedge q$

f) $[\sim p\rightarrow(q\wedge r)]\leftrightarrow\sim q$

Demostraciones de proposiciones

A continuación, se presentan diversas fórmulas que cumplen las proposiciones, para realizar demostraciones de equivalencia.

Fórmulas importantes:

Ley de la doble negación:

$\sim(\sim p)\leftrightarrow p$

Leyes de Idempotencia

$p\wedge p\leftrightarrow p$

$p\vee p\leftrightarrow p$

Leyes del Complemento

$p\wedge\sim p\leftrightarrow F$

$p\vee\sim p\leftrightarrow V$

Leyes de Identidad

$p\wedge V\leftrightarrow p$

$p\vee V\leftrightarrow V$

$p\wedge F\leftrightarrow F$

$p\vee F\leftrightarrow p$

Leyes Conmutativa

$p\wedge q\leftrightarrow q\wedge p$

$p\vee q\leftrightarrow q\vee p$

Leyes Asociativa

$(p\wedge q)\wedge r\leftrightarrow p\wedge(q\wedge r)$

Negaciones importantes

$\sim(p\rightarrow q)\leftrightarrow(p\wedge\sim q)$

$\sim(p\leftrightarrow q)\leftrightarrow(p\wedge\sim q)\vee(q\wedge\sim p)$

Leyes Asociativa:

$(p\vee q)\wedge r\leftrightarrow p\vee(q\vee r)$

$(p\leftrightarrow q)\leftrightarrow r \quad p\leftrightarrow(q\leftrightarrow r)$

Leyes Distributiva

$p\wedge(q\vee r)\leftrightarrow(p\wedge q)\vee(p\wedge r)$

$p\vee(q\wedge r)\leftrightarrow(p\vee q)\wedge(p\vee r)$

Leyes de Morgan

$(p\wedge q)\leftrightarrow\sim p\vee\sim q$

$(p\vee q)\leftrightarrow\sim p\wedge\sim q$

Leyes de Absorción

$p\wedge(p\vee q)\leftrightarrow p$

$p\vee(p\wedge q)\leftrightarrow p$

Ley del Contra-recíproco

$(p\rightarrow q)\leftrightarrow(\sim q\rightarrow\sim p)$

Otras leyes

$(p\leftrightarrow q) \quad (p\rightarrow q)\wedge(q\rightarrow p)$

$(p\leftrightarrow q) \quad (p\wedge q)\vee(\sim p\wedge\sim q)$

$(p\rightarrow q) \quad (\sim p\vee q)$

Ejemplo

Demuestre la siguiente proposición $(p\vee q)\rightarrow q\leftrightarrow p\rightarrow q$

Iniciamos de izquierda a derecha

$(p\vee q)\rightarrow q$	
$\sim(p\vee q)\vee q$	otras leyes $(p\rightarrow q)\leftrightarrow(\sim p\vee q)$
$(\sim p\wedge\sim q)\vee q$	Leyes de Morgan $\sim(p\vee q)\leftrightarrow\sim p\wedge\sim q$
$(\sim p\vee q)\wedge(\sim q\vee q)$	Leyes Distributiva $p\vee(q\wedge r)\leftrightarrow(p\vee q)\wedge(p\vee r)$
$(\sim p\vee q)\wedge V$	Leyes del Complemento $p\vee\sim p\leftrightarrow V$
$(\sim p\vee q)$	Leyes de identidad $p\wedge V\leftrightarrow p$
$(p\rightarrow q)$	otras leyes $(p\rightarrow q)\leftrightarrow(\sim p\vee q)$

Ejemplo

Simplifique la expresión $(p \vee q) \rightarrow \sim p$ e indique cada propiedad utilizada en cada paso.

$(p \vee q) \rightarrow \sim p$	
$\sim(p \vee q) \vee \sim p$	Por propiedad condicional
$(\sim p \wedge \sim q) \vee \sim p$	Por leyes de Morgan
$\sim p \vee (\sim p \wedge \sim q)$	Por propiedad conmutativa
$\sim p$	Por propiedad de absorción

$\therefore (p \vee q) \rightarrow \sim p \leftrightarrow \sim p$

Cuantificadores Lógicos

Existen tres tipos de cuantificadores típicos, que se presentan a continuación:

Cuantificador universal

Nos indica que una función proposicional es verdadera para todos los elementos de un conjunto y se denota por $\forall$

Cuantificador existencial

Nos indica que una función proposicional es verdadera para algunos elementos de un conjunto y se denota por $\exists$.

Cuantificador existencia y unicidad

Nos indica que una función proposicional es verdadera para un único elemento de un conjunto y se denota por $\exists!$

Negaciones de proposiciones con cuantificadores

$\sim(\forall x \in A : p(x)) \leftrightarrow \exists x \in A : \sim p(x)$

$\sim(\exists x \in A : p(x)) \leftrightarrow \forall x \in A : \sim p(x)$

$\sim(\exists! x \in A : p(x)) \leftrightarrow (\forall x \in A : \sim p(x)) \vee (\exists x,y \in A : (p(x) \wedge p(y)) \wedge x \neq y)$

Ejemplo

Escriba la negación de la proposición:

$(\exists\, x \in A)\ (\forall\, y \in B)\ [\,(\,(\,p(x) \wedge q(x)\,) \vee r(y)) \rightarrow (x,y) \in AXB]$

Luego:

$\sim (\,(\,\exists\, x \in A\,)\ (\,\forall\, y \in B\,)\ [\,(\,(\,p(x) \wedge q(x)\,) \vee r(y)\,) \rightarrow (x,y) \in A\ X\ B])$

$\Leftrightarrow (\forall x \in A)\ (\exists y \in B)\ [\,(\,(p(x) \wedge q(x)\,) \vee r\,(y)\,) \wedge (x,y) \notin A\ X\ B]$

Ejemplo

Escriba la negación de “cada número real positivo tiene un inverso multiplicativo”.
Sea el universo el conjunto de todos los números reales; el enunciado puede representarse por:
$\forall x, x>0 \Rightarrow \exists y, xy=1$

Su negación es:
$\sim(\forall x, x>0 \Rightarrow \exists y, xy=1)$

Lo cual resulta en
$\exists x, \sim(x>0 \Rightarrow \exists y, xy=1)$

Esta última se lee: Existe un número positivo x para el que no hay inverso multiplicativo.

EJERCICIOS PROPUESTOS

a) Niegue la proposición: $(\sim p \rightarrow q \wedge r)$

b) Considere el conjunto $C=\{8,9,10\}$ y la proposición:
$\forall x \in C: \exists y \in C, x-y<0$,
Determine el valor de verdad de la proposición.

c) Niegue la proposición $\forall x \in C: \exists y \in C, x-y<0$

Tip

Es importante mencionar que en este ejercicio no sólo se realizó la negación de los cuantificadores, también se aprecia la negación de una proposición lógica, ya que se niega un condicional $\longrightarrow$, obteniendo como resultado $\wedge$.

Te invitamos a ver:

Teoría de conjuntos

Un conjunto es una colección de objetos que está bien definido y se denotan por letras mayúsculas. Cada objeto de un conjunto se llama elemento.

Ejemplo:

Son conjuntos:
A = {Los alumnos de la carrera de pedagogía en matemáticas de la UCSC}
B = {a, e, i, o, u}
C= {Los números naturales mayores a 5 y menores que 10 }

Si un elemento pertenece a un conjunto se denota con el símbolo $\in$ y si no pertenece se usa el símbolo $\notin$.

Ejemplo:

Denotación de pertenencia
Sea el conjunto B tal que $B = \{a, e, i, o, u\}$
$a \in B$
$j \notin B$

Formas de escribir un conjunto

Usualmente un conjunto se escribe de dos maneras:
Por Comprensión: En esta forma se escribe una característica de los elementos
Por Extensión: Escritura en la cual los elementos se identifican.
Ejemplo
Conjunto C expresado por compresión:
C= {Los números naturales mayores a 5 y menores que 10 }
Conjunto C expresado por extensión:
$C= \{6, 7, 8, 9\}$
O bien
$C = \{n \,/\, 5<n<10\}$

EJERCICIO PROPUESTO

Escribir por extensión los siguientes conjuntos
A = {x / x es una vocal de la palabra “ambiguo”}
B = {x / x es una vocal de la palabra “amapola”}

Tipos de conjuntos

Conjunto Vacío

Conjunto Vacío este conjunto es aquel que no tiene elementos. Se simboliza por $\varnothing$ ó $\{\}$.

Ejemplo
Conjunto Vacío: conjunto de canciones rancheras interpretadas por el grupo Metallica.

Conjunto Universo

Conjunto Universo Es el conjunto que contiene todos los elementos a los cuales pudiéramos hacer referencia en un momento dado, estos pueden ser infinitos o finitos.
Ejemplo

El conjunto de alumnos de un curso es finito
El conjunto de los números reales es infinito

Conjuntos Disjuntos

Son aquellos conjuntos que no tienen ningún elemento en común
Ejemplo
El conjunto de alumnos aprobados en Algebra es un conjunto disjunto con el de los alumnos reprobados.

Tip

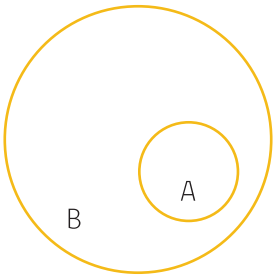
Conjuntos de alta importancia son los conjuntos numéricos, a saber los Naturales, Cardinales, Enteros, Racionales, Irracionales, Reales, Complejos.

Conjuntos iguales

Decimos que dos conjuntos A y B son iguales si tienen los mismos elementos, no importa el orden de éstos. La igualdad se representa por A = B .

Subconjuntos

Decimos que A es subconjunto de B si cada elemento del conjunto A es también un elemento del conjunto B, es decir, A está contenido en B, simbolizándose por $\subset$



$A \subset B$

Operaciones con conjuntos

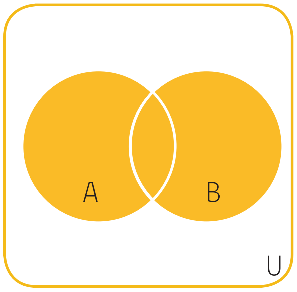
Consideremos dos conjuntos cualquieras, a los cuales llamaremos A y B .

Unión de conjuntos

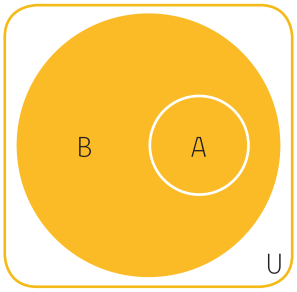
La Unión de los conjuntos A y B es el conjunto formado por todos los elementos de A o B o ambos.
La unión de A y B se representa simbólicamente por $A \cup B$, simbólicamente se escribe:

$$A \cup B = \{ x / x \in A \vee x \in B \}$$

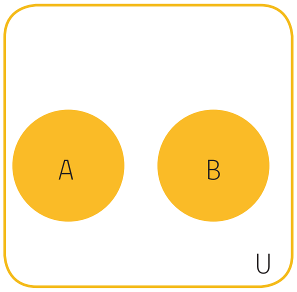
Los siguientes diagramas de Venn muestran gráficamente, que lo achurado representa en cada caso la unión de A y B.



$A \cup B$



$A \subset B\ ,\ A \cup B = B$



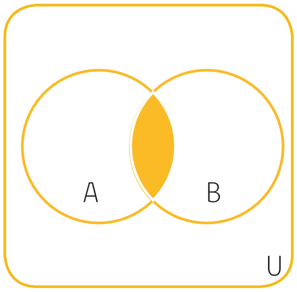
$A \cup B$

Ilustración 4 Diagramas de Venn unión de conjuntos

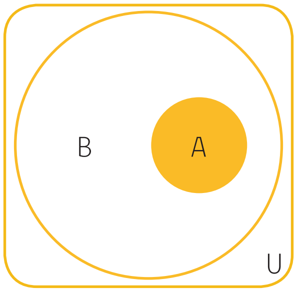
Intersección de conjuntos

La Intersección de los conjuntos A y B se define como el conjunto formado sólo por los elementos que tienen en común A y B.
La intersección se representa por $\mathbf{A \cap B}$, simbólicamente se escribe:

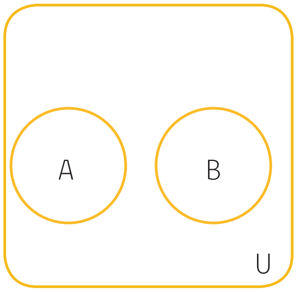
$$A \cap B = \{\, x \,/\, x \in A \wedge x \in B \,\}$$



$A \cap B$



$A \subset B\ ,\ A \cap B = A$



$A \cup B = \emptyset$

Ilustración 5 Diagramas de Venn intersección de conjuntos

Complemento de un conjunto

Se define Complemento de un conjunto de la siguiente forma: sea A un conjunto cualquiera, el complemento de A son todos aquellos elementos que están en el Universo, pero que no están en A.
Simbólicamente, se representa por A^c ó $A`$

$$A^c = \{\, x \in U \,/\, x \notin A \,\}$$

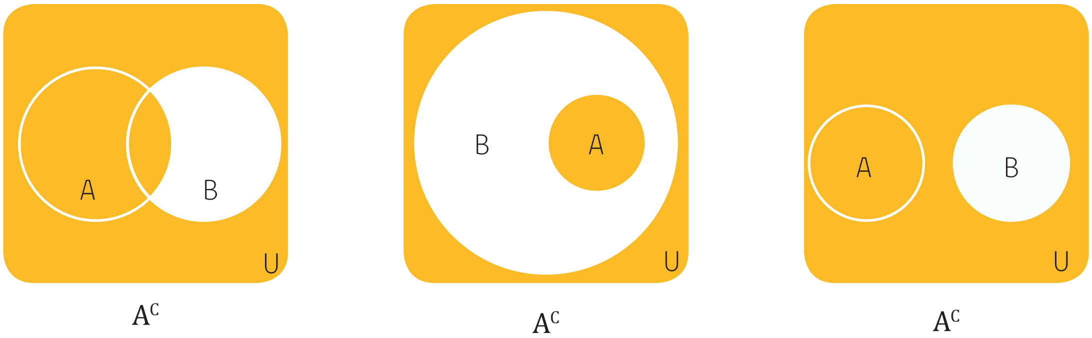


Ilustración 6 Diagramas de Venn complemento de conjuntos

Diferencia de conjuntos

La Diferencia entre dos conjuntos A y B, la cual se denota por A - B, es el conjunto formado por todos lo elementos que están en A y no están en B.
La Diferencia entre B y A la cual se denota por B - A es el conjunto formado por todos los elementos que están en B y no están en A.

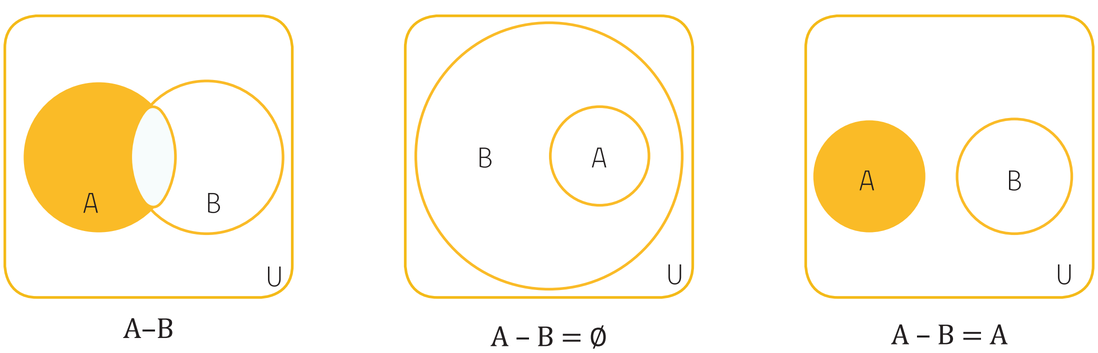
Tip

$A - B \neq B - A$

Ejemplo

Dados los conjuntos
$A = \{1,2,3\}$
$B = \{2,4,6\}$
Calculemos la diferencia de conjuntos
$A - B = \{1,3\}$
$B - A = \{4,6\}$
En el ejemplo anterior se verifica que efectivamente $A - B \neq B - A$
Luego la diferencia se puede expresar

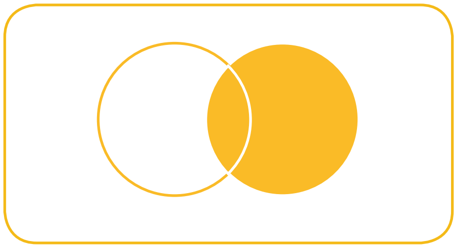
$$A - B = \{ x \in A \wedge x \notin B \}$$

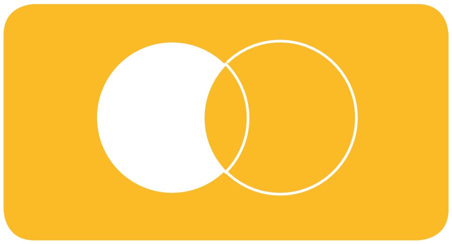


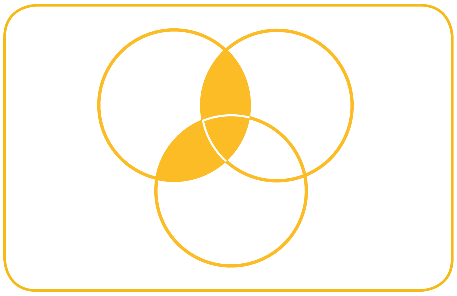
EJERCICIOS PROPUESTOS

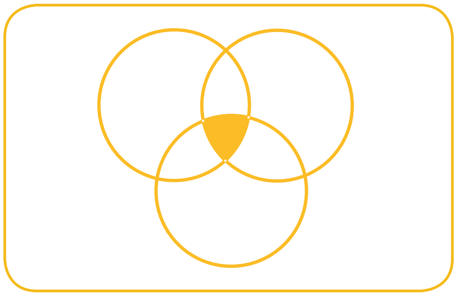
1. Indique en cada caso lo que se achuró

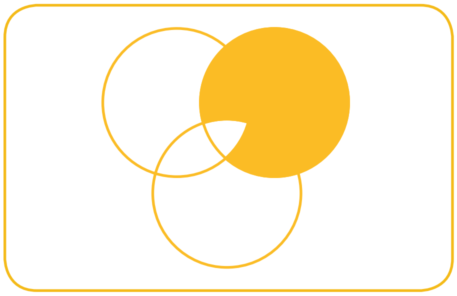
Te invitamos a ver:











2. Achure en el siguiente diagrama de Venn lo que se solicita

a) (A-C)-B
b) $(A \cup B)^2$
c) $A - (B \cap C)^2$
d) $A^c - (B \cap A)^c$
e) $(A \cap B)^c - (C \cap B)$
f) $B \cup (A - C^c)$
g) $(A \cap B \cap C) - A$

Propiedades para resolver problemas en contextos cotidianos

Sean A, B y C tres conjuntos arbitrarios. Definimos la inclusión y la igualdad entre conjuntos, respectivamente, como sigue:

$A \subseteq B \leftrightarrow (\forall x \in U)(x \in A \rightarrow x \in B)$
$A = B \leftrightarrow (A \subseteq B \wedge B \subseteq A)$
$A = A$
$(A = B \text{ y } B = C) \rightarrow A = C$
$(A \subseteq B \wedge B \subseteq C) \rightarrow A \subseteq C$
$\emptyset \subseteq A$
$A \cup B = (x \in A \vee x \in B)$
$A \cap B = (x \in A \wedge x \in B)$

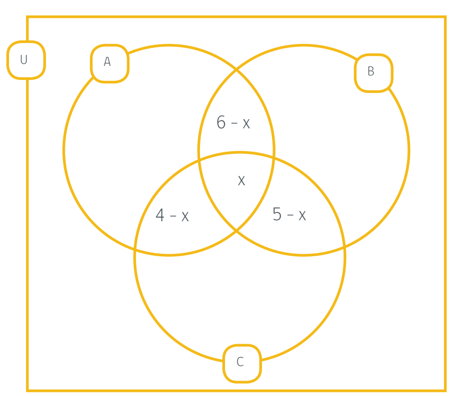
$A \triangle B = \{x \in U : x \in A (\veebar) \quad x \in B\}$
$A x B = \{(a,b) : a \in A \wedge b \in B\}$

Ejemplo

En el Mall plaza el trébol Talcahuano se ofrecen 29 puestos de trabajo, 13 deben ser promotores, 13 vendedores y 15 cajeros. De éstos 6 tienen que ser promotores y vendedores, 4 vendedores y cajeros, y 5 promotores y cajeros. ¿Cuántos tienen que ser las tres cosas a la vez? ¿A cuántas personas que sólo tengan el oficio de promotor se les puede ofrecer empleo?

Solución:

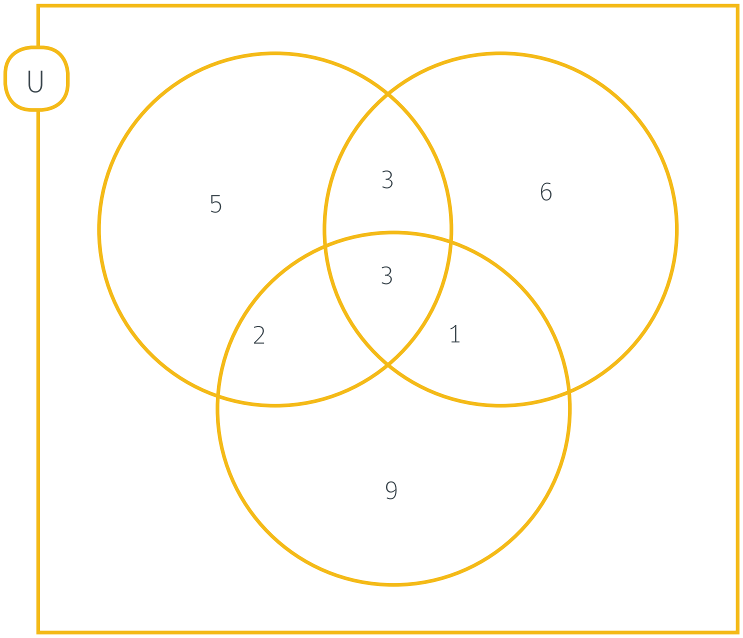
Sea A=promotores ;B=vendedores y C=cajeros, se tiene que:



$|A|=13$
$|B|=13$
$|C|=15$
$|U|=29$
$|A \cap B|=6$
$|A \cap C|=5$
$|B \cap C|=4$
$|A \cap B \cap C|=x$

Como en total se ofrecen 29 puestos de trabajo, se aplica la siguiente fórmula de la teoría de conjunto.
$|A \cup B \cup C|=|A|+|B|+|C|-|A \cap B|-|A \cap C|-|B \cap C|+|A \cap B \cap C|$

Reemplazando:
$29=13+13+15-6-5-4+x$
$29=26+x$
$x=3$



Respuestas:
3 personas deben cumplir la labor de promotor, vendedor y cajero. A 5 personas se les debe contratar solamente como promotor.

Ejemplo:

Recientemente los estudiantes de la UCSC están empleando el uso de tecnologías para complementar su aprendizaje. Se realizó una encuesta sobre el tipo de aparato tecnológico que prefieren, los resultados fueron los siguientes: 60 prefieren notebook, 25 prefieren tablet, 10 prefieren smartphone, 2 prefieren los tres aparatos, 10 prefieren notebook y tablet, 4 prefieren tablet y smartphone, 4 ninguno, 70 no prefieren smartphone. ¿Cuántos prefieren notebook y smartphone, pero no Tablet?

Solución:

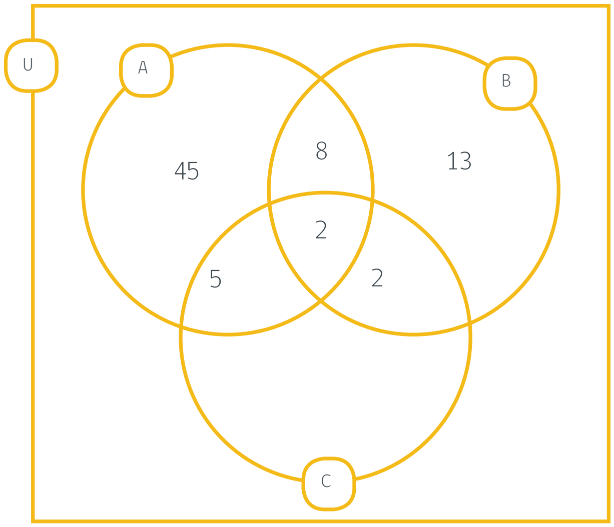
Sea U={Estudiantes de la UCSC}
A= {Estudiante que prefieren notebook}
B= {Estudiantes que prefieren tablet}
C= {Estudiantes que prefieren smarphone}

$|A|=60$ $\quad$ $|A\cap B|=10$
$|B|=25$ $\quad$ $|B\cap C|=4$
$|C|=10$ $\quad$ $|A\cup B\cup C|^c=4$
$|A\cap B\cap C|=2$ $\quad$ $|C|^c=70$

Según los datos presentados el diagrama de Venn queda de la siguiente manera:
Ahora como nos dicen que hay 70 estudiantes que no prefieren smartphone, debemos restarle a este valor los 8 estudiantes que solamente prefieren notebook y tablet, los 13 estudiantes que solamente prefieren tablet y los 4 estudiantes que no utilizan ninguna de estas tecnologías, tal como se muestra a continuación:

70-8-13-4=45

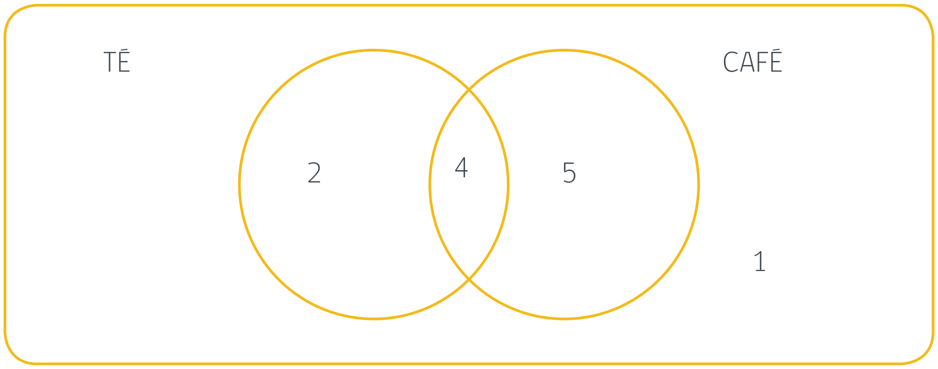
Estos 45 estudiantes restantes solamente prefieren utilizar el notebook, tal como se expresa en el diagrama de Venn de más abajo:



Por consiguiente el área achurada corresponde a las personas que solamente prefieren notebook y smartphone pero no tablet las cuales son 5 estudiantes, ya que sumando los 45 estudiantes que solamente prefieren notebook con los 8 estudiantes que solamente prefieren notebook y tablet y los 2 estudiantes que prefieren notebook, tablet y smartphone, se obtiene como resultado 55 estudiantes y en el problema inicial nos indican que existen 60 estudiante que prefieren notebook, por ende en el área achurada están esos 5 estudiantes. Tal como se muestra en la siguiente figura.

EJERCICIOS PROPUESTOS

1. En el diagrama que colocamos a continuación, se han volcado los datos obtenidos en una encuesta, realizada a personas, a las que se les preguntó si tomaban té o café. Los números que aparecen se refieren a las cantidades de personas que respondieron a la pregunta en las diversas formas posibles: solamente té, té y café, ninguna de las dos bebidas, etc.



a) ¿Cuántas personas tomaban té?
b) ¿Cuántas personas tomaban café?
c) ¿Cuántas personas tomaban té y café?
d) ¿Cuántas personas no tomaban ninguna de las dos bebidas?
e) ¿Cuántas personas no tomaban té?
f) ¿Cuántas personas no tomaban café?
g) ¿Cuántas personas tomaban por lo menos una de esas dos bebidas?
h) ¿Cuántas personas tomaban sólo una de esas dos bebidas?
i) ¿Cuántas personas tomaban sólo café?
j) ¿Cuántas personas tomaban alguna de esas bebidas?

2. Se investigó un grupo de 5500 personas en relación con la estrategia a seguir con objeto de conservar el combustible. De éstas, 2000 opinaron que lo aceptable era el racionamiento, 1500 dijeron que lo apropiado sería fijar un impuesto adicional por litro, y 750 personas indicaron que lo apropiado sería la aplicación de ambos procedimientos. El resto de las personas no aceptan ninguno de los dos sistemas. Determinar:

a) Desarrolle un diagrama de Venn, que resuma lo anterior.
b) ¿Cuántas personas aceptarían en forma voluntaria el racionamiento pero no el impuesto?
c) ¿Cuántas personas aceptarían en forma voluntaria el impuesto, pero no el racionamiento?
d) ¿Cuántas personas no aceptarían en forma voluntaria ninguno de los cursos de acción?

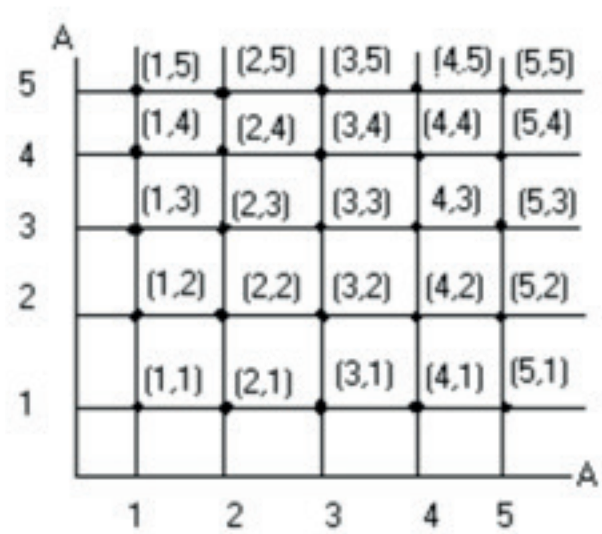
3. Se realizó una encuesta a 200 alumnos de Ingeniería en Ejecución en diversas disciplinas acerca de la forma en que ocupaban su tiempo libre, 30 dicen que sólo leen, 60 dicen que solamente escuchan música, 20 dicen que sólo estudian, 16 dicen que leen y escuchan música, 50 dicen que estudian, 16 dicen que escuchan música y estudian y 8 hacen las tres cosas. De acuerdo a la encuesta, responda:
a) Grafique la información.
b) ¿Cuántos sólo leen o estudian?
c) ¿De los que opinan, cuántos dicen que no leen?
d) ¿Cuántas personas no contestan alguna de estas tres alternativas?
e) ¿Cuántas personas escuchan música, pero no leen?
f) ¿Cuántas personas estudian y escuchan música, pero no leen?

Funciones

Relaciones

Establecer relaciones entre varios tipos de fenómenos es fundamental para hacer predicciones en todo ámbito, por ejemplo, la correspondencia entre un estudiante y su número de matrícula es una relación. Considere el conjunto $A = \{1, 2, 3, 4, 5\}$ la representación gráfica del producto AXA que se llama Producto Cartesiano, y se lee "A cruz A" se hace mediante un diagrama cartesiano, como se ve en la figura:

Ilustración 8 Producto cartesiano del conjunto A



Cada elemento de $A X A$ es un par ordenado de la forma (a,b). Al primer elemento del par ordenado lo llamaremos “abscisa” o “pre imagen” y al segundo elemento “ordenada” o “imagen”.
Matemáticamente una relación es una correspondencia entre un primer conjunto llamado Dominio y un segundo conjunto llamado Codominio, de modo que a cada elemento del dominio corresponde uno o más elementos del codominio. En general, una relación es cualquier conjunto de pares ordenados de números reales.

Propiedades del producto cartesiano:

Para todo conjunto A, B y C cualquiera se tiene:

Asociatividad

$A \times (B \times C) = (A \times B) \times C$

Distributividad respecto de la intersección

$A \times (B \cap C) = (A \times B) \cap (A \times C)$

Distributividad respecto de la unión

$A \times (B \cup C) = (A \times B) \cup (A \times C)$

El producto cartesiano no es conmutativo

$A \times B \napprox B \times A$

Te invitamos a ver:

Ejemplos

Sea $A = \{1,2,3\}$ determine las siguientes relaciones:

a) $R_1 = \{(a,b) \in A \times A \ / \ a \preccurlyeq b\}$

$R_1 = \{(1,1), (1,2), (1,3), (2,2), (2,3), (3,3)\}$

b) $R_2 = \{(a,b) \in A \times A \ / \ a \succ b\}$

$R_2 = \{(2,1), (3,1), (3,2)\}$

c) $R_3 = \{(a,b) \in A \times A \ / \ a + 1 = b\}$

$R_3 = \{(1,2), (2,3)\}$

EJERCICIOS PROPUESTOS

Sea $A = \{1,2,3\}$ determine las siguientes relaciones:

$R = \{(a,b) \in A \times A \ / \ a + 2 \preccurlyeq 3\}$

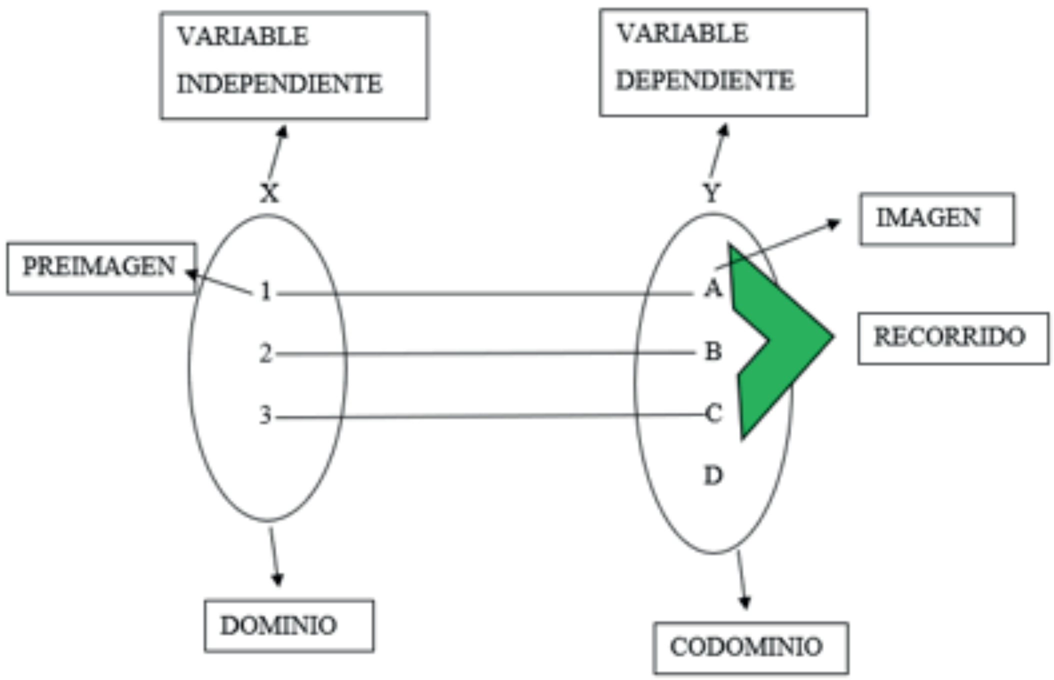
$R = \{(a,b) \in A \times A \ / \ a + 2 \preccurlyeq 3\}$

$R = \{(a,b) \in A \times A \ / \ a = 1\}$

$R = \{(a,b) \in A \times A \ / \ b = 4\}$

Elementos básicos de una función

Ilustración 9 Diagrama de flechas de una función



Tip

Si el codominio de una función real no está señalado, se asume que es el conjunto de los números reales.
En la figura presentada se aprecia que el recorrido sólo son los elementos A, B y C, ya que son estos elementos que se relacionan con los valores que toman la variable independiente, mientras que el codominio es todo el conjunto de la variable independiente.
Ahora bien, reiteradamente se ha mencionado el concepto de variable por lo cual es preciso definir ésta.

Variable

Una variable es un objeto tangible o intangible que cambia, como, por ejemplo, el clima, la temperatura, el precio, etc.
En una función existe la variable independiente y la variable dependiente, la primera es relevante, ya que ésta, a medida que va tomando valores produce cambios en la variable dependiente por ejemplo el tiempo realiza diversos cambios en distintos objetos. Un ejemplo de estas variables, sería la distancia que puede recorrer un automóvil dependerá de la cantidad de combustible que éste posea. A continuación, se presenta la forma algebraica en que se representa una función:

$f:A\rightarrow B$
$x\longmapsto y=f(x)$

- $\{x\epsilon A:\exists y\in B, y=f(x)\}\subseteq A$ se llama dominio de la función, y se denota por $\mathrm{Dom}(f)$.
- B se llama codominio de la función y se denota por $B=\mathrm{Cod}(f)$
- $\{y\in B:\exists x\in A, y=f(x)\}\subseteq B$ se llama recorrido de la función, y se denota por $\mathrm{Rec}(f)$.
- $y=f(x)$ se llama la imagen de x por f o variable dependiente
- x se llama variable de una función o variable independiente.

Ejemplos

Determine el Dominio y Recorrido de las siguientes relaciones:

1. $R1 = \{(1,2), (2,3)\}$
   $\mathrm{Dom}\ R1 = \{1, 2\}$
   $\mathrm{Rec}\ R1 = \{2, 3\}$

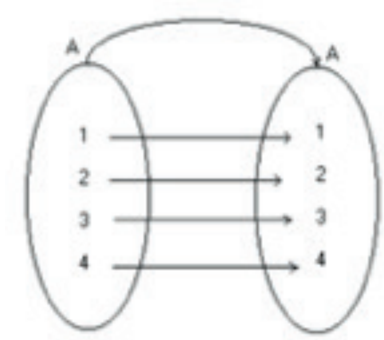
2. $R2 = \{(a,b), (a,c), (a,d)\}$
   $\mathrm{Dom}\ R2 = \{a\}$
   $\mathrm{Rec}\ R2 = \{b, c, d\}$

3. $R3 = \{(1,2), (2,1)\}$
   $\mathrm{Dom}\ R3 = \{1, 2\}$
   $\mathrm{Rec}\ R3 = \{1, 2\}$

Te invitamos a ver:

Ejemplo

Para el siguiente diagrama de flechas determinar la relación.

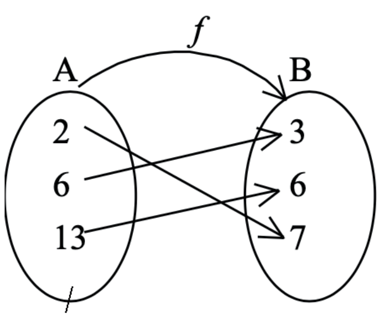


A = { 1,2,3,4 } y R ={ (a,b) ∈ A x A / a = b}

Ejemplos

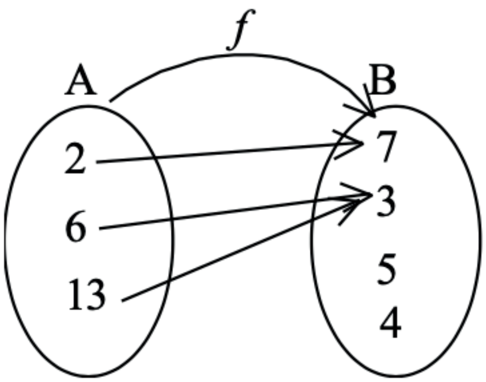
Determine si las siguientes relaciones son función e indique sus dominios.

a)



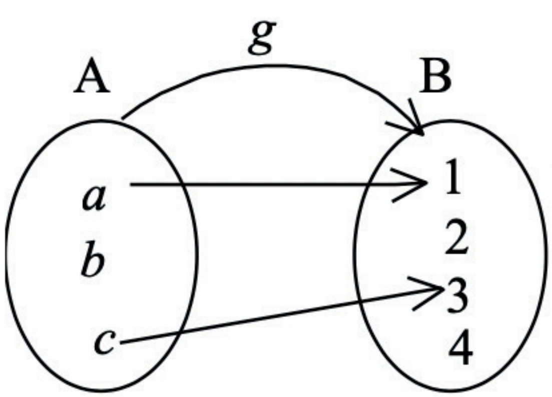
f(2) = 7
f(6) = 3
f(13) = 6
Dom f = {2, 6, 13}

b)



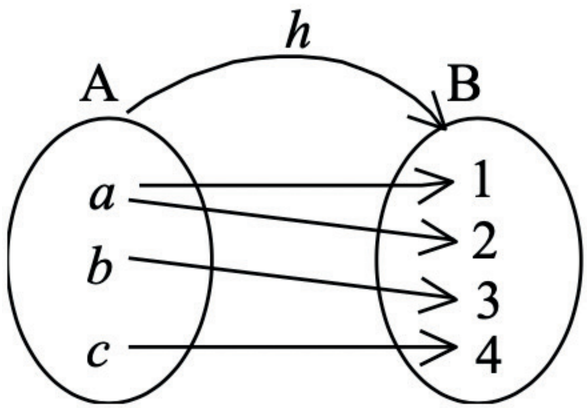
$f(2) = 7$
$f(6) = 3$
$f(13) = 3$
$\text{Dom } f = \{2, 6, 13\}$

c)



g no es función ya que b no tiene imagen, por lo que no cumple con la condición de que cada elemento del conjunto de partida debe ser parte del Dominio de la función.

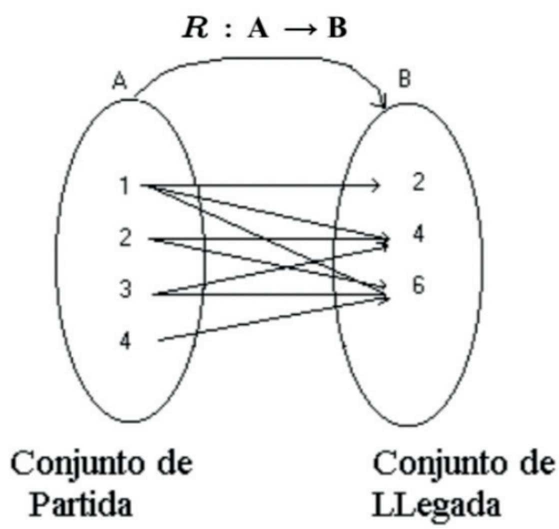
d)



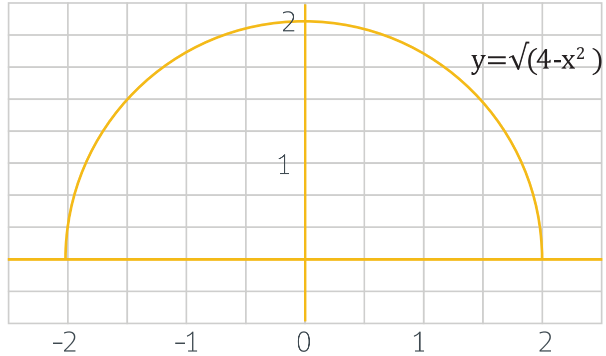
h no es función ya que a tiene dos imágenes y no cumple con la condición de unicidad en las imágenes.

EJERCICIOS PROPUESTOS

1. Escriba la relación existente de A y B



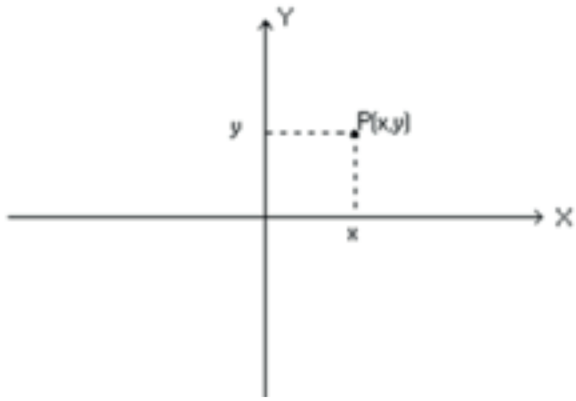
2. La grafica de la función $y=\sqrt{(4-x^2)}$ es una semi circunferencia con centro en el origen y radio 2, determine su dominio y recorrido.



Plano cartesiano

El Plano Cartesiano está formado por dos rectas perpendiculares entre sí llamada cada una Eje, y que se intersectan en un par común llamado Origen, el cuál (0,0).
Cada par (x,y) se llama punto (x,y).

Ilustración 10 Plano Cartesiano

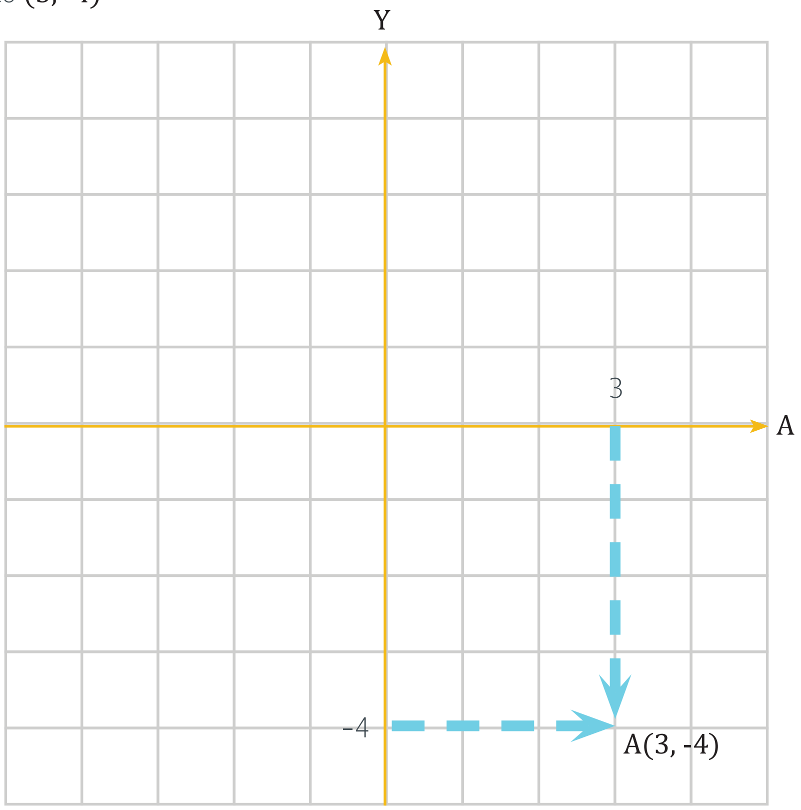


El eje de las abscisas es la recta Horizontal

El eje de las Ordenadas es la recta Vertical

Ejemplo

Grafiquemos el punto (3, -4)



EJERCICIOS PROPUESTOS

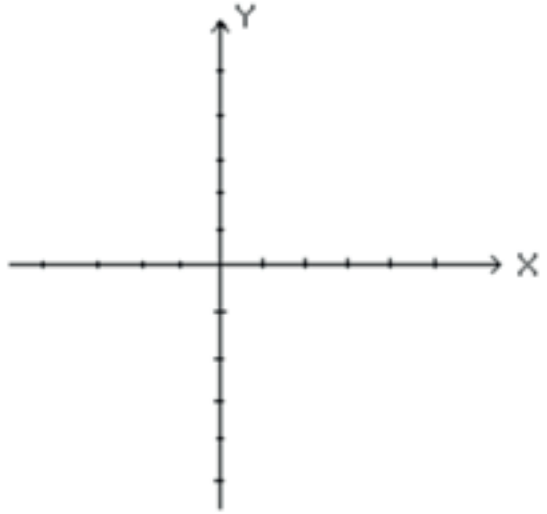
Grafique los siguientes puntos en el plano cartesiano

A) (1,-1)

B) (-3,0)

C) (-2,2)

D) (0,-2)

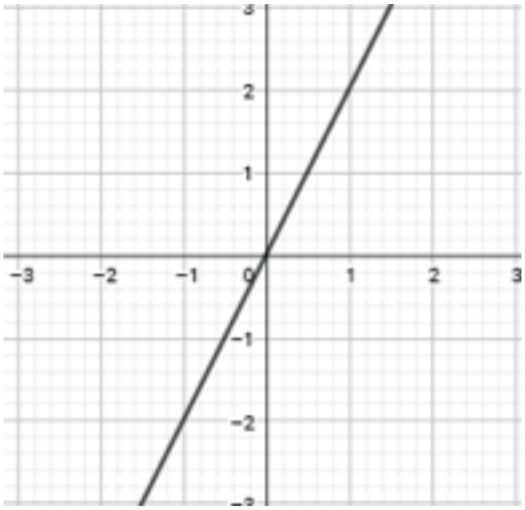


Tipos de Funciones

Función lineal y afín

Este tipo de funciones corresponden a rectas que pasan por el origen o rectas que cortan al eje y.

Ilustración 11 Ejemplo de gráfica función lineal que pasa por el origen y = 2x



$f:\text{dom}(f) \subseteq R \rightarrow R$

$x \longmapsto f(x) = 2x$

$\text{Dom}(f) = \{x \in R : f(x) \in R\}$

$\text{Rec}(f) = \{y \in R : \exists x \in \text{Dom}(f) \wedge f(x) = y\}$

Tip

En resumen, tanto la función lineal como la función afín tienen su dominio y recorrido en el conjunto de los números reales. La forma más común de obtener la gráfica de estas funciones es generando una tabla de valores que incluyan pre imágenes e imágenes.

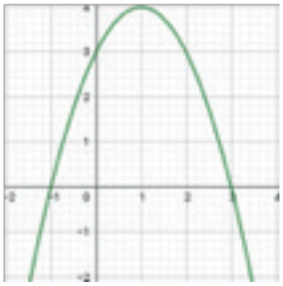
Función cuadrática

Este tipo de función, tiene la siguiente forma algebraica: $f(x)=ax^2+bx+c$, donde cada parámetro cumple las siguientes funciones.

Ilustración 12: Concavidad de una función cuadrática

Si $a>0$ entonces se tiene una parábola cóncava hacia arriba. $f(x)=x^2+2x-3$ (en este caso $a=1$), el parámetro c determina donde la parábola corta al eje y, en este caso $c=-3$

Si $a<0$ entonces se tiene una parábola cóncava hacia abajo. $f(x)=-x^2+2x+3$ (en este caso $a=-1$ y $c=3$)



Las raíces (puntos de corte del eje x) de la ecuación de segundo grado se obtienen mediante:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

En donde b^2-4ac se llama discriminante y nos otorga importante información como por ejemplo la cantidad de raíces que tendrá una función cuadrática.

$f: Dom(f) \subseteq R \rightarrow R$
$x \longmapsto f(x) = -x^2+2x+3$
$Dom(f) = \{x \in R : f(x) \in R\}$
$Rec(f) = \{ y \in]-\infty,4] : \exists x \in Dom(f) \wedge f(x) = y \}$

$$\left(\frac{-b}{2a}, f\left(\frac{-b}{2a}\right)\right)$$

Tip

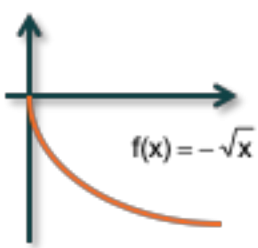
Es importante recalcar que el dominio de la función cuadrática siempre será el conjunto de los números reales y el recorrido dependerá del vértice de ésta y su orientación, es decir si la parábola abre hacia arriba o hacia abajo.

Función raíz cuadrada:

Es de la forma $f(x)=\sqrt{x}$, con $x\geq 0$; Su representación gráfica es la mitad superior de una parábola que empieza en el origen y se abre hacia la derecha.

Ilustración 13 Representación gráfica de la función raíz cuadrada

$\mathrm{Dom}(f) = \{x \in [0,+\infty[: f(x) \in R_{0+} \}$
$\mathrm{Rec}(f) = \{y \in [0,+\infty[: \exists x \in \mathrm{Dom}(f) \wedge f(x)=y\}$



$\mathrm{Dom}(f) = \{x \in [0,+\infty[: f(x) \in R_{0-} \}$
$\mathrm{Rec}(f) = \{y \in]-\infty,0] : \exists x \in \mathrm{Dom}(f) \wedge f(x) = y\}$

Si $f(x)= \sqrt{x-h}+k$, con h y k en los reales, entonces
$\mathrm{Dom}(f)=[h,+\infty[$ y $\mathrm{Rec}(f)= [k,+\infty[$

Tip

La restricción fundamental para encontrar el dominio de la función raíz cuadrada, es que la cantidad subradical (expresión algebraica que está dentro de la raíz) debe ser mayor o igual que cero.

Esta igualdad $\sqrt{(x^2)}=|x|$ se cumple, ya que "x", puede tomar valores tanto positivos como negativos y siempre se llegará al mismo resultado.

Ejemplo

Con $x=-2$ y $x=2$,se tiene:

$\sqrt{(-2)^2} = \sqrt{4} = 2$

$\sqrt{(2)^2} = \sqrt{4} = 2$

Función racional

Es de la forma $f(x)=\frac{p(x)}{q(x)}$, donde el denominador debe ser distinto de cero, $(q(x)\neq 0)$, lo cual es fundamental para determinar su dominio y recorrido.

Ejemplo

$$f:X\rightarrow R$$

$$x \mapsto f(x) = \frac{1}{x-7}$$

$$Dom(f) = \{x - 7 \neq 0: f(x) \in R\}$$

$$Dom(f) = \{x \neq 7: f(x) \in R\}$$

$$Dom(f) = \{x \in R - \{7\}: f(x) \in R\}$$

$$Rec(f) = \left\{y = \frac{1}{x-7} : \exists x \in Dom(f) \wedge f(x) = y\right\}$$

$$Rec(f) = \{y * (x-7) = 1 : \exists x \in Dom(f) \wedge f(x) = y\}$$

$$Rec(f) = \left\{(x-7) = \frac{1}{y} : \exists x \in Dom(f) \wedge f(x) = y\right\}$$

$$Rec(f) = \left\{(x = \frac{1}{y} + 7 : \exists x \in Dom(f) \wedge f(x) = y\right\}$$

$$Rec(f) = \{(y \neq 0: \exists x \in Dom(f) \wedge f(x) = y\}$$

$$Rec(f) = \{(y \in R - \{0\} : \exists x \in Dom(f) \wedge f(x) = y\}$$

Tip

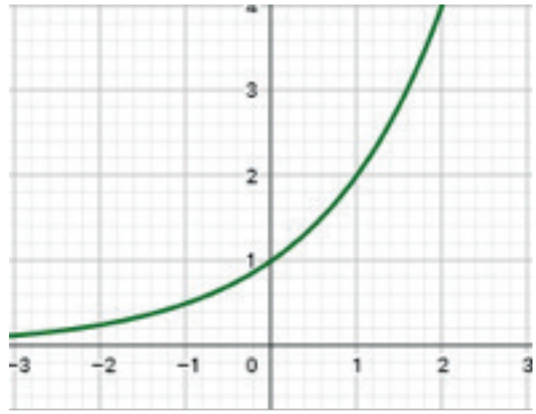
Restricción principal para el cálculo del dominio de la función racional, es que el denominador de la función debe ser distinto de cero.

Función exponencial:

Se define como $f(x)=a^x$.

Si $a>1$, la función es creciente.

$Dom(f) =\{ x \in R: f(x) \in R\}$
$Rec(f) =\{ y \in]0,+\infty [:\exists\ x \in Dom(f) \wedge f(x)=y \}$



Si $0<a<1$, la función es decreciente
$Dom(f)=\{x\in R:f(x)\in R\}$
$Rec(f)=\{y \in]-\infty,0[:\exists x \in Dom\ (f) \wedge f(x)=y\}$

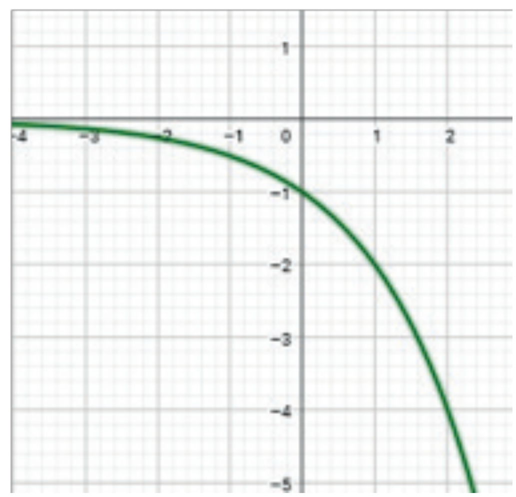


Ilustración 14 Representación gráfica de la función exponencial

Es importante mencionar que en la función exponencial siempre **"a"** debe ser mayor que cero, por ende, este parámetro siempre va tomar valores pertenecientes a los números reales positivos. A continuación, se presentan algunas propiedades de las potencias que serán útiles para resolver algunos problemas asociados a la función exponencial.

Propiedades de las potencias

Con, a y b números reales.

$$a^0 = 1, con\ a \neq 0$$

$$(a^n)^m = a^{nm}$$

$$a^1 = a$$

$$a^n b^n = (ab)^n$$

$$a^n a^m = a^{n+m}$$

$$\frac{a^n}{b^n} = \left(\frac{a}{b}\right)^n, con\ b \neq 0$$

$$\frac{a^n}{a^m} = a^{n-m}, con\ a \neq 0$$

$$a^{-n} = \frac{1}{a^n}, con\ a \neq 0$$

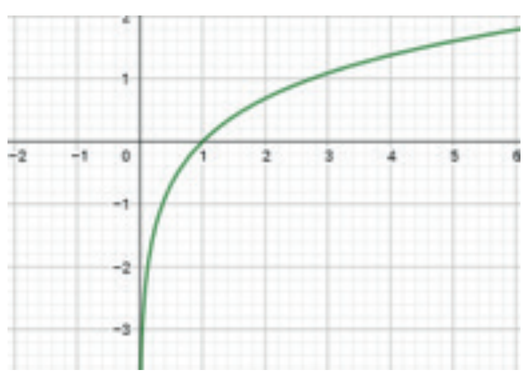
$$\left(\frac{a}{b}\right)^{-n} = \left(\frac{b}{a}\right)^n, con\ a\ y\ b\ \neq$$

Función logarítmica

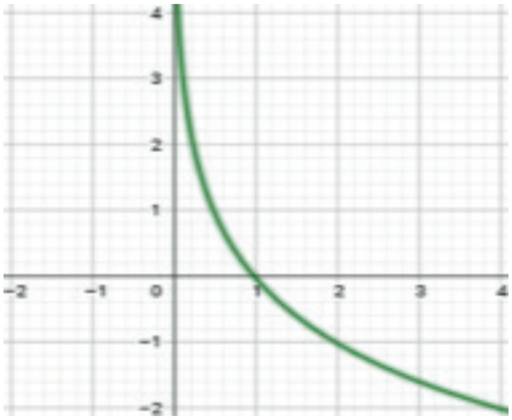
Se define de la forma $f(x)=\log_a (x)$, con $a>0$ y $a\neq1$

Ilustración 15 Representación gráfica de la función exponencial

Si a>1, la función es creciente
$Dom(f) = \{x \in R_+ : f(x) \in R\}$
$Rec(f) = \{y \in R : \exists\ x \in Dom(f) \wedge f(x) = y\}$



Si $0<a<1$, la función es decreciente
$Dom(f) = \{x \in R_+ : f(x) \in R\}$
$Rec(f) = \{ y \in R : \exists\, x \in Dom(f) \wedge f(x)=y\}$



Definición de logaritmo: $\log_b a = x \leftrightarrow b^x=a$, esto quiere decir que la base elevada a x es igual al argumento, ejemplo: $\log_{10} 100=2 \leftrightarrow 10^2=100.$

Propiedades de los logaritmos

Con $a>0$ y $a\neq1$
$\log_a (1)=0$
$\log_a (a)=1$
$\log_a (xy)=\log_a (x)+\log_a (y)$

$\log_a \left(\frac{x}{y}\right)=\log_a (x)-\log_a (y)$

$\log_a (x)^n=n*\log_a (x)$

$\log_a (x)= \frac{\log_b(x)}{\log_b(a)}$,propiedad del cambio de base.

Notaciones importantes:

$\log_{10}=\log$
$\log_e=\ln$

Tip

Para encontrar el dominio de la función logarítmica, siempre el argumento debe ser mayor que cero, es la restricción principal que se debe utilizar.

Ejemplo

$f(x) = \log(x-1)$

Restricción para encontrar el dominio:

$x-1>0$

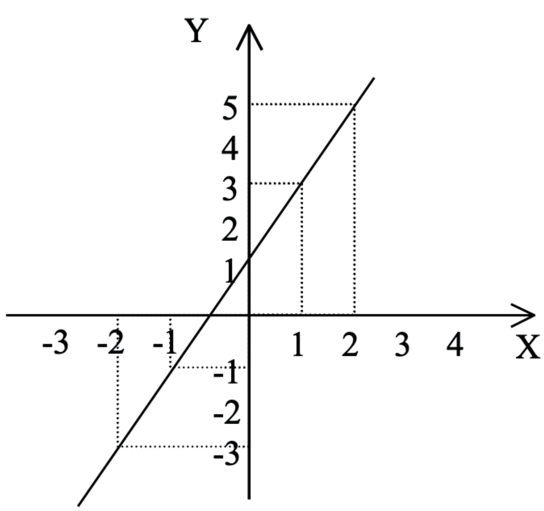
$x>1$

Ejemplo

Graficar $y = f(x) = 2x + 1$ para los puntos conformados por $x = \{-2,-1,0,1,2\}$

$$y = f(x) = 2x + 1 \quad \begin{cases} f(-2) = 2 \cdot (-2) + 1 = -4 + 1 = -3 \\ f(-1) = 2 \cdot (-1) + 1 = -2 + 1 = -1 \\ f(0) = 2 \cdot 0 + 1 = 0 + 1 = 1 \\ f(1) = 2 \cdot 1 + 1 = 2 + 1 = 3 \\ f(2) = 2 \cdot 2 + 1 = 4 + 1 = 5 \end{cases}$$

x	$y = f(x)$
-2	-3
-1	-1
0	1
1	3
2	5



Ejemplo

Graficar la función $y = x^2 z - x - 6$

Identificando los parámetros a, b y c de la ecuación, tenemos: a = 1, $b = -1$ y $c = -6$

Calculemos Δ discriminante

$\Delta = (-1)^2 - 4 \cdot 1 \cdot (-6) = 1 + 24 = 25$

Calculemos ahora las raíces de la parábola.

$x^2 - x - 6 = 0$

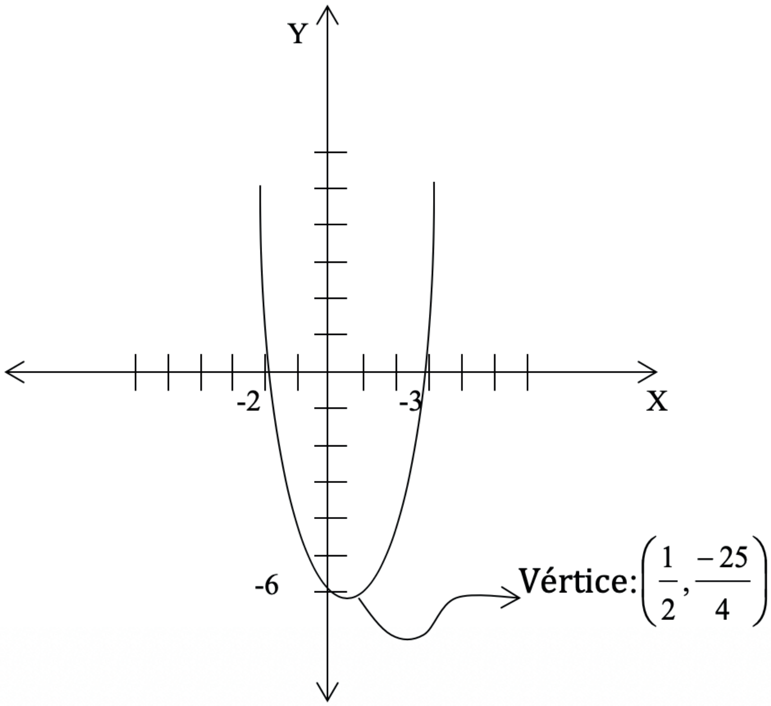
$(x - 3)(x + 2) = 0$

$x1 = -2$ y $x2 = 3$

Como $a > 0$ y $\Delta > 0$, entonces la parábola tiene ramas hacia arriba y corta al eje X en dos puntos, $(-2; 0)$ y $(3; 0)$. Además, la parábola corta al eje Y en el punto $(0; c) = (0; -6)$. Ubiquemos ahora el vértice de la parábola

$$x_v = \frac{-B}{2A} = \frac{-(-1)}{2 \cdot 1} = \frac{1}{2} \quad ; \quad y_v = C - \frac{B^2}{4A} = -6 - \frac{(-1)^2}{4 \cdot 1} = \frac{-25}{4}$$

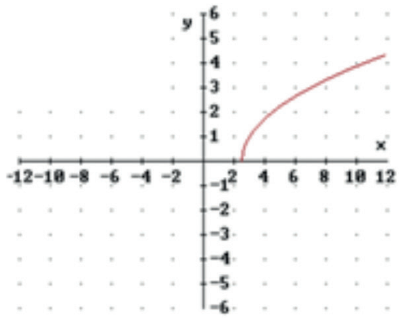
Gráfico de la función:



Ejemplo

Graficar la función $f(x)=\sqrt{(2x-5)}$ y determinar su dominio y recorrido.
Si desarrollamos la desigualdad $2x-5\geq 0$ que asegura la existencia de la función raíz cuadrada encontramos que $x\geq 2.5$

x	f(x)
2,5	0
3	1
4	1.7
5	2.2
6	2.6



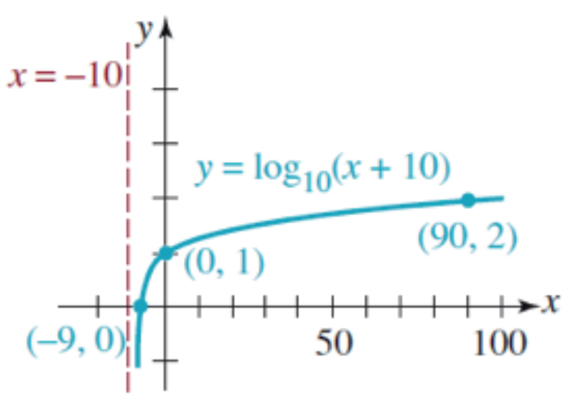
D: [2.5, +∞[R: [0, +∞[

Ejemplo

Graficar $f(x)=\log_{10}(x+10)$

La gráfica de $f(x)$ se encuentra desplazada en 10 unidades hacia la izquierda. Verificamos que el dominio de la función es el conjunto de los números reales, calculamos el dominio con el requisito que $x+10>0$, que indica el desplazamiento de 10 unidades hacia la izquerda ya que $x>-10$. Luego para un esbozo mas acabado de la grafica creaos una tabla de valores de x y evaluamos la función.

x	-9	0	90
$f(x)$	0	1	2



Notar que $x=-10$ es una asíntota vertical de la función en estudio.

EJERCICIOS PROPUESTOS

Graficar las siguientes funciones:

a) $f(x) = 7$

b) $g(x) = 2x - 6$

c) $f(x) = \frac{4}{3}x - 1$

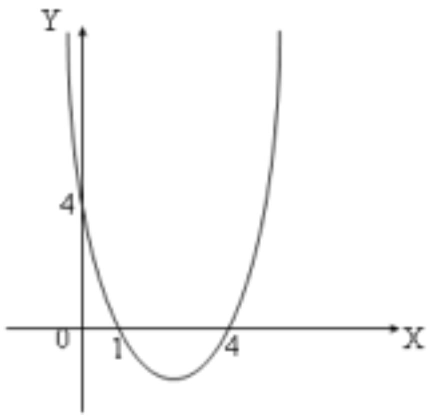
d) $f(x) = \frac{x-1}{3}$

e) $y = 3x - 1$

f) $\mathbf{2x + 3y - 2 = 0}$

EJERCICIOS PROPUESTOS

Determine la función del siguiente gráfico



Determine la función de segundo grado que tiene por soluciones $x1$ y $x2$ tales que $x1 + x2 =$ y $x1/x2 =$

Graficar la función $y=\sqrt{(2\text{-}3x)}$ y determinar su dominio y recorrido

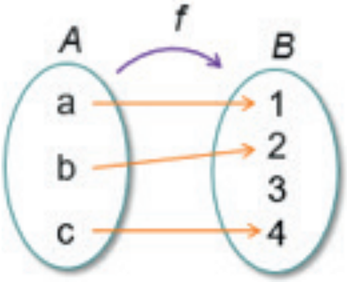
Grafique la función $f(x)=\log_2 (3\text{-}x)$, determoine su dominio y asíntota vertical.

Inyectividad, sobreyectividad y biyectividad de funciones reales.

Inyectividad

Sea f una función que va de A hasta B, f es inyectiva si se cumple que cada elemento del recorrido se relacione con un único elemento del dominio, es decir que cada imagen se relaciona con una única preimagen, por ende, se cumple que: $f(a)=f(b) \rightarrow a=b$

Ilustración 16 Inyectividad

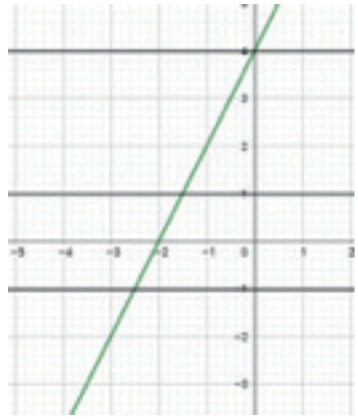


Te invitamos a ver:

Por otro lado, también se puede determinar la Inyectividad de una función a través de su gráfica, a continuación, se presenta un ejemplo que detalla lo expuesto.

Ejemplo

Sea la función $f(x)=2x+4$ cuya gráfica es:



Determinamos su Inyectividad a través de su gráfica trazando rectas paralelas al eje "x", si se visualiza que estas rectas paralelas cortan en un solo punto a la gráfica, entonces se concluye que la función es inyectiva.

Sobreyectividad

Sea f una función que va de A hasta B, f es sobreyectiva (epiyectiva) si se cumple que el codominio es igual al recorrido. Como se aprecia en la siguiente figura.

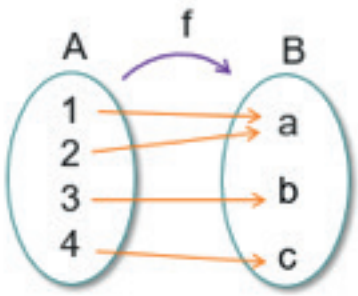


Ilustración 17 Sobreyectividad

Rec (f): {a,b,c}
Cod(f): {a,b,c}

Luego se cumple que el **cod(f)=Rec(f)**

Biyectividad

Para que una función sea biyectiva se debe cumplir que ésta sea inyectiva y sobreyectiva. A continuación, se presenta un ejemplo de una función biyectiva.

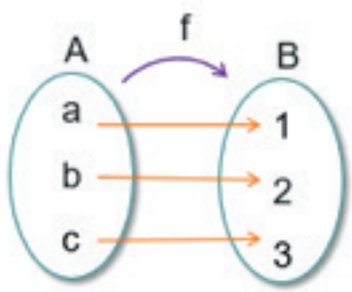


Ilustración 18 Biyectividad

En este caso se cumple que cada imagen se relaciona con una única preimagen (inyectiva) y además el codominio es igual al recorrido. Por ende, f es biyectiva.

Sea f una función que va de A hasta B. Para que f sea biyectiva se debe cumplir que sea inyectiva y sobreyectiva (o epiyectiva) a la vez.

Función inversa

Sea **f** una función que va de **A hasta B**. Si f es biyectiva, entonces existe la función inversa de **f**. en caso contrario no.

$f:A \to B$, entonces $f^{(-1)}:B \to A$

$a \longmapsto f(a)=b$ $\quad :b \mapsto f^{(-1)}(b)=a$

$Dom\ f^{(-1)}=Recf=B$ y $Recf^{(-1)}=Domf=A$

Para determinar la función inversa de una función biyectiva se debe expresar la variable independiente (x) en función de la variable dependiente (y), para luego intercambiar las variables. Por ejemplo:

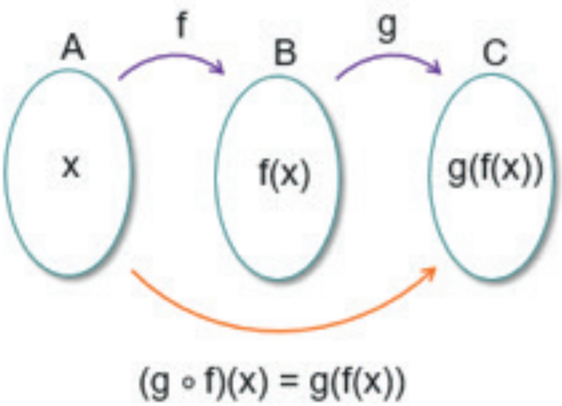
$$f(x) = \frac{2-x}{3x} \to x = \frac{2}{3y+1} \to f^{-1}(x) = \frac{2}{3x+1}$$

Composición de funciones

Sean f y g dos funciones tales que: $f:A \to B$ $g:B \to C$, Entonces, se define "**g compuesta con f**" como una función definida de **A hasta C**.

$gof:A \to C$

Ilustración 19 Composición de funciones



Tip

En general para que exista la composición de funciones debe ocurrir que el recorrido de la segunda función este contenido en el dominio de la primera función.

Ejemplo

Sea una función f, cuyo dominio es el **conjunto {1,2,3},** definida por $f(x)=x-1$, sea una la función g, con dominio el **conjunto {0,1,2,3}**, definida por $g(x)=x+1$. Se puede dar **fog y gof.**

Solución: Para determinar si puede existir **fog**, debemos determinar el recorrido de **g** y verificar que este contenido en el dominio de **f**, por lo cual reemplazaremos los elementos del domino de **g** en **g(x)**:

$g(0)=1$
$g(1)=2$
$g(2)=3$
$g(3)=4$

Entonces se tiene que el recorrido de $g(x):\{1,2,3,4\}$, este conjunto no está contenido en el dominio de f, ya que este es {1,2,3}. Por ende, no puede existir fog. Por otro lado, gof, sí existe, ya que el recorrido de f(x):{0,1,2}, está contenido en el dominio de **g** que es $\{0,1,2,3\}$.

$gof = g(\ f(x)\ ) = (x\text{-}1) + 1 = x$

Ejemplo

Considere la siguiente función:

$f{:}Dom(f) \subseteq R \to R$

$$x \longmapsto f(x)=\sqrt{(2\text{-}x)}+3$$

Determine el Dominio y recorrido de f
Determine si f es biyectiva, si no lo es, redefínala para que lo sea
Determine la inversa de la función biyectiva definida en el ítem anterior

Solución:

a) $dom(f)=]\text{-}\infty,2]$
$Rec(f) = [3,+\infty[$

b) Para comprobar que **f** es inyectiva aplicaremos el siguiente teorema:

Si $f(x_1)= f(x_2) \to x_1=x_2$, con $x_1,x_2 \in]\text{-}\infty,2]$ por ende, se tiene que:

$$f(x_1) = f(x_2) \to \sqrt{2-x_1} + 3 = \sqrt{2-x_2} + 3$$

$$\sqrt{2-x_1} = \sqrt{2-x_2}$$

$$x_1 = x_2$$

Por lo tanto **f** es inyectiva.

La función no es sobreyectiva, ya que el $Rec(f) = [3,+\infty[\neq Cod(f)=R$, luego **f** no es biyectiva, por ende, se debe redefinir la función, restringiendo el codominio:

$f\hat{}{:}]\text{-}\infty,2] \to [3,+\infty[$
$x \longmapsto f(x) = \sqrt{(2\text{-}x)}+3$

Una vez que se ha restringido el codominio, **f** cumple con la biyectividad.

La inversa de f es:

$f^{(-1)}\hat{}{:}\ ]3,+\infty] \to [\text{-}\infty,2\ [$
$x \longmapsto f^{(-1)}\ (x)=\text{-}(x\text{-}3)2+2$

Ejemplo

Considere las siguientes funciones:

$f:Dom(f)\subseteq R\to R$ $\qquad$ $g:Dom(g)\subseteq R\to R$

$x\longmapsto f(x)=\dfrac{1}{\sqrt{27-x}}$ $\qquad$ $x\longmapsto g(x)=3^{(x-1)}$

a) Determine el dominio de f
b) Determine el dominio de g
c) Defina (fog)

Solución:
a) $Dom(f) =]-\infty,27[$
b) $Dom(g)=R$
c) $Dom(fog)=\{x \in R : 3^{(x-1)} \in]-\infty, 27[\} =]-\infty,4[$

Luego se tiene que:

$fog:]-\infty,4[\subseteq R\to R$

$x\longmapsto f(g(x))=\dfrac{1}{\sqrt{27-3^{x-1}}}$

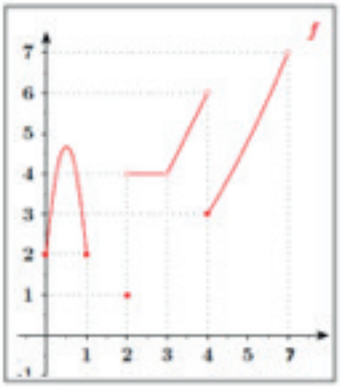
Ejemplo

Determine el dominio de la función: $f(x)=\dfrac{log_2(x+3)+log_5(5-x)}{\dfrac{1}{e\sqrt{x^2-1}}}$

Solución: $Dom(f) = \{x \in R:f(x)\in R\}$
$= \{x \in R:x+3>0\wedge 5-x>0\wedge x^2-1>0\}$
$= \{x \in R:x>-3\wedge x<5\wedge |x|>1\}$
$= \{x \in R:x>-3\wedge x<5\wedge (x>1\vee x<-1\}$
$=]-3,-1[\cup]1,5[$

Ejemplo

Encuentre el dominio y recorrido de la función $f:D\subset R\to R$, cuya gráfica es:



Solución: $Dom(f)=[0,1]\cup[2,7[-\{3\}$
$Rec(f) =[2,7[\cup\{1\}$

Ejemplo

Encuentre el dominio y recorrido de la función: $f(x)=\frac{3}{\sqrt{x-4}}$

Solución:

$$\begin{aligned} Dom(f) &= \{x \in R: f(x) \in R\} \\ &= \{x \in R: x-4>0\} \\ &= \{x \in R: x>4\} \\ &=]4,+\infty[\end{aligned}$$

$$\begin{aligned} Rec(f) &= \{y \in R: \exists x \in Dom(f), y=f(x)\} \\ &= \{y \in R: \exists x \in\ ]4,+\infty[,\ \frac{3}{\sqrt{x-4}}\} \\ &= \{y \in R: \exists x \in\ ]4,+\infty[, x=\frac{9}{y^2}+4^y>0\} \\ &= \{y \in R: y>0\} \\ &=]0,+\infty[\end{aligned}$$

EJERCICIOS PROPUESTOS

Sea $f: R \to R$ definida por $f(x)=2x-1$, ¿Es f una función biyectiva? Si es así determine su inversa.
Sea $f: R \to R$ definida por $f(x)=4x-2$, ¿Es f una función biyectiva? Si es así determine su inversa.
Sea $f:]-2,+\infty[\to R_0^-$ definida por $f(x)=\sqrt{(5x+10)}$, ¿Es f una función biyectiva? Si es así determine su inversa.

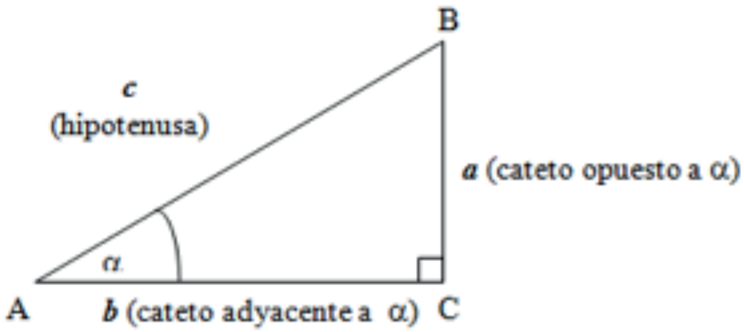
Trigonometría

La trigonometría trata la medición de ángulos mediante relaciones entre trazos. Estas relaciones se llaman razones trigonométricas.

Razones trigonométricas en el triángulo rectángulo

Sea $\Delta\ ABC$ rectángulo en C de catetos a y b y de hipotenusa c

Ilustración 20 Razones trigonométricas en un triángulo rectángulo



Se definen las razones trigonométricas de la siguiente manera:

$$\text{Seno de } \alpha = \text{sen } \alpha = \frac{\text{cateto opuesto}}{\text{hipotenusa}} = \frac{a}{c}$$

$$\text{Coseno de } \alpha = \cos \alpha = \frac{\text{cateto adyacente}}{\text{hipotenusa}} = \frac{b}{c}$$

$$\text{Tangente de } \alpha = \text{tg } \alpha = \frac{\text{cateto opuesto}}{\text{cateto adyacente}} = \frac{a}{b}$$

Y sus respectivos recíprocos

$$\text{Cotangente de } \alpha = \text{cotg } \alpha = \frac{1}{tg\ \alpha} = \frac{\text{cateto adyacente}}{\text{cateto opuesto}} = \frac{b}{a}$$

$$\text{Secante de } \alpha = \sec \alpha = \frac{\text{hipotenusa}}{\text{cateto adyacente}} = \frac{c}{b}$$

$$\text{Cosecante de } \alpha = \text{cosec } \alpha = \frac{1}{sen\ \alpha} = \frac{\text{hipotenusa}}{\text{cateto opuesto}} = \frac{c}{a}$$

Tip

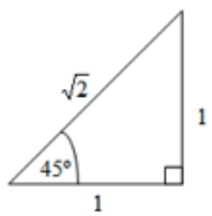
Las razones trigonométricas son cantidades adimensionales, es decir, no tienen dimensiones a pesar de que los lados de un triángulo corresponden a longitudes.

Las razones trigonométricas son números reales y como tales les corresponden todas las operaciones y las propiedades de los números reales, es decir, se pueden sumar, multiplicar, asociar, distribuir con otros números, etc.
Utilicemos ángulos conocidos para ejemplificar el uso de las razones trigonométricas, vemos los siguientes ejemplos.

Ejemplo

Sea un triángulo rectángulo isósceles, de la explicación de las razones trigonométricas podemos inferir que no dependen del tamaño del triángulo, entonces nos daremos un triángulo de lados convenientes para hacer fáciles los cálculos.

1. Sea el triángulo rectángulo isósceles de catetos iguales a 1 y ángulo de 45º.

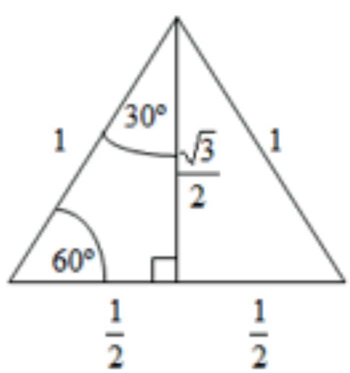


$$\text{sen } 45^\circ = \frac{1}{\sqrt{2}} = \frac{\sqrt{2}}{2} \qquad \cos 45^\circ = \frac{1}{\sqrt{2}} = \frac{\sqrt{2}}{2} \qquad \text{tg } 45^\circ = \frac{1}{1} = 1$$

$$\text{cotg } 45^\circ = \frac{1}{1} = 1 \qquad \sec 45^\circ = \frac{\sqrt{2}}{1} = \sqrt{2} \qquad \text{cosec } 45^\circ = \frac{\sqrt{2}}{1} = \sqrt{2}$$

Ejemplo

Para determinar las razones trigonométricas de los ángulos de 30º y 60º, aprovechemos las relaciones que se dan en un triángulo equilátero de lado 1.



Para el ángulo de 30º:

$$\text{sen}\,30º = \frac{\frac{1}{2}}{1} = \frac{1}{2} \qquad \cos 30º = \frac{\frac{\sqrt{3}}{2}}{1} = \frac{\sqrt{3}}{2} \qquad \text{tg}\,30º = \frac{1}{\text{tg}\,60°} = \frac{1}{\sqrt{3}} = \frac{\sqrt{3}}{3}$$

$$\text{cotg}\ 30º = \frac{\frac{\sqrt{3}}{2}}{\frac{1}{2}} = \sqrt{3} \qquad \sec 30º = \frac{1}{\text{sen}\,60°} = \frac{1}{\frac{\sqrt{3}}{2}} = \frac{2\sqrt{3}}{3}$$

$$\text{cosec}\ 30º = \frac{1}{\cos 60°} = \frac{1}{\frac{1}{2}} = 2$$

EJERCICIO PROPUESTO

a) Realiza los cálculos de las razones trigonométricas para el Angulo de 60º

b) Genera una tabla resumen de las razones trigonométricas para estos ángulos comunes y compáralas con alguna que encuentres en algún teto de matemáticas.

Identidades trigonométricas

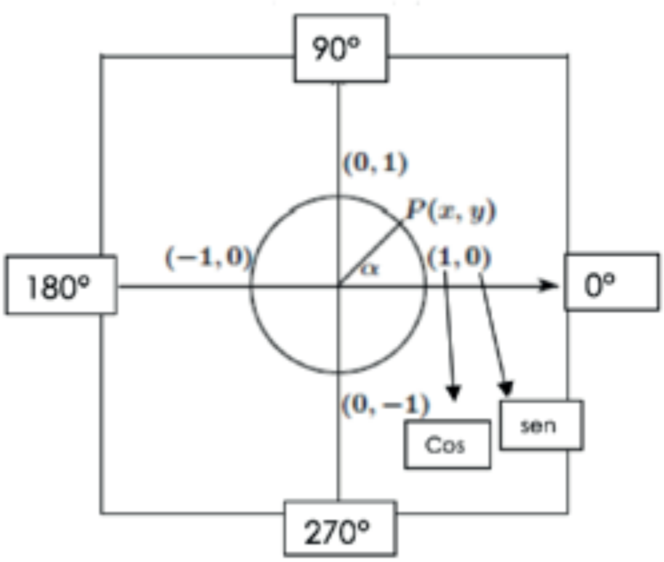
Dada una circunferencia unitaria (de radio 1) $x^2+y^2=1$, se define:

$$x=\cos(\alpha), y=\text{sen}(\alpha)$$

De lo cual se deduce que:

$\cos(0°)=1$	$\cos(90°)=0$	$\cos(180°)=-1$	$\cos(270°)=0$
$\text{sen}(0°)=0$	$\text{sen}(90°)=1$	$\text{sen}(180°)=0$	$\text{sen}(270°)=-1$

Ilustración 21 Circunferencia de radio unitario



Dadas las funciones definidas en una circunferencia unitaria centrada en el origen:

$sen:R\to R$, $sen(x)$ y $cos:R\to R, cos(x)$, se definen:

Tangente:

$D\subset R\to R$

$\alpha\longmapsto y=\tan 3(\alpha)$

con $\tan(\alpha)=\dfrac{sen(\alpha)}{\cos(\alpha)}$ $y\ D=R-\left\{\dfrac{(2k-1)\pi}{2}\right\}, k\epsilon Z$

Cotangente:

$D\subset R\to R$

$\alpha\longmapsto y=\cot(y)=\dfrac{1}{\tan(\alpha)}$

donde $D=R-\{k\pi\},\ k\epsilon Z$

Identidades Trigonométricas importantes

$sen^2(x)+\cos^2(x)=1$

$1+\tan^2(x)=\sec^2(x)$

$1+\cot^2(x)=\csc^2(x)$

$sen(x\pm y)=sen(x)*\cos(y)\pm sen(y)\cos(x)$

$\cos(x\pm y)=\cos(x)*\cos(y)\mp sen(x)\,sen(y)$

$\tan(x\pm y)=\dfrac{\tan(x)\pm\tan(y)}{1\mp\tan(x)*\tan(y)}$

$sen(2x)=2sen(x)*\cos(x)$

$\cos(2x)=\cos^2(x)-sen^2(x)$

$\cos(2x)=2\cos^2(x)-1$

$\cos(2x)=1-2sen^2(x)$

$\tan(2x)=\dfrac{2\tan(x)}{1-tan^2(x)}$

$sen\left(\dfrac{x}{2}\right)=\pm\sqrt{\dfrac{1-\cos(x)}{2}}$

$\cos\left(\dfrac{x}{2}\right)=\pm\sqrt{\dfrac{1+\cos(x)}{2}}$

Inversas de funciones trigonómetricas

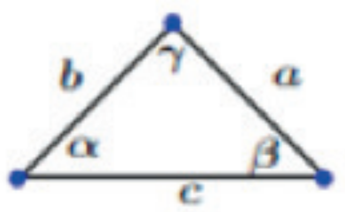
arcsen: $[-1,1] \rightarrow \left[\frac{-\pi}{2}, \frac{\pi}{2}\right]$,$x \rightarrow \text{arcsen}(x)$

arccos: $[-1,1] \rightarrow [0,\pi], x \rightarrow \text{arccos}(x)$

arctan:$\mathbb{R} \rightarrow \left[\frac{-\pi}{2}, \frac{\pi}{2}\right]$,$x \rightarrow \text{arctan}(x)$

Teorema del seno y del coseno

Dado un triángulo cualquiera, tenemos:



Ley del Coseno:

$a^2 = b^2+c^2 - 2bc * \cos(a)$
$b^2 = a^2+c^2 - 2ac * \cos(b)$
$c^2 = a^2+b^2 - 2ab * \cos(y)$

Ley del seno:

$$\frac{sen(a)}{a} = \frac{sen(b)}{b} = \frac{sen(y)}{c}$$

Curvas Sinusoidales

Forma algebraíca de las curvas sinusoidales: $f(x)=A\,sen(bx+c)+d$

Descripción de los parámetros:
- Desplazamiento vertical: d
- Amplitud: $|A|$
- Fase: solución de $bx+c=0$

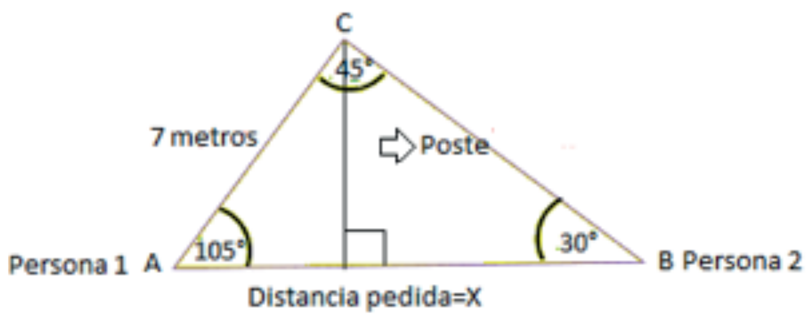
- Periodo: $P=\frac{2\pi}{b}$

Ejemplo

Trigonometría, Ley del seno

Dos personas se encuentran en lados opuestos de un poste con un ángulo de elevación de $75°$ y $60°$ respectivamente. Si la distancia entre el primero y el extremo superior del poste es de 7 metros, determine. la distancia entre ambas personas.

Para resolver este enunciado debemos realizar una representación de la la situación expuesta.



La información que nos entrega la figura, nos permite determinar que a través del teorema del seno podemos encontrar la distancia pedida.

$$\frac{sen(30°)}{7} = \frac{sen(45°)}{x}$$

$$\frac{\frac{1}{2}}{7} = \frac{\frac{\sqrt{2}}{2}}{x}$$

$x=7\sqrt{2}$, luego la distancia solicitada es $7\sqrt{2}$ metros.

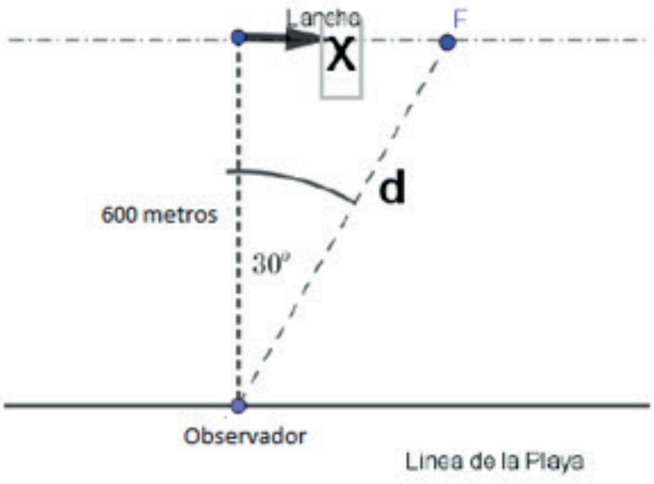
Ejemplo

Una lancha que viaja en forma paralela a una playa recta (de Norte a Sur), pasa a las 2 p.m. a 600 metros exactamente enfrente de un observador de la guardia costera que se encuentra en la orilla de la playa. Después de 5 minutos el mismo observador, ubicado en la misma posición de la playa, observa a la lancha con un ángulo de 30° Medidos de Oeste a Norte. Determine la velocidad de la lancha en kilómetros por hora.

Solución: Para resolver este enunciado debemos realizar una representación de la la situación expuesta.

Te invitamos a ver:

Te invitamos a ver:



Tenemos que $\cos(30°)=\frac{600}{d}$ y $\text{sen}(30°)=\frac{x}{d}$, luego $d=\frac{1200}{\sqrt{3}}$ y $x=\frac{d}{2}$, por ende $x=\frac{600}{\sqrt{3}}$.

Por ende, la lancha en 5 minutos recorre una distancia de $\frac{600}{\sqrt{3}}$ metros y en 1 hora recorre una distancia de $\frac{7200}{\sqrt{3}}$ metros, lo que equivale a $\frac{7.2}{\sqrt{3}}$ kilómetros. Aplicando la siguiente fórmula de física $V=\frac{d}{t}$, tenemos que la velocidad de la lancha en una hora es de $\frac{7.2}{\sqrt{3}}$ km/hr.

Ejemplo

La ecuación $\cos(x)*(1-\sqrt{3}\ \tan(x))=0$, con $x\in[0,2\pi]$

Solución:

Como la expresión algebraíca esta igualada a 0 y además esta factorizada, tenemos que:

$\cos(x)=0$ y $1-\sqrt{3}\ \tan(x)=0$, pero como $\tan(x)=\frac{sen(x)}{\cos(x)}$, entoces $\cos(x)\neq 0$

Por ende: $1-\sqrt{3}\ \tan(x)=0$

$\tan(x)=1/\sqrt{3}$, de lo cual se deduce que: $x=\frac{1}{6}\pi; x=\frac{7}{6}\pi$

Finalmente el conjunto solución es $x\in\left\{\frac{1}{6}\pi,\frac{7}{6}\pi\right\}$

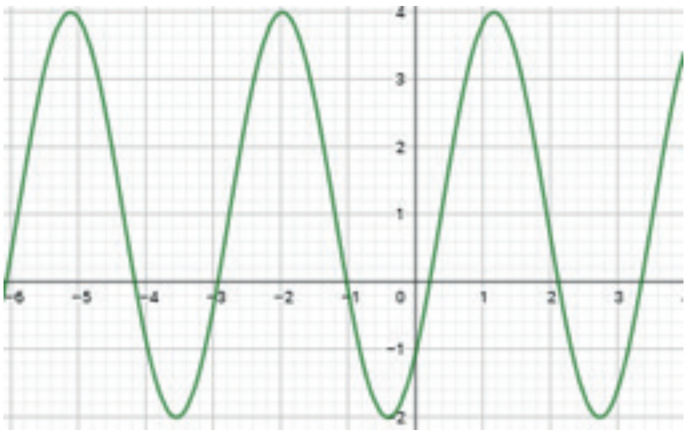
Ejemplo

Considere la siguiente función sinusoidal: $f: R \to R$

$$x \longmapsto f(x) = 3\text{sen}\left(2x - \frac{\pi}{4}\right) + 1$$

Determine la amplitud, período y desfase de **f**. Esboze su gráfica.

Solución:
Para resolver este enunciado debes recordar las fórmulas expuetas anteriormente en la página 34.
Amplitud: $A=3$
Período: $P=\pi$
Desface: $2x - \frac{\pi}{4} = 0 \to x = \frac{\pi}{8}$



EJERCICIOS PROPUESTOS

a) Sea la función $y=3\ \text{sen}\left(\frac{\pi}{2}\right)x$, determine su grafica.

b) Sea la función $y=2\ \text{sen}\ (2x-\pi)$, determine su grafica.

Una torre forma un ángulo de 113 o con el plano inclinado sobre el cual está y desde una distancia de 89 m de su base medida hacia abajo del plano se ve la torre bajo un ángulo de 23 o. Calcular la altura de la torre.

Dos boyas están apartada por una distancia de 64,2 m y un bote está a 74,1 m de la más cercana. El ángulo que forman las dos visuales del bote a las boyas es de27. ¿Qué distancia hay del bote a la boya más alejada?

Números Complejos

Un número de la forma $z=a+bi$, en que $a,b \in \mathbb{R}$ se llama número complejo y pertenece al Conjunto de los números complejos $\mathbb{C}$.

z: es el número complejo, $z \in \mathbb{C}$
a: es la parte real
bi: es la parte imaginaria
i: es la unidad imaginaria, $i=\sqrt{(-1)}$

Tip

Si $a=0$ el número Z es un complejo puro
Si $b=0$ el número Z es real y no complejo
Luego, el conjunto $\mathbb{C}$ se define por:

$$\mathbb{C} = \{ x + iy \ / \ x, y \in \mathbb{R} \}$$

$$i = \sqrt{-1} = i$$
$$i^2 = -1$$
$$i^3 = -\sqrt{-1} = -i$$
$$i^4 = 1$$

y en general $\forall\ n \in \mathbb{Z}$

$$i^{4n-3} = \sqrt{-1} = i$$
$$i^{4n-2} = -1$$
$$i^{4n-1} = -\sqrt{-1} = -i$$
$$i^{4n} = 1$$

El conjunto de los números imaginarios es cíclico, siempre se repiten los mismos resultados (i,-1,-i y 1). Por ende, para obtener cualquier potencia de **i**, se debe tomar el exponente del número imaginario y dividirlo por 4, si el resto resultante es 1 entonces el número imaginario será igual a **i**, si el resto obtenido es 2, entonces el número imaginario será igual a -1, si el resto resultante es 3, entonces el número imaginario será igual -i y si el resto obtenido es 0, entonces el número imaginario será igual a 1.

$i^1=i \longrightarrow$ resto 1
$i^2=-1 \longrightarrow$ resto 2
$i^3=-i \longrightarrow$ resto 3
$i^4=1 \longrightarrow$ resto 0

Ejemplo

Reducir hasta la mínima expresión utilizando las potencias de la unidad imaginaria

i^2-i^3= -1+i

$3i^4+2i-i^5-4 = 3*1+2i-i^3\ i^2-4 = 3+2i-(-1)(-i)-4=-1+i$

: i^{343}: 343:4=85 por tanto i^{343}=-i

EJERCICIOS PROPUESTOS

Reducir hasta la mínima expresión utilizando las potencias de la unidad imaginaria

$i^{(-1)}+2i^{73}-i^{(-35)}$

$i^{37}+i^{126}$

Representación gráfica: El plano complejo

Sea el número complejo z=x+yi, z puede representarse gráficamente por el punto T de coordenadas rectangulares (x,y).

El punto O, de coordenadas (0,0) representa el complejo $0 + 0i$. Todos los puntos del eje X tienen coordenadas de la forma (x,0) y corresponden a números reales $x + 0i = x$. Por tal razón se llama al eje X eje de los reales o eje real.

Todos los puntos del eje Y tienen coordenadas de la forma (0, y) corresponden a números imaginarios puros $0 + yi$. El eje Y se llama por eso eje de los imaginarios o eje imaginario.

El plano en que se representan los números complejos se llama plano complejo.

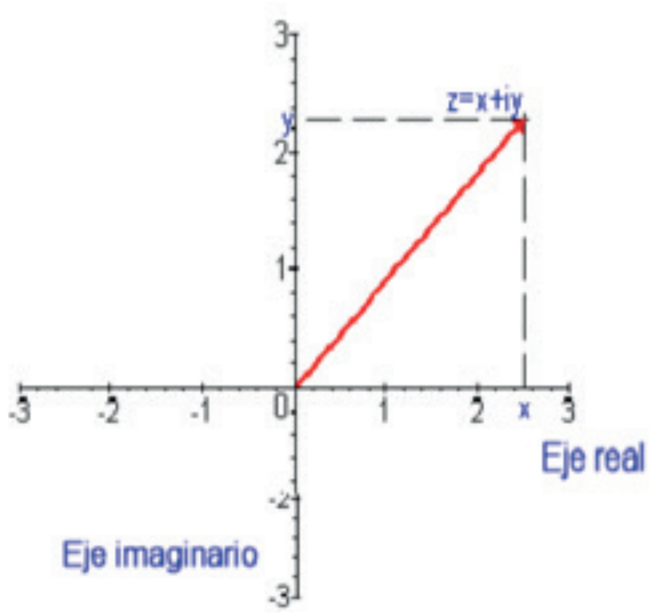
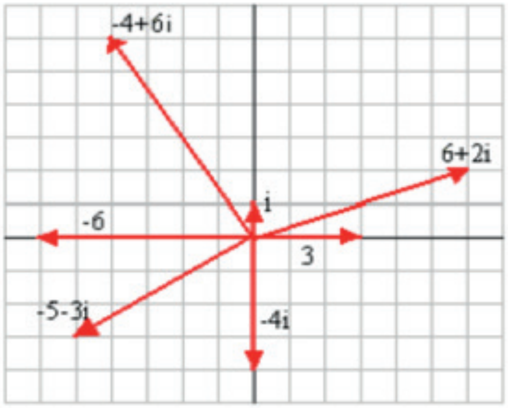


Ilustración 23 Plano complejo

Ejemplo
Represente gráficamente los siguientes números complejos
a) $-4+6i$
b) $6+2i$
c) 3
d) i
e) $-5-3i$
f) $-4i$
g) -6



Conjugado de un número complejo

El conjugado del número complejo $z = a + bi$ es $\bar{z}=a-bi$ (se lee el conjugado de zeta), y para obtenerlo se cambia el signo de la parte imaginaria del número complejo original)

Propiedades de los conjugados

a) $\bar{\bar{z}} = z$

b) $\overline{z_1 + z_2} = \overline{z_1} + \overline{z_2}$

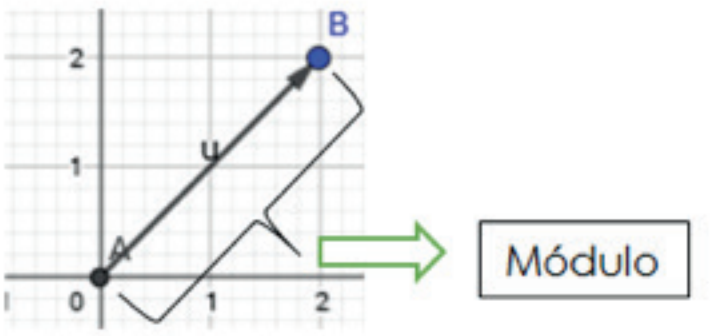
c) $\overline{z_1 \cdot z_2} = \overline{z_1} \cdot \overline{z_2}$

d) $z + \bar{z} = 2a + 0i = 2a, \forall\, a \in \mathbb{R}$

e) $z - \bar{z} = 0 + 2bi = 2bi, \forall\, b \in \mathbb{R}$

Módulo

$|z|=\sqrt{(a^2+b^2)}$ (Se lee el modulo de zeta, y es la distancía desde el origen hasta donde llega el complejo en el plano cartesiano)



Propiedades del Módulo

a) $|z| \geq 0$

b) $|z_1 \cdot z_2| = |z_1| \cdot |z_2|$

c) $\left|\frac{z_1}{z_2}\right| = \frac{|z_1|}{|z_2|}$

d) $|z_1 + z_2| \leq |z_1| + |z_2|$

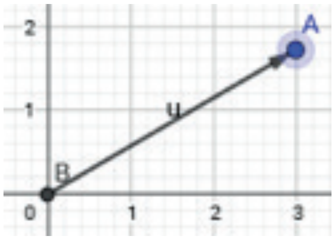
e) $||z_1| - |z_2|| \leq |z_1| - |z_2|$

Representaciones de los números complejos

Un número complejo se puede representar de 4 formas:

- Forma canónica = $3+\sqrt{3}\, i$
- Par ordenado = $(3,\sqrt{3})$
- Representación gráfica
- Forma polar = $2\sqrt{3}\, \text{cis}(30°)$

Ilustración 25 Representación gráfica de un complejo



La forma polar se representa a través de la siguiente fórmula:

$r\text{cis}(\theta)=r(\cos(\theta)+i\text{sen}(\theta))$, donde r es la distancia desde el origen $(0,0)$ hasta el punto final del número complejo en este caso A y θ es el ángulo que forma este número complejo con el eje x y se obtiene aplicando la siguiente función trigonométrica $\theta=\arctan\left(\frac{\sqrt{3}}{3}\right)$ entonces $\theta=30°$.

Operaciones con números complejos

Adición y sustracción

Para sumar o restar dos números complejos se suman o restan las partes reales y las imaginarias por separado.

Veamos el caso de la suma que es análogo a la resta.

$$\begin{aligned}(x+yi)+(a+bi) &= (x+a)+(yi+bi)\\ &= (x+a)+(y+b)i\end{aligned}$$

Ejemplo

$(2+3i)+(4-5i)=(2+4)+(3i-5i)=6+(-2i)=6-2i$

Multiplicación

Para multiplicar dos complejos, se usa la multiplicación de binomios y se utilizan las potencias de complejos vistas anteriormente.

$$\begin{aligned}(x+iy)(a+ib) &= xa+xib+iya+i^2yb\\ &= ax+bxi+ayi+i^2by\\ &= ax+i(bx+ayi)+(-1)by\\ &= (ax-by)+i(bx+ay)\end{aligned}$$

Ejemplo

$$\begin{aligned}(2+3i)(4-5i) &= 8-10i+12i-15i^2\\ &= 8+2i-15(-1)\\ &= 8+2i+15\\ &= 23+2i\end{aligned}$$

División

Para la división de complejos se multiplica el numerador y el denominador por el conjugado del denominador.

$$\frac{(x+iy)}{(a+ib)} = \frac{(x+iy)}{(a+bi)}\frac{(a-bi)}{(a-bi)}$$

$$= \frac{(ax-bxi+ayi-byi^2)}{(a^2-abi+abi-b^2i^2)}$$

$$= \frac{(ax-bi(-1)+i(ay-by)}{(a^2+b^2)}$$

Ejemplo

$$\frac{(2+3i)}{(4-5i)} = \frac{(2+3i)(4+5i)}{(4-5i)(4+5i)}$$

$$= \frac{8+10i+12i+15i^2}{16+20i-20i-25i^2}$$

$$= \frac{8+22i-15}{16+25}$$

$$= \frac{-7+22i}{41}$$

$$= \frac{-7}{41} + \frac{22}{41}i$$

Teorema de Moivre

$z = |z|\ \mathrm{cis}(\theta) \in C,\ n \in N$

Esta fórmula se utiliza para calcular potencias de números complejos, como por ejemplo $(1+i)^{20}$.

$$\sqrt[n]{z} = |z|^{\frac{1}{n}} cis\left(\frac{\theta + 2k\pi}{n}\right), k = 0, \ldots.., n-1$$

Esta fórmula se utiliza para calcular las raíces de un número complejo, como, por ejemplo, $\sqrt[3]{(2\text{-}3i)}$.

Ejercicio

Encuentre las soluciones de la siguiente ecuación para $z \in C$:

$2z^4-4z^2+4=0$

Solución: se procede factorizando por 2

$2(z^4-2z^2+2)=0$, luego queda $z^4-2z^2+2=0$

Completamos el cuadrado:

$(z^2-1)^2+1 = 0$

$(z^2-1)^2 = -1/$ se aplica raíz cuadrada

$|z^2-1| = i \rightarrow z^2-1=i \vee z^2-1=-i$

$z^2 = 1+i \vee z^2=1-i$

$z^2 = \sqrt{2}cis\left(\frac{\pi}{4}\right) \vee z^2 = \sqrt{2}cis\left(-\frac{\pi}{4}\right)$

Se aplica el teorema de Moivre (cálculo de raíces de un complejo)

Caso 1:

$$z_k = \sqrt[4]{2}cis\left(\frac{\frac{\pi}{4}+2k\pi}{2}\right); con\ k = 0,1$$

$$z_0 = \sqrt[4]{2}cis\left(\frac{\pi}{8}\right)$$

$$z_1 = \sqrt[4]{2}cis\left(\frac{\frac{\pi}{4}+2\pi}{2}\right)$$

$$z_1 = \sqrt[4]{2}cis\left(\frac{9\pi}{8}\right)$$

Caso 2:

$$z_k = \sqrt[4]{2}cis\left(\frac{-\frac{\pi}{4}+2k\pi}{2}\right); con\ k = 0,1$$

$$z_0' = \sqrt[4]{2}cis\left(\frac{-\pi}{8}\right)$$

$$z_1' = \sqrt[4]{2}cis\left(\frac{7\pi}{8}\right)$$

Por lo tanto, la solución de las ecuaciones dadas es $\{z_0, z_1, z_0'\ y\ z_1'\}$

EJERCICIOS PROPUESTOS

1. Sean los complejos $Z_1=1+i$ y $Z_2=2-i$ calcule:

   a) $Z_1\,Z_2$

   b) Z_1+Z_2

   c) $\overline{Z}_1$

   d) $\frac{Z_1}{Z_2}$

Resuelva y exprese los resultado en las cuatro formas posibles

   a) $2(3+i)-4(5+i)-7(4-i)$

   b) $(3+2i)^2$

   c) $\frac{2+4i}{4-2i}$

   d) $6-3(5-\frac{2}{5}i)$

Te invitamos a ver:

Polinomios

Un polinomio es una función P definida por la ecuación
$P(x)=a_n x^n+a_{(n-1)} x^{(n-1)}+a_{(n-2)}-x^{(n-2)}+\cdots+a_1 x^1+a_0 x^0$
con a valores constantes y $n \in N$.
El grado del polinomio es la mayor potencia de la variable x.
La operatoria con polinomios fue vista en el primer capítulo de la presente guía "Conceptos fundamentales del álgebra". Luego de haber estudiado con detención dichos aspectos nos abocaremos a la división y a encontrar las raíces.

Descomposición en suma de fracciones parciales.

Definición:
Sean $p(x)$ y $q(x) \in k(x), q(x) \neq 0$, llamaremos fracción racional al cuociente: $\frac{p(x)}{q(x)}$

Ejemplo

$$\frac{2x+4}{3x+5}, con\ x \neq \frac{-5}{3}$$

Se llama fracción propia a la fracción racional $\frac{p(x)}{q(x)}$ donde $gr(\ p(x)\ ) < gr(\ q(x)\ )$. Esto quiere decir que el grado de $p(x)$ es menor que el grado de $q(x)$.

Ejemplo
Fracción propia:

$$\frac{2x+5}{x^2-x-6}$$

si $gr(p(x)) \geq gr(q(x))$, se efectúa la división de modo que:
$\frac{p(x)}{q(x)}=q(x)+\frac{r(x)}{q(x)}$, donde $gr(r(x))<gr(q(x))$ y la fracción $\frac{r(x)}{q(x)}$ es propia.

Teorema de descomposición en suma de fracciones parciales

Cualquier fracción propia $\frac{p(x)}{q(x)}$ con $p(x)$,$d(x) \in R[x]$, se puede descomponer en suma de fracciones parciales como sigue:

Si $q(x)$ tiene un factor lineal de la forma $ax+b$, no repetido, entonces la descomposición en fracciones parciales tiene un término de la forma:

$\frac{A}{ax+b}$,donde A es constante.

Si $q(x)$ tiene un factor lineal de la forma ax+b, repetido k veces, es decir $(ax+b)^k$, entonces la descomposición en fracciones parciales contiene los términos: $\frac{A_1}{ax+b}+\frac{A_2}{(ax+b)^2}+\frac{A_3}{(ax+b)^3}+\cdots+\frac{A_k}{(ax+b)^k}, A_i, i=1,2,\ldots\ldots,k$ son constantes.

Si $q(x)$ tiene un factor cuadrático irreductible de la forma ax^2+bx+c, no repetido, entonces la descomposición en fracciones parciales contiene un término de la forma: $\frac{Ax+b}{ax^2+bx+c}$, A y B son constantes.

Si $q(x)$ tiene un factor cuadrático irreductible en R[x] de la forma ax^2+bx+c, repetido k veces, es decir, $(ax^2+bx+c)^k$, entonces la descomposición en fracciones parciales contiene los términos:

$$\frac{A_1x+B_1}{ax^2+bx+c}+\frac{A_2x+B_2}{(ax^2+bx+c)^2}+\frac{A_3x+B_3}{(ax^3+bx+c)^3}+\cdots+\frac{A_kx+B_k}{(ax^2+bx+c)^k}, A_i \; y \; B_i$$

con $i=1,2,\ldots\ldots,k$ son constantes.

Regla de Ruffini

Esta regla se utiliza para dividir polinomios, siendo el divisor $(x\text{-}c)$, de manera rápida.
Para dividir $p(x)=a_n x^n+a_{(n-1)} x^{(n-1)}+\cdots+a_1 x+a_0$, $p(x) \in p_n (R)$ y $c\in R$. (polinomio) por $(x\text{-}c)$, construimos la siguiente tabla para calcular los b_i, con $i \in \{0,\ldots.,n\text{-}1\}$:

	a_n	a_{n-1}	$\cdots$	a_1	a_0
c					
	$b_{n-1}=a_n$				

En el paso $i\leq 1$, multiplicabamos $b_{(i+1)}$ por c y sumamos el resultado a $a_{(i+1)}$. O sea $b_i=a_{(i+1)}+b_{(i+1)}c$.

	a_n	a_{n-1}	$\cdots$	a_1	a_0
c		$b_{n-1}c$	$\cdots$	b_1c	b_0c
	$b_{n-1}=a_n$	$b_{n-2}=a_{n-1}+b_{n-1}c$	$\cdots$	$b_0=a_1+b_1c$	$r=a_0+b_0c$

Luego, el cuociente de dividir p por x–c es:
$q(x)=b_{(n-1)} x^{(n-1)}+ b_{(n-2)} x^{(n-2)}+\cdots+b_1 x+b_0$.

Además, el último término calculado en la tabla es el resto de dividir p por $(x\text{-}c)$:
$r(x)=r=a_0+b_0c$.
$p(x)=q(x)(x\text{-}c)+r(x)$

Luego, el cuociente de dividir p por x-c es:
$q(x)=b_{(n-1)}\, x^{(n-1)}+ b_{(n-2)}\, x^{(n-2)}+\cdots+b_1\, x+b_0$.
Además, el último término calculado en la tabla es el resto de dividir **p** por (x-c):
$r(x) = r = a_0+b_0\, c$.
$p(x) \;=q(x)\,(x\text{-}c) + r(x)$.

Ejemplo
$p(x)=-2x^3+x^2+x-1:(x+1)$

	-2	1	1	-1	En esta fila se anotan los coeficientes de $p(x)$
-1		2	-3	2	Es el resultado de multiplicar el −1 con la fila 1
	2	3	-2	1	

$\underbrace{2x^2+3x-2}$; $resto = 1$ Es la suma de la primera fila con la segunda.

Cuando se anotan los coeficientes del polinomio, éste debe estar ordenado del grado mayor al menor y si no aparece un grado se considera un coeficiente de cero es decir, $q(x)=x^2+4$, entonces sus grados serían, 1,0 y 4, ya que el polinomio real es $q(x)=x^2+0x+4$.

Teorema fundamental del álgebra

Todo polinomio de grado **n**, con coeficientes complejos, tiene **n** raíces complejas.
Propiedades.

Sea $p\in P(R)$ y $z\in C$ una raíz de p, entonces, el conjugado $\bar{z}$ también es raíz de p.

Sea $p\in P(R)$ y $a+\sqrt{b}$, con $a,b\in R$ y $\sqrt{b}\in I$, una raíz de p, entonces $a-\sqrt{b}$, es también raíz de **p**.

Sea $p(x) \in P(R)$, con coeficientes $a_0,\ldots,a\_n\in Z$ (polinomio con coeficientes enteros). Para calcular las raices racionales de p(x) se determinan todos los divisores **r** de a_0 y todos los divisores s de a_n, luego se forman todos los números r/s posibles y se verifican cuales de ellos son raíces.

Ejemplo
Encuentre las raíces en C de $p(x)=x^5+9x^3-4x^4-18x^2+20x-8$
Las posibles raices son los divisores de 8, es decir, $\{\pm1,\pm2,\pm4,\pm8\}$

	1	-4	9	-18	20	-8
1		1	-3	6	-12	8
	1	-3	6	-12	8	0
1		1	-2	4	-8	
	1	-2	4	-8	0	
2		2	0	8		
	1	0	4	0		

Luego el polinomio se puede factorizar como
$p(x)=(x-1)^2\,(x-2)(x^2+4)=(x-1)^2\,(x-2)(x-2i)(x+2i)$, por lo tanto las raices son $\{1,2,2i,-2i\}$

Ejemplo

Descomponga en suma de fracciones parciales:

$$\frac{x^2-1}{x(x^2+1)}$$

Solución:

$$\frac{x^2-1}{x(x^2+1)}=\frac{A}{x}+\frac{bx+c}{x^2+1}=\frac{A(x^2+1)+x(bx+c)}{x(x^2+1)}$$

$$=\frac{x^2(A+B)+cx+A}{x(x^2+1)}$$

$A+B=1$
$C=0$
$A=-1$, por lo tanto $B=2$

Por consiguiente, la descomposición queda:

$$\frac{x^2-1}{x(x^2+1)}=\frac{-1}{x}+\frac{2x}{x^2+1}$$

EJERCICIOS PROPUESTOS

1. Encuentre las raíces de los siguientes polinomios
a) x^3-7x-6
b) $x^4-5x^3+5x^2+5x-6$
c) x^5-16
d) $3x^3-10x^2+9x-2$
2. Es 2 una raíz de x^4-2x^2-x+7

Bibliografía

Barnett, R. (1998). Álgebra y Trigonometría. México. Editorial McGraw-Hill.

Hernández, E. (1994). Álgebra y Trigonometría. España, Madrid. Addison Wesley Editores.

Leithold, L. (1994). Álgebra y Trigonometría con Geometría Analítica. México. Editorial Harla.

Swokowski, E. (1998). Álgebra y Trigonometría con Geometría Analítica. Madrid. Grupo Editorial Iberoamericano.

ZILL, D. (1997). Álgebra y Trigonometría. Madrid, España. Editorial McGraw-Hill.

Además de estos textos que te recomendamos estudies, nos hemos basado en las guías de estudio elaboradas por la oficina de apoyo docente de la Facultad de Ingeniería UCSC.

¡¡Revisa la bibliografía!!
¿Has consultado los textos que menciona el programa del curso?