

# **CATEDRA DE COMPUTACION II**

## **TEMAS DEL LENGUAJE C++**



### **Tomo 3**

**UNIVERSIDAD NACIONAL DE ENTRE RIOS  
FACULTAD DE INGENIERIA – BIOINGENIERÍA  
2004**



## TABLA DE CONTENIDOS

---

PARTE II OTRAS BIBLIOTECAS .....	5
GRAFICACIÓN CON OPENGL.....	6
Ventana básica.....	6
Puntos, líneas y polígonos (Primitivas OpenGL) .....	9
Puntos .....	9
Líneas.....	9
Polígonos .....	9
Definición de los vértices.....	10
Primitivas.....	10
Rotación/Introducción a las transformaciones .....	13
GRAFICACIÓN CON GNUPLOT .....	21
Introducción .....	21
Primer ejemplo.....	21
Los comandos <i>plot</i> y <i>splot</i> .....	22
Segundo ejemplo.....	23
Graficación de puntos.....	24
Resumen de algunas personalizaciones .....	26
Gráficos de superficie.....	26
Salida de gráfico a archivo .....	27
CÁLCULO NUMÉRICO .....	28
Introducción .....	28
Ejemplo 1: interpolación.....	28
Implementación del programa .....	29

---

Ejemplo 2: solución de ecuaciones no lineales .....	30
Método de bisección .....	30
Método de Newton .....	31
PARTE III ANEXOS .....	33
INTRODUCCIÓN A HTML .....	34
Directivas .....	34
Caracteres Especiales.....	34
Estructura de un documento HTML .....	35
Un documento inicial.....	35
Título.....	35
Cabeceras .....	35
Párrafos.....	36
Texto con formato .....	36
Divisiones horizontales .....	37
Primer ejemplo completo.....	37
Listas .....	38
Imágenes .....	38
Tablas.....	39
Colores.....	40
Segundo ejemplo completo.....	40
DIRECTIVAS DEL PREPROCESADOR.....	42
#define .....	42
#undef.....	42
#ifdef, #ifndef, #if, #endif, #else y #elif.....	43
#ifdef - #ifndef .....	43
#ifdef .....	43
#ifndef .....	43
#line.....	44
#error.....	44
#include.....	45

---

---

#pragma .....	45
<b>BASES NUMÉRICAS .....</b>	<b>46</b>
Números octales (base 8) .....	47
Números hexadecimales (base 16) .....	48
Representaciones binarias .....	49
<b>CÓDIGO ASCII.....</b>	<b>50</b>
<b>LÓGICA BOOLEANA .....</b>	<b>52</b>
Operaciones AND, OR, XOR y NOT .....	52
AND (&) .....	52
OR ( ) .....	52
XOR (Or exclusivo: ^) .....	53
NOT (~) .....	53
<b>ARCHIVOS DE ENCABEZAMIENTO ESTÁNDAR.....</b>	<b>54</b>
<b>LISTADO DE PÁGINAS DEL LENGUAJE EN INTERNET .....</b>	<b>56</b>
Los más recomendados .....	56
Sitios en español .....	56
Sitios en inglés listado por temas .....	57

---

## PARTE II

# OTRAS BIBLIOTECAS

# Graficación con OpenGL

## Ventana básica

OpenGL contiene comandos para dibujo, pero se diseñó para tener independencia de los sistemas operativos, por lo que no posee comandos para abrir ventanas, ni para leer eventos de teclado y/o mouse. Para ello, se usa a **GLUT** (GL Utility Toolkit), cuyas instrucciones comienzan con el prefijo **glut**.

La inicialización de una ventana en OpenGL requiere 5 rutinas básicas:

**glutInit(int \*argc, char \*\*argv)** se encarga de inicializar a GLUT, y debe ser llamada antes de cualquier otra rutina GLUT.

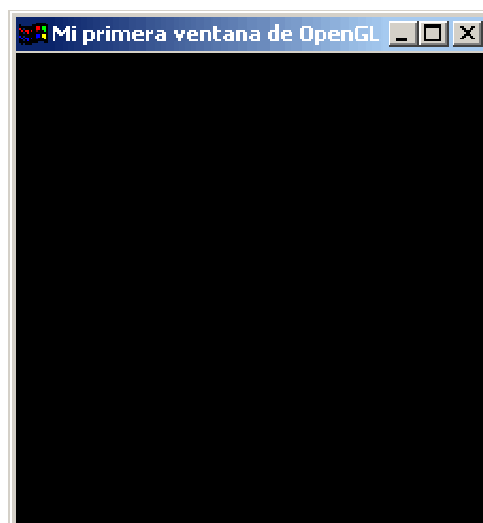
**glutInitDisplayMode(unsigned int mode)** especifica si se usará un modelo RGB o uno basado en índices; se puede también especificar si se trata de modelos de buffer sencillo o doble, y finalmente puede asociarse a otros buffers como son el stencil o la profundidad.

**glutInitWindowPosition(int x, int y)** especifica la ubicación en la pantalla de la esquina superior izquierda de nuestra ventana.

**glutInitWindowSize(int x, int y)** especifica el tamaño (en píxeles) de nuestra ventana.

**glutCreateWindow (char \*string)** se encarga de crear la ventana con un contexto OpenGL.

La pantalla se despliega cuando es llamada la función **glutMainLoop()**.



```
// =====  
// PRIMER EJEMPLO DE USO DE OPENGL  
// Objetivo: Funciones más importantes para crear una ventana en OpenGL.  
// =====  
  
#include <windows.h> // MS Windows requiere que este header se incluya antes  
// de gl.h y glu.h  
#include <GL/glut.h> // glut.h garantiza que gl.h y glu.h se incluyan  
#include <stdio.h> // Ya que muchas aplicaciones en OpenGL hacen uso de C,  
// es común incluir estos headers  
#include <stdlib.h>  
  
void init (void)  
{  
    glEnable (GL_DEPTH_TEST);  
    glClearColor (0.0, 0.0, 0.0, 0.0); // Color para borrar la pantalla  
}  
  
void display (void)  
{// función que es llamada repetida-veces para redibujar la pantalla.  
  
    glClear (GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);  
    // Limpia la pantalla con el color dado por glClearColor().  
  
    glPushMatrix ();  
    // Aquí se codificarán los dibujos...  
    glPopMatrix ();  
  
    glutSwapBuffers ();  
    // Tener doble buffer tiene la ventaja que cuando se está mostrado por  
    // pantalla, en el otro buffer se está dibujando la próxima imagen.  
    // Cada frame es mostrado únicamente cuando el dibujo ha sido terminado,  
    // si se observa la pantalla, nunca se verá un dibujo parcial.  
}  
  
void reshape(int w, int h)  
{  
    glViewport ( 0, 0, w, h ); // "VIEWING"  
    // "Viewport" hace referencia a la salida por la ventana.  
    // Que relación hay con el cero,cero de la ventana "interna" y el cero,cero  
    // de la ventana que saldrá en la pantalla.  
  
    glMatrixMode ( GL_PROJECTION );  
    // Con este comando se dice que el glLoadIdentity(), que está a  
    // continuación, van a afectar a la proyección de la matriz del  
    // modelo, en vez de la matriz modelview ("punto de vista").  
  
    glLoadIdentity ( );  
    // La matriz de transformación va a ser la matriz identidad.Esto es necesario  
    // porque las transform. se realizan multiplicando por la actual matriz de  
    // transform.. Si no se carga la matriz identidad las transformaciones  
    // se van a ir acumulando.  
  
    glOrtho (-5.0, 5.0, -5.0, 5.0, -5.0, 5.0); // "PROJECTION"  
    // Establece el sistema de coordenadas de cómo un objeto es mostrado en la  
    // pantalla. Además, con los parámetros se establece el rango de coordenada  
    // de lo que se verá. La proyección ortogonal no cambia el volumen de los  
    // objetos con la distancia de la cámara, esto no sucede con vistas en
```

```

// perspectiva.
glMatrixMode( GL_MODELVIEW ); // Vuelve al modelo de matriz "punto de vista"

glLoadIdentity ( );
}

void keyboard (unsigned char key, int x, int y)
{
    switch (key) {
        case 27:          // tecla de Escape
            exit (0);
            break;
        case 'f':
            glutFullScreen (); // De ventana a pantalla completa
            break;
        case 'w':
            glutReshapeWindow (250,250); // Salida gráfica en una ventana
            break;
        default:
            break;
    }
}

int main (int argc, char** argv)
{
    glutInit (&argc, argv); // negocia una sesión con el sistema operativo
    glutInitDisplayMode (GLUT_RGB | GLUT_DOUBLE); // Modo display. Color: RGB.
                                                    // Buffer: doble

    glutInitWindowSize (250,250);
    glutInitWindowPosition (100,100); // posición inicial de la ventana gráfica
    glutCreateWindow ("Mi primera ventana de OpenGL");

    init ();          // llamada a inicialización

    glutReshapeFunc (reshape); // Comunica con que función se
                                // va a redibujar la pantalla

    glutKeyboardFunc (keyboard); // Llamada para eventos del teclado
    glutDisplayFunc (display); // Comunica cual es la función encargada de
                                // hacer el dibujo y los repetidos refrescos
                                // de pantalla.

    // Muestra y refresca constantemente la ventana gráfica con el dibujo
    glutMainLoop (); // Crea la ventana y dibuja...

    return (0);
}

```

**glClearColor (0.0, 0.0, 0.0, 0.0)** permite definir el color con el que se limpiará la pantalla. Modo RGBA.

En el ejemplo se ha codificado para que cuando se presiona la tecla "f" o "w", GLUT detecta la acción y determina cuál tecla fue presionada. Esta información es utilizada por el programa para el cambio de pantalla completa a pantalla con tamaño determinado.

En OpenGL, las instrucciones se escriben con minúsculas (**gl**) y las constantes son con mayúsculas (**GL**).

## Puntos, líneas y polígonos (Primitivas OpenGL)

En OpenGL, las primitivas geométricas son usadas para realizar los dibujos que se desean realizar. Las primitivas son: **puntos**, **líneas** y **polígonos**. Estas primitivas se describen básicamente a partir de sus **vértices** -las coordenadas que definen al punto como tal, los extremos de los segmentos de línea y las esquinas de los polígonos-.

En un sentido matemático, la definición de las primitivas mencionadas es relativamente simple, y en OpenGL se trabaja en un contexto similar, siendo diferente en sólo lo relativo a la implementación como tal.

Las diferencias incluyen la precisión de los valores y error de redondeo, factores que influyen en las coordenadas en OpenGL, y las limitaciones del *raster graphics display*, cuya unidad mínima (píxel) es mucho mayor del concepto de matemático de infinitamente pequeño (para un punto) o infinitamente delgado (para una línea).

OpenGL se refiere a los puntos como un vector, manejado como un número de punto flotante, pero ello no implica que la precisión sea de un 100% en el momento de dibujar.

### Puntos

Se trata de números de punto flotante llamados **vertex**. Todos los cálculos que se involucran se realizan con vectores tridimensionales, y si no se define la coordenada Z (como en un dibujo de 2D), se asume un valor de cero.

### Líneas

Las líneas en OpenGL son en realidad segmentos de recta, y no el concepto de una extensión al infinito como ocurre en matemáticas. Los vértices definen los puntos extremos de cada segmento.

### Polígonos

Son las áreas encerradas por un lazo cerrado de segmentos de recta, con los vértices indicando los extremos de los segmentos que le forman. Por omisión, los polígonos se dibujan rellenos al interior.

Hay funciones avanzadas, que permiten tener polígonos "huecos", o definidos sólo como vértices en sus esquinas, colores distintos para una cara que para la otra, etc.

Es importante comprender que debido a la complejidad posible de los polígonos, es necesario definir restricciones:

- Las orillas no pueden intersectarse.
- Sólo es válido tener polígonos convexos, es decir, polígonos que cumplen la norma de regiones convexas (una región es convexa si dados cualesquiera dos puntos en sus interior, el segmento que los une está también en el interior).

El no respetar las restricciones no garantiza que el dibujo resulte como se espera. La restricción se debe a que es más sencillo tener hardware rápido para polígonos simples.

Debido a que se trata de vértices 3D, es necesario tener cuidado ya que después de algunas operaciones, si algún vértice no está en el mismo plano, puede llegarse a resultados inesperados. El usar triángulos garantiza que siempre se trabaja en un mismo plano.

### Definición de los vértices

```
glVertex{1234}{bsifd}{v} (TYPE coords)
```

Donde las coordenadas posibles son 1, 2, 3 ó 4, y los tipos de datos son sufijos de tipo:

Sufijo	Tipo de datos	Tipo de Datos en C	Definición de tipo OpenGL
b	Entero 8 bits	Signed char	GLbyte
s	Entero 16 bits	Short	GLshort
i	Entero 32 bits	Long	GLint, GLsizei
f	Real 32 bits	Float	GLfloat, GLclampf
d	Real 64 bits	Double	GLdouble, GLclampd
ub	Entero sin signo 8 bits	unsigned char	GLubyte, GLboolean
us	Entero sin signo 16 bits	unsigned short	GLushort
ui	Entero sin signo 32 bits	unsigned long	GLuint, GLenum, GLbitfield

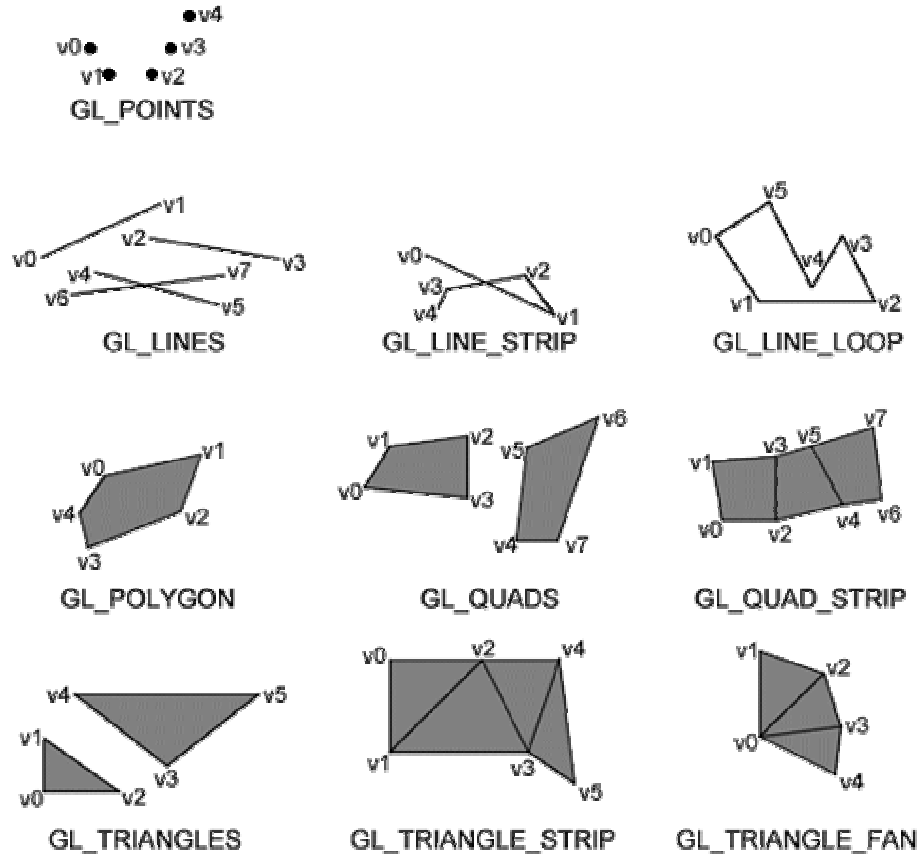
### Ejemplos:

```
glVertex2s (2, 3);
glVertex3d (0.0, 0.0, 3.1415926536898)
GLdouble dvect[3] = {5.0, 9.0, 1992.0};
glVertex3dv (dvect);
```

Las llamadas que se hacen a **glVertex** son efectivas sólo entre el par de instrucciones **glBegin()** y **glEnd()**, ya que los vértices son en realidad argumentos de estas instrucciones.

### Primitivas

Se supone un arreglo de n vértices contenidos dentro de **glBegin()** y **glEnd()**.



Además de la posición de los vértices, es posible por ejemplo definir el color, y otras funcionalidades (texturas, vectores normales, etc.) que por el momento no serán consideradas.

El código del nuevo ejemplo es el siguiente:

```
// =====
// SEGUNDO EJEMPLO DE USO DE OPENGL
// Objetivo: Ejemplo de primitivas geométricas en OpenGL.
// =====
#include <windows.h>
#include <GL/glut.h>
#include <stdio.h>
#include <stdlib.h>

void init ( void )
{
    glEnable      ( GL_DEPTH_TEST );
    glClearColor ( 0.0, 0.0, 0.0, 0.0 );
}

void display ( void )
{
    glClear ( GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT );
    glPushMatrix ( );
    glColor4f ( 1.0, 1.0, 1.0, 1.0 );
    glTranslatef(-1.5f,0.0f,0.0f);    // Las próximas figuras se
```

```

// desplazarán hacia la izquierda.

glBegin(GL_TRIANGLES); // Dibujar usando triángulos
glColor3f(1.0f,0.0f,0.0f); // Color Rojo
glVertex3f( 0.0f, 1.0f, 0.0f); // Arriba

glColor3f(0.0f,1.0f,0.0f); // Color Verde
glVertex3f(-1.0f,-1.0f, 0.0f); // Izquierda Abajo

glColor3f(0.0f,0.0f,1.0f); // Color Azul
glVertex3f( 1.0f,-1.0f, 0.0f); // Derecha Abajo
glEnd(); // Termina de dibujar triángulo

glTranslatef(3.0f,0.0f,0.0f); // Las próximas figuras se
// desplazarán hacia la derecha.

glBegin(GL_QUADS); // Dibuja cuadrado
glVertex3f(-1.0f, 1.0f, 0.0f); // Izquierda Arriba
glVertex3f( 1.0f, 1.0f, 0.0f); // Derecha Arriba
glVertex3f( 1.0f,-1.0f, 0.0f); // Derecha Abajo
glVertex3f(-1.0f,-1.0f, 0.0f); // Izquierda Abajo
glEnd();

glPopMatrix ( );
glutSwapBuffers ( );
}

void reshape(int w, int h)
{
glViewport ( 0, 0, w, h );
glMatrixMode ( GL_PROJECTION );
glLoadIdentity ( );
glOrtho (-5.0, 5.0, -5.0, 5.0, -5.0, 5.0);
glMatrixMode ( GL_MODELVIEW );
glLoadIdentity ( );
}

void keyboard ( unsigned char key, int x, int y )
{
switch ( key ) {
case 27: // tecla de Escape
exit ( 0 );
break;
case 'f':
glutFullScreen ( );
break;
case 'w':
glutReshapeWindow ( 250,250 );
break;
default:
break;
}
}

int main ( int argc, char** argv )
{
glutInit (&argc, argv);
glutInitDisplayMode (GLUT_RGB | GLUT_DOUBLE);
glutInitWindowSize (250,250);
glutInitWindowPosition (100,100);

```

```
glutCreateWindow ("Primitivas en OpenGL");
init ();

glutReshapeFunc ( reshape );
glutKeyboardFunc ( keyboard );
glutDisplayFunc ( display );

glutMainLoop ( );

return (0);
}
```

En este código se realizan dibujos a color de dos primitivas.

Podemos ver que las modificaciones son en el "área de trabajo" del programa, entre las "matrices de trabajo":

```
void display (void)
{
    glClear (GL_COLOR_BUFFER_BIT |
            GL_DEPTH_BUFFER_BIT);
    glPushMatrix ();
    [nuevos contenidos]
    glPopMatrix ();
    glutSwapBuffers ();
}
```

Esto es lo que se debe obtener:



## Rotación/Introducción a las transformaciones

Partiendo del código anterior, se agregaron algunas ligeras modificaciones significativas para dar interactividad a los dibujos.

```

// =====
// TERCER EJEMPLO DE USO DE OPENGL
// Objetivo: Ejemplo de animación en OpenGL.
// Rotación pulsando la tecla "t" o "c"
// =====

#include <windows.h>
#include <GL/glut.h>
#include <stdio.h>
#include <stdlib.h>

GLfloat rtri; // Angulo de rotación del triángulo...
GLfloat rquad; // Angulo de rotación del cuadrado...

void init (void)
{
    glEnable (GL_DEPTH_TEST);
    glClearColor (0.0, 0.0, 0.0, 0.0);
}

void display (void)
{
    glClear (GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glPushMatrix ();

    glTranslatef(-1.5f,0.0f,0.0f); // Desplaza próximos dibujos a izquierda
    glRotatef(rtri,0.0f,1.0f,0.0f); // Rotación en eje Y, ángulo variable.
    // Próximos dibujos aparecerán rotados.

    // Verificar que no es lo mismo rotar y trasladar que al revés.

    glBegin(GL_TRIANGLES); // Dibuja un Triángulo
    glColor3f(1.0f,0.0f,0.0f); glVertex3f( 0.0f, 1.0f, 0.0f);
    glColor3f(0.0f,1.0f,0.0f); glVertex3f(-1.0f,-1.0f, 0.0f);
    glColor3f(0.0f,0.0f,1.0f); glVertex3f( 1.0f,-1.0f, 0.0f);
    glEnd();

    glLoadIdentity(); // "Reset" de la matriz actual de visualización
    // modelview porque va acumulando las matrices
    // de transformación. Observar que sucede al no
    // ejecutarse esta línea (se acumula las rotación
    // al rededor del eje Y).

    glTranslatef(1.5f,0.0f,0.0f); // Desplazar próximos dibujos a derecha
    glRotatef(rquad,0.0f,1.0f,0.0f); // Rotación en eje X, ángulo variable.
    // Los próximos dibujos aparecerán rotados

    glColor3f(0.5f,0.5f,1.0f); // Próximos dibujos van a ser en Azul

    glBegin(GL_QUADS); // Cuadrilátero
    glVertex3f(-1.0f, 1.0f, 0.0f);
    glVertex3f( 1.0f, 1.0f, 0.0f);
    glVertex3f( 1.0f,-1.0f, 0.0f);
    glVertex3f(-1.0f,-1.0f, 0.0f);
    glEnd();

    glPopMatrix ();
    glutSwapBuffers ();
}

```

```

void reshape(int w, int h)
{
    glViewport ( 0, 0, w, h );
    glMatrixMode ( GL_PROJECTION );
    glLoadIdentity ( );
    glOrtho (-5.0, 5.0, -5.0, 5.0, -5.0, 5.0);
    glMatrixMode ( GL_MODELVIEW );
    glLoadIdentity ( );
}

void keyboard (unsigned char key, int x, int y)
{
    switch (key) {
        case 27: // tecla de Escape
            exit (0);
            break;
        case 'f':
            glutFullScreen ();
            break;
        case 'w':
            glutReshapeWindow (250,250);
            break;
        case 't': // tecla para rotar "t"riángulo
            rtri+=1.2f;
            glutPostRedisplay(); // llama a la función display.
            break;
        case 'c': // tecla para rotar "c"uadrado
            rquad-=1.2f;
            glutPostRedisplay();
            break;
        default:
            break;
    }
}

int main (int argc, char** argv)
{
    glutInit (&argc, argv);
    glutInitDisplayMode (GLUT_RGB | GLUT_DOUBLE);
    glutInitWindowSize (250,250);
    glutInitWindowPosition (100,100);
    glutCreateWindow ("Rotaciones - Pulsar [t] o [c]");
    init ();

    glutReshapeFunc (reshape); // llamada p/eventos reshape de ventana
    glutKeyboardFunc(keyboard); // llamada para eventos del teclado
    glutDisplayFunc (display); // llamada para dibujar...
    glutMainLoop (); // Crea la ventana y dibuja...

    return (0);
}

```

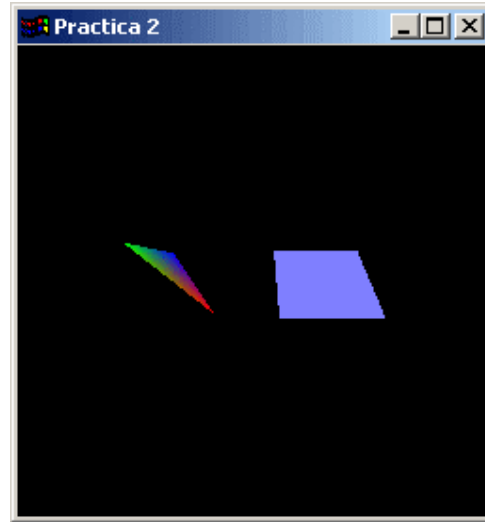
Las instrucciones que aparecen por primera vez en el programa son de este tipo: (en la sección de **display**)

```
glRotatef(rtri,0.0f,1.0f,0.0f);
```

Donde se agregó una instrucción nueva que permite la rotación alrededor de un vector, y (en la sección de **keyboard**) del tipo:

```
case 't':  
    rtri+=1.2f;  
    glutPostRedisplay();  
    break;
```

donde se definieron nuevas variables para determinar la rotación de las primitivas. El efecto permite la rotación con las teclas "t" o "c":



Los parámetros de **glRotatef** son los que siguen:

```
void glRotatef(float angle, TYPE x, TYPE y, TYPE z)
```

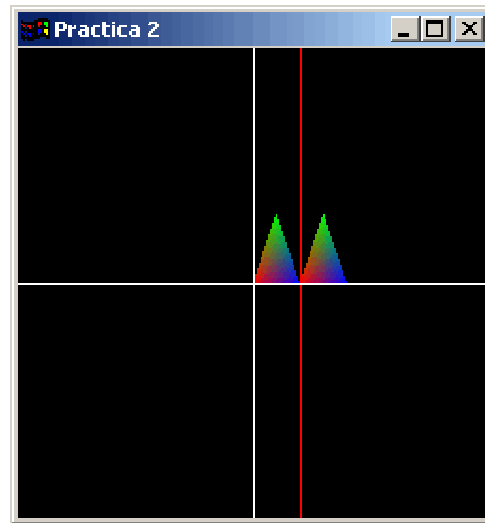
Donde el ángulo de rotación está expresado en grados.

```
glLoadIdentity();
```

se encarga de limpiar las transformaciones sobre el "modelo" o dibujo.

Es interesante observar lo que se logra cuando es comentada la línea que posee a **glLoadIdentity();** dentro de la función **void display (void)**, para que no sea compilada. Cuando es pulsada la tecla "t" la rotación es de ambas primitivas respecto a un mismo eje.

Para observar la posición de una figura y la rotación alrededor de los ejes (puede verse también como una rotación de un sistema de coordenadas), se presenta el siguiente ejemplo:



```
// =====
// CUARTO EJEMPLO DE USO DE OPENGL
// Objetivo:
// Ejemplo para observar la rotación respecto a los ejes. Teclas:
// <ESC>: salir
// f: pantalla completa
// w:en ventana
// y:rotación en eje "y"
// x:rotación en eje "x"
// a: rotación en "a"mbos ejes.
// =====

#include <windows.h>
#include <GL/glut.h>
#include <stdio.h>
#include <stdlib.h>

GLfloat Rot_Y; // Angulo de rotación respecto a "Y"
GLfloat Rot_X; // Angulo de rotación respecto a "X"

void init (void)
{
    glEnable      (GL_DEPTH_TEST);
    glClearColor (0.0, 0.0, 0.0, 0.0);
}

void display (void)
{
    glClear (GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glPushMatrix ();
    glTranslatef(0,0.0f,-6.0f);

    glBegin(GL_LINES); // dibujo de los "ejes"
        glColor3f(1.0f,1.0f,1.0f); // Blanco
        glVertex3f(-10.0f,0.0f,0.0f);
        glVertex3f(10.0f,0.0f,0.0f);
        glVertex3f(0.0f,-10.0f,0.0f);
        glVertex3f(0.0f,10.0f,0.0f);
        glColor3f(1.0f,0.0f,0.0f); // Rojo

```

```

    glVertex3f(2.0f,10.0f,0.0f);
    glVertex3f(2.0f,-10.0f,0.0f);
glEnd();

glRotatef(Rot_Y,0.0f,1.0f,0.0f);    // Rotación con eje "Y"
glRotatef(Rot_X,1.0f,0.0f,0.0f);    // Rotación con eje "X"

glBegin(GL_TRIANGLES);    // Primer triángulo
    glColor3f(1.0f,0.0f,0.0f);    glVertex3f( 0.0f, 0.0f, 0.0f);
    glColor3f(0.0f,1.0f,0.0f);    glVertex3f(1.0f,3.0f, 0.0f);
    glColor3f(0.0f,0.0f,1.0f);    glVertex3f( 2.0f,0.0f, 0.0f);
glEnd();

glLoadIdentity();
glTranslatef(2.0f,0.0f,-6.0f);
glRotatef(Rot_Y,0.0f,1.0f,0.0f);    // Rotación con eje "Y"
glRotatef(Rot_X,1.0f,0.0f,0.0f);    // Rotación con eje "X"

glBegin(GL_TRIANGLES);    // Segundo triángulo
    glColor3f(1.0f,0.0f,0.0f);    glVertex3f( 0.0f, 0.0f, 0.0f);
    glColor3f(0.0f,1.0f,0.0f);    glVertex3f(1.0f,3.0f, 0.0f);
    glColor3f(0.0f,0.0f,1.0f);    glVertex3f( 2.0f,0.0f, 0.0f);
glEnd();

glPopMatrix ();
glutSwapBuffers ();
}

void reshape(int w, int h)
{
    glViewport ( 0, 0, w, h );
    glMatrixMode ( GL_PROJECTION );
    glLoadIdentity ( );
    glOrtho (-10.0, 10.0, -10.0, 10.0, -0.0, 10.0);
    glMatrixMode ( GL_MODELVIEW );
    glLoadIdentity ( );
}

void keyboard (unsigned char key, int x, int y)
{
    switch (key) {
        case 27: // tecla de Escape
            exit (0); break;
        case 'f':
            glutFullScreen (); break;
        case 'w':
            glutReshapeWindow (250,250); break;
        case 'y':
            Rot_Y+=5.0f;
            glutPostRedisplay(); break;
        case 'x':
            Rot_X-=5.0f;
            glutPostRedisplay(); break;
        case 'a':
            Rot_Y+=5.0f; Rot_X-=5.0f;
            glutPostRedisplay(); break;
        default:
            break;
    }
}

```

```

int main (int argc, char** argv)
{
    glutInit (&argc, argv);
    glutInitDisplayMode (GLUT_RGB | GLUT_DOUBLE);
    glutInitWindowSize (250,250);
    glutInitWindowPosition (100,100);
    glutCreateWindow ("Rotaciones - Pulsar [y]-[x]-[a]");
    init (); // llamada a inicialización

    glutReshapeFunc (reshape); // llamada p/eventos de reshape de ventana
    glutKeyboardFunc (keyboard); // llamada para eventos de kbd
    glutDisplayFunc (display); // llamada para dibujar...

    glutMainLoop (); // Crea la ventana y dibuja...

    return (0);
}

```

Las teclas **y**, **x** y **a** permiten apreciar 3 rotaciones distintas (respecto al eje y, al eje x, y respecto a ambos ejes). Analizando el código puede interpretarse cada eje y la diferenciación de los sistemas de coordenadas.

Para poder compilar y ejecutar los archivos de ejemplo, tener en cuenta usar el makefile adecuado:

```

=====
# Prototipo Básico de makefile para compilación con GNU C/C++
=====
# Uso en Windows
=====

CXX=g++
OPCIONES=-g -Wall
DIRECTORIOS=-I../util -I/usr/include/GL -I/bin -I/usr/include/w32api -
I/usr/include
BIBLIOTECAS=-L/lib -lglui -lglut32 -lglu32 -lopengl32 -lm

all:
${CXX} ${OPCIONES} ${DIRECTORIOS} main.cpp -o main ${BIBLIOTECAS}

# Para usar OpenGL, incluir en el programa las siguientes líneas
# include <GL/glut.h>
# include <glui.h>

=====
# Uso en Linux
=====

CXX=g++
OPCIONES=-g -Wall
DIRECTORIOS=-I../util -I/usr/include/GL -I/bin
BIBLIOTECAS=-L/lib -lm -L/usr/lib/ -lGL -lglut -lGLU

all:
${CXX} ${OPCIONES} ${DIRECTORIOS} main.cpp -o main ${BIBLIOTECAS}

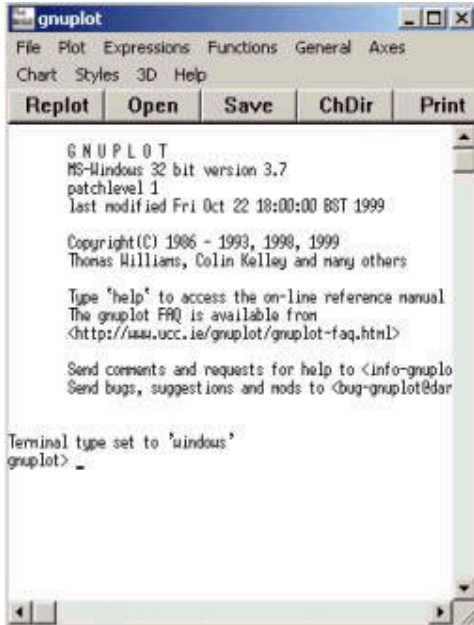
```

```
# Para usar OpenGL, incluir en el programa la siguiente línea  
# include <GL/glut.h>
```

# Graficación con GNUPlot

## Introducción

Gnuplot es un programa que produce gráficos en 2D y 3D a partir de ecuaciones o valores. Puede ser ejecutado desde la línea de comando y trabajar en forma interactiva dentro de una ventana en un entorno como el mostrado en la figura siguiente:



Pero también puede ser llamado desde la línea de comando, pasándole un archivo que posea la lista de instrucciones indicando lo que gnuplot deberá graficar. Teniendo en cuenta esta última modalidad, puede llamarse desde un programa en C y graficar en base a un archivo construido también desde C mediante:

```
system("gnuplot graf.plt");
```

se está invocando desde un programa en C al sistema operativo para que ejecute *gnuplot*, donde *graf.plt* tiene las instrucciones de lo que se quiere graficar. Si se trabaja en Windows, el comando es *wgnuplot*.

[Gnuplot \(www.gnuplot.info\)](http://www.gnuplot.info) es de libre distribución y existe en varias versiones de sistema operativo.

Se mostrará las funcionalidades más usadas en la graficación de funciones a partir de programas en C. Mayor información sobre el uso del programa puede obtenerse en: [manual de Gnuplot \(http://www.ucc.ie/gnuplot/gnuplot.html\)](http://www.ucc.ie/gnuplot/gnuplot.html), [página de demos de Gnuplot \(http://www.gnuplot.vt.edu/gnuplot/gpdocs/all2.html\)](http://www.gnuplot.vt.edu/gnuplot/gpdocs/all2.html) y [el tutorial de la Universidad de Iowa \(http://www.cs.uni.edu/Help/gnuplot/\)](http://www.cs.uni.edu/Help/gnuplot/). También se puede consultar en el newsgroup *comp.graphics.apps.gnuplot*.

## Primer ejemplo

Los ejemplos que se verán a continuación contemplan el llamado a GnuPlot desde un programa escrito en C. En el siguiente ejemplo<sup>1</sup> `"/"` está puesto porque el ejecutable `gnuplot`, en este ejemplo, está instalado en el directorio superior al que se está corriendo en programa. Si estuviese en el mismo directorio que el ejecutable de este ejemplo entonces colocar `"/"`. Si está en cualquier otro directorio, colocar el camino completo.

<sup>1</sup> Se debe verificar que en el directorio `"c:\cygwin\bin"` exista el archivo `"sh.exe"`. Si no existe, hacer una copia de `"bash.exe"` y renombrarlo `"sh.exe"`. No se debe borrar el archivo `"bash.exe"`.

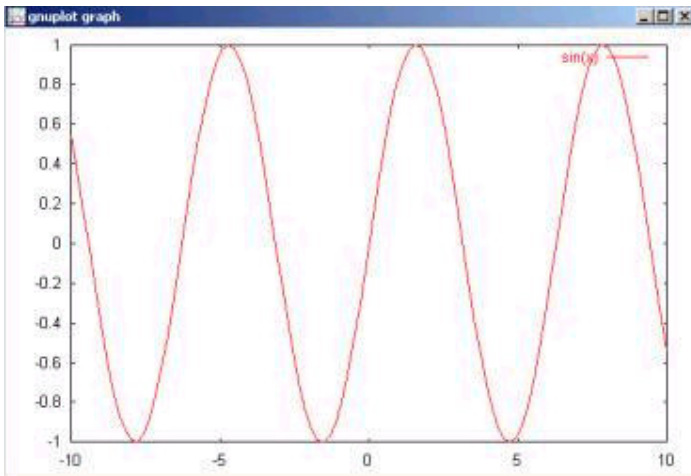
Si el directorio donde está instalado wgnuplot se encuentre en el path del "autoexec.bat", no es necesario la colocación de "./".

```
#include <stdlib.h>
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
// generación del archivo script
ofstream graf("graf.plt");

graf << "plot sin(x)" << endl;
graf << "pause -1 'Pulsar para finalizar' " << endl;
graf.close();

// llamado a gnuplot con el archivo generado
system("../wgnuplot graf.plt"); // en Linux es "gnuplot" y no se coloca "../"
}
```



La primera parte del programa genera el archivo que va a procesar gnuplot. Luego, `plot sin(x)` indica que se grafique la función seno(x) y `pause -1 'Pulsar para finalizar'` indican que realice una pausa mostrando un botón a pulsar con el mensaje indicado. En este caso, Gnuplot eligió automáticamente el rango de las escalas.

En general, son válidas las expresiones matemáticas aceptadas por C, por ejemplo: `abs(x)`, `acos(x)`, `asin(x)`, `atan(x)`, `cos(x)`, `cosh(x)`,

`erf(x)`, `exp(x)`, `inverf(x)`, `invnorm(x)`, `log(x)`, `log10(x)`, `norm(x)`, `rand(x)`, `sign(x)`, `sin(x)`, `sinh(x)`, `sqrt(x)`, `tan(x)` y `tanh(x)`.

Otras funciones como Bessel, gamma, ibeta, igamma, lgamma, funciones con argumentos complejos. Además operadores binarios y unarios. El operador de exponente es `**`.

El archivo de texto con las instrucciones que debe procesar GnuPlot puede ser generado con un editor, pero en estos ejemplos se supone que se está integrando la graficación en un programa más completo.

## Los comandos *plot* y *splot*

**plot** y **splot** son los comandos de graficación. `plot` es usado para graficar funciones y tabla de datos en 2D, mientras que `splot` es para superficies y datos en 3D.

**Sintaxis:**

```

plot {[rangos]}
  {[función] | {"[archivodedatos]" { archivodedatos -modificadores}}}
  {ejes [ejes] } { [titulo-spec] } {with [estilo] }
  {, {definiciones,} [función] ...}

```

donde [función] o el nombre del archivo de datos deberá estar entre comillas. En los siguientes ejemplos se verá con mayor claridad.

**Segundo ejemplo**

En este ejemplo se agregan otras características que permite gnuplot.

```

ofstream graf("graf.plt");
graf << "set xrange [-2:2]" << endl
  << "set yrange [-1:4]" << endl
  << "set xlabel 'X-Values' " << endl
  << "set ylabel 'Y-Values' 0, -10 " << endl
  << "set zeroaxis" << endl
  << "plot sin(x**4) title 'El Seno', x**2 title 'Cuadratica' " << endl
  << "pause -1 'Pulsar para terminar' " << endl;

```

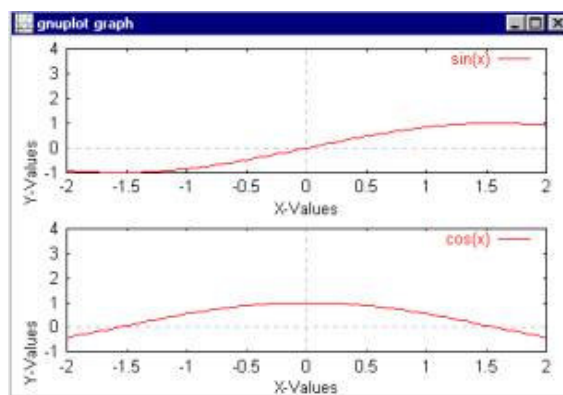
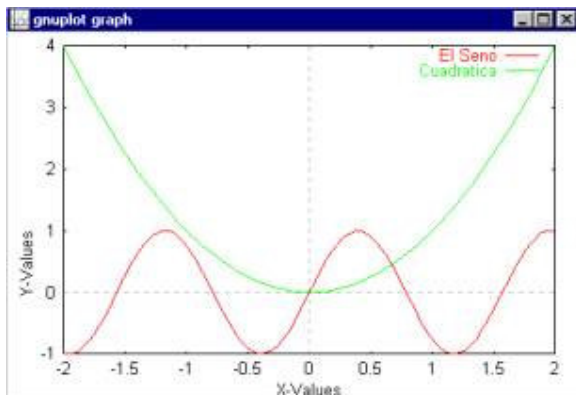
Con **xrange** e **yrange** se establece el rango de los ejes. Mediante **xlabel** e **ylabel** se muestran los carteles de los ejes. Con **zeroaxis** se indica que se debe dibujar los ejes para  $x=0$  y  $y=0$ . Por último, se muestra la graficación de dos funciones en un mismo sistema de ejes, cada una con un título personalizado.

También puede realizarse gráficos con distintos ejes en una misma ventana. Utilizando la instrucción **multiplot**. Con **nomultiplot** se vuelve a establecer la condición de que todos los gráficos van en los mismos ejes de coordenadas.

```

graf << "set multiplot" << endl
  << "set size 1,0.5" << endl
  << "set origin 0.0,0.5; plot sin(x)" << endl
  << "set origin 0.0,0.0; plot cos(x)" << endl
  << "set nomultiplot" << endl
  << "pause -1 'Salir' " << endl;

```



## Graficación de puntos

Un conjunto de datos contenidos en un archivo también pueden ser graficados mediante **plot** o **splot**. Los datos deben estar en columnas separados por espacios en blanco o tabulaciones únicamente, no está permitido separar por comas. Las líneas que comienzan con el caracter # son tratadas como un comentario e ignoradas por gnuplot.

Por ejemplo se tiene los siguientes tres archivos de texto:

# Este archivo es "datos1.dat"	# Este archivo es "datos2.dat"	# Este archivo es "datos3.dat"
# Datos de Flexión de una viga	# Datos de Flexión y Fuerza	# Datos de Flexión y Fuerza
# Flexión	# aplicada a una viga y una barra.	# aplicada a una viga y una barra.
0.000	# Flexión      Fuerza-barra	# Flexión      Fuerza-barra      Fuerza-Viga
0.001	0.000            0	0.000            0            0
0.002	0.001            104	0.001            104            51
0.003	0.002            202	0.002            202            101
0.0031	0.003            298	0.003            298            148
0.004	0.0031           290	0.0031           290            149
0.0041	0.004            289	0.004            289            201
0.005	0.0041           291	0.0041           291            209
0.010	0.005            310	0.005            310            250
0.020	0.010            311	0.010            311            260
	0.020            280	0.020            280            240

El primer archivo corresponde a valores de flexión de una barra o viga de acero. El segundo archivo posee los valores correspondientes a la fuerza aplicada a la barra para obtener la flexión consignada. En el tercer archivo se agregó la fuerza que se necesitaría en una viga para lograr las mismas flexiones.

Para mostrar el primer archivo sólo se agrega el nombre del archivo y el título. El eje x muestra el número de orden del dato. Para el segundo archivo, como se tiene dos columnas, se interpreta la primera como eje x y la segunda como eje y. Para el tercer archivo se debe especificar qué columnas se graficarán siendo la primer columna indicada lo correspondiente al eje x y la otra al eje y.

Con en el siguiente código se muestran los gráficos:

```
ofstream graf("graf.plt");
graf << "plot 'datos1.dat' title 'Flexión' << endl
      << "pause -1 'Próximo gráfico' " << endl;

// segundo gráfico
graf << "set grid" << endl
      << "plot 'datos2.dat' title 'Flexión y Fuerza' with lines" << endl
      << "pause -1 'Próximo gráfico' " << endl;

// tercer gráfico
graf << "set ngrid" << endl
      << "plot 'datos3.dat' using 1:2 title 'Flexión y Fuerza-Barra',\
      'datos3.dat' using 1:3 smooth csplines title 'Flexión y Fuerza-Viga' "
      << endl
      << "pause -1 'Próximo gráfico' " << endl;

// gráfico con más detalles
graf << "set title 'Fuerza de Flexión en una Columna y Viga'" << endl
      << "set xlabel 'Flexión (metros)'" << endl
      << "set ylabel 'Fuerza (kN)'" << endl
```

```

<< "set key 0.01,100" << endl // coordenadas para los 'titles'
<< "set label 'Punto de Inflexión' at 0.003,260" << endl
<< "set arrow from 0.0028,250 to 0.003,280" << endl // dibujo de una
flecha
<< "set xr [0.0:0.022]" << endl
<< "set yr [0:325]" << endl
<< "plot 'datos3.dat' using 1:2 title 'Flexión y Fuerza-Barra' with
linespoints,\
'datos3.dat' using 1:3 title 'Flexión y Fuerza-Viga' with points" << endl
<< "pause -1 'Salir' " << endl;

```

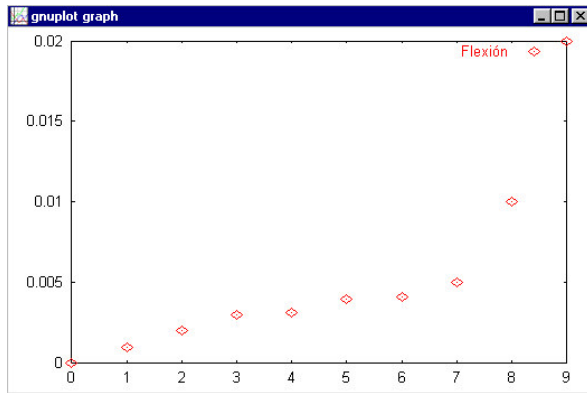


Gráfico 1

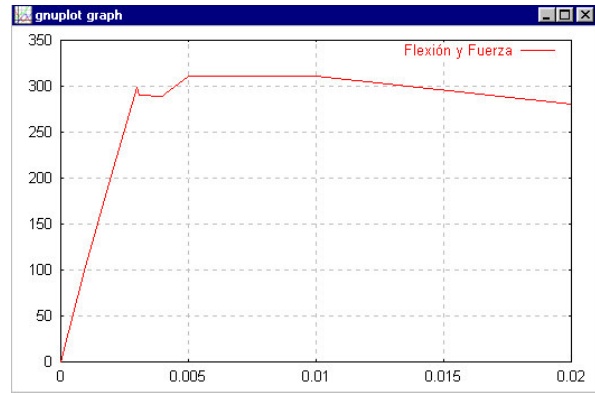


Gráfico 2

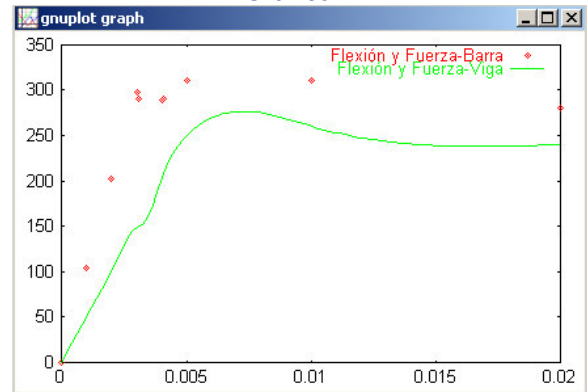


Gráfico 3

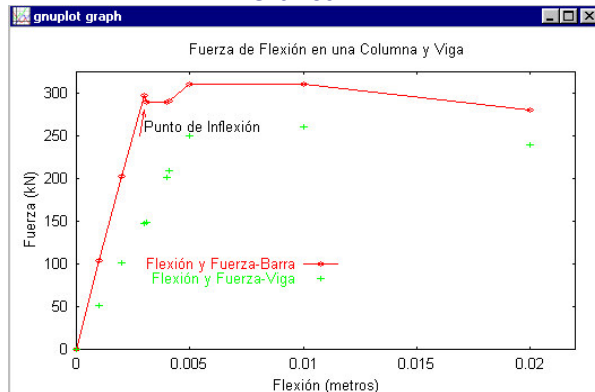


Gráfico 4

En el segundo gráfico se indicó que los puntos sean unidos con líneas mediante **with lines**. Una línea en blanco en el archivo de los datos implica una interrupción en las líneas que los unen. Otras alternativas a líneas son:

points, linespoints, impulses, dots, steps, fsteps, histeps, errorbars, xerrorbars, yerrorbars, xyerrorbars, boxes, boxerrorbars, boxxyerrorbars, financebars, candlesticks o vector.

Además, los nombres **using**, **title** y **with** pueden ser abreviados mediante **u**, **t** y **w**. También puede graficarse en una escala logarítmica con **set logscale**.

No se debe tipear ningún espacio en blanco a continuación del carácter "\", utilizado para cortar una línea de código en C.

## Resumen de algunas personalizaciones

Las personalizaciones realizadas con el comando **set**, vistas en los ejemplos anteriores, están resumidas a continuación (los valores numéricos son arbitrarios). Cuando se establece una personalización, se afectarán todos los gráficos que se realicen a continuación. Si se quiere aplicar sobre un gráfico ya dibujado, se deberá hacer **replot**. También puede utilizarse lo detallado anteriormente para quitar personalizaciones realizadas para un gráfico previo.

Crear el título	Set title "Fuerza de Flexión "
Poner una etiqueta en el eje x	Set xlabel "Flexión (metros)"
Poner una etiqueta en el eje y	Set ylabel "Fuerza (kN)"
Cambiar el rango del eje x	Set xrange [0.001:0.005]
Cambiar el rango del eje y	Set yrange [20:500]
Rangos automáticos	Set autoscale
Mover la referencia	Set key 0.01,100
Eliminar la referencia	Set nokey
Poner una etiqueta	Set label "Punto inflexión" at 0.003, 260
Eliminar todas las etiquetas	Set nolabel
Utilizar escala logarítmica en x	Set logscale
Utilizar escala logarítmica en y	Set nologscale; set logscale y
Posición de las divisiones en x	Set xtics (0.002,0.004,0.006,0.008)
Divisiones por omisión	Set noxtics; set xtics

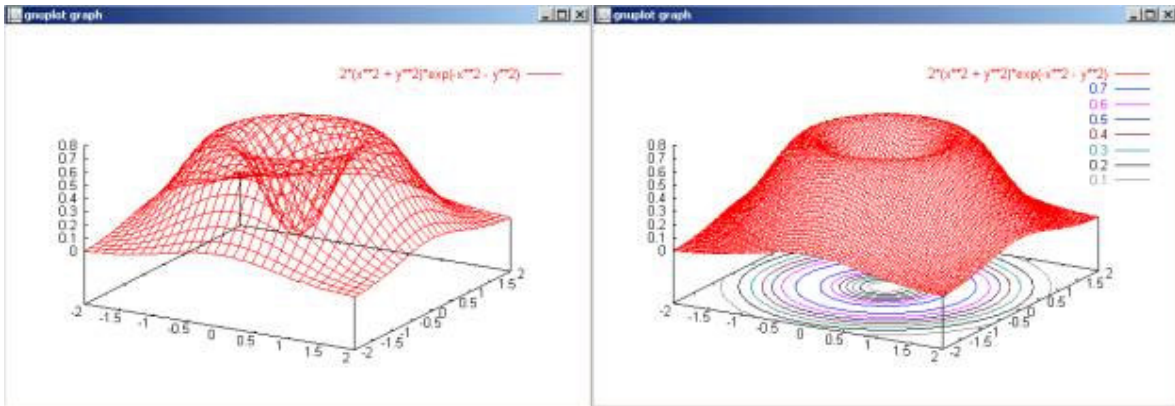
Otras características para usar con el comando **set** son: arrow, border, clip, contour, grid, mapping, polar, surface, time, view, y muchas otras más.

## Gráficos de superficie

El siguiente ejemplo muestra el gráfico de una superficie utilizando **splot**.

```
ofstream graf("graf.plt");
graf << "set isosamples 30, 30" << endl
  << "splot [-2:2] [-2:2] 2*(x**2 + y**2)*exp(-x**2 - y**2)" << endl

  << "pause -1 'Continuar' " << endl
  << "set isosamples 100, 100" << endl
  << "set hidden3d" << endl
  << "set contour base" << endl
  << "replot" << endl
  << "pause -1 'Salir' " << endl;
```



Las personalizaciones realizadas son:

- **Isosamples:** especifica el número de muestras a graficar. Observar que a mayor argumento, se obtiene una superficie más definida.
- **Hidden3d:** indica que la superficie no será transparente.
- **Contour base:** muestra curvas de nivel en la base del gráfico.

Muchas otras personalizaciones están disponibles para superficies, como distintas vistas y zoom (set view horizontal\_angle, vertical\_angle, zoom).

## Salida de gráfico a archivo

De una manera muy simple se puede crear archivos gráficos, por ejemplo, imágenes en formato gif. El siguiente código muestra la creación de una imagen de la función seno grabada en el archivo "seno.gif".

```
graf << "set out 'seno.gif' " << endl
  << "set size 1.0, 0.5" << endl
  << "set terminal gif size 640,480" << endl
  << "plot sin(x)" << endl;
```

# Cálculo numérico

## Introducción

Los métodos numéricos son herramientas muy poderosas para el análisis y la búsqueda de soluciones en los campos de la ingeniería, física, química, etc.

Ante la necesidad de utilización de un método numérico, es recomendable explorar primero la extensa biblioteca existente a disposición en internet. En este capítulo se mostrará algunos usos de la biblioteca "Numerical Recipes" desarrollada por la Universidad de Harvard y que puede ser libremente descargada desde la página <http://cfatab.harvard.edu/nr/bookcpdf.html> donde también se encuentra una extensa bibliografía sobre la biblioteca.

Esta biblioteca tiene métodos y funciones muy optimizados y adaptados a la generalidad de las aplicaciones. Sin embargo, hay que tener en cuenta que para algunas aplicaciones específicas es posible que se requiera de una implementación particular y adecuada al caso.

Para ilustrar como se utilizan algunos métodos de solución incluidos en esta biblioteca se presenta a continuación su aplicación<sup>2</sup> a algunos ejemplos.

## Ejemplo 1: interpolación

Un programa de investigación cuyo objetivo es el estudio de propiedades de una determinada sustancia mediante el bombardeo con electrones, incluye en su desarrollo una etapa experimental que permite obtener una curva del comportamiento aproximado de la energía cinética de las partículas utilizadas para el bombardeo, en función de la velocidad que les confiere un acelerador de partículas (ciclotrón).

La experiencia consiste básicamente en determinar la energía cinética de las partículas (a partir de la energía de impacto en una placa especial) para diferentes velocidades dadas por variación del campo eléctrico del acelerador de partículas.

Lectura	1	2	3	4	5	6	7	8	9	10
$v [m/s]. 10^8$	0.5	0.75	1	1.25	1.5	1.75	2	2.25	2.35	2.5
$E [J] . 10^{-12}$	1.14	2.56	4.55	7.12	10.2	14	18.2	23	25.1	28.5

Se busca realizar un programa que permita calcular los valores de energía cinética para velocidad distintas a las tabuladas.

<sup>2</sup> Para estudiar la teoría de estos métodos se recomienda "Métodos Numéricos Aplicados con Software" – Shoichiro Nakamura – Ed. Prentice Hall.

## Implementación del programa

Cuando un vector tiene N elementos, la biblioteca indexa de 1 hasta N y no de 0 a N-1 como en C/C++. Por este motivo, cuando se pasan vectores a funciones se lo hace de la siguiente forma: **&vector[-1]**.

```
#include "nr.h"
#include <vector>
#include <fstream>
using namespace std;

int main()
{ vector< float > vector_x, vector_y;
  float tmp_x, tmp_y;

  // Se lee el archivo con los valores (por fila: x y)
  ifstream archivo("valores.txt");
  while (archivo >> tmp_x >> tmp_y)
  { vector_x.push_back(tmp_x);
    vector_y.push_back(tmp_y);
  }
  archivo.close();

  // Se muestran los valores leidos
  for (unsigned k=0; k<vector_x.size(); k++)
    cout << vector_x[k] << " " << vector_y[k] << endl;

  // Cálculo de la media y varianza con funciones de la biblioteca
  float valor_medio, varianza;
  avevar(&vector_y[-1], vector_y.size(), &valor_medio, &varianza);
  cout << "Valor Medio Y: " << valor_medio << " - Varianza Y: "
        << varianza << endl;

  // Interpolación con la biblioteca
  vector< float > vector_interp;
  float val_int_x, val_int_y, err;

  cout << "Ingrese valor a interpolar: ";
  cin >> val_int_x;
  polint(&vector_x[-1], &vector_y[-1], vector_x.size(), val_int_x, &val_int_y, &err);
  cout << endl << " Valor interpolado: " << val_int_y << endl;

  cin.get();
  return (0);
}

/* MAKEFILE */
all:
  g++ -Wall main.cpp -o main nrc.a
```

La biblioteca debe ser incluida a través de la línea `#include "nr.h"` y además se debe poseer el archivo **nrc.a** (que es la versión compilada de la librería para el sistema operativo que se esté usando) que es linkeado a través del makefile.

Un primer uso de la biblioteca es "avevar" que calcula el valor medio y la varianza de un vector de valores. Como parámetros, se le debe pasar el vector con los valores, la cantidad de valores, devolviendo por referencia el valor medio y la varianza.

```
avevar(&vector_y[-1],vector_y.size(),&valor_medio,&varianza);
```

Otra función es la que permite obtener un valor interpolado, a partir de una tabla de datos conocidos.

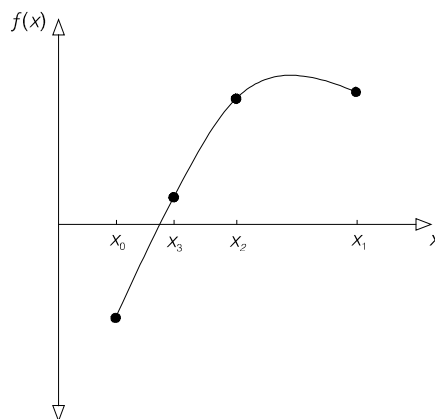
```
polint(&vector_x[-1],&vector_y[-1],vector_x.size(), val_int_x,&val_int_y,&err);
```

En donde:

- **&vector\_x[-1]**: vector con los datos de la variable independientes (x).
- **&vector\_y[-1]**: vector con los datos de la variable dependiente (y).
- **vector\_x.size()**: cantidad de valores de la tabla.
- **val\_int\_x**: valor de x a interpolar.
- **&val\_int\_y**: valor interpolado calculado por la función polint.
- **&err**: error obtenido en el cálculo.

## Ejemplo 2: solución de ecuaciones no lineales

Las soluciones de una ecuación no lineal se llaman raíces o ceros y corresponde al valor de la variable independiente para cuando la función vale cero.



La razón para resolver ecuaciones no lineales por medio de métodos computacionales es que esas ecuaciones carecen de solución exacta, excepto para muy pocos problemas. La solución analítica de las ecuaciones polinomiales existe sólo hasta el orden cuatro, pero no existen soluciones en forma exacta para órdenes superiores. Por lo tanto, las raíces de esas ecuaciones no lineales se obtienen mediante métodos computacionales basados en procedimientos iterativos.

### Método de bisección

Este método es el más simple, aunque también el más seguro y sólido para encontrar una raíz en un intervalo dado donde se sabe que existe dicha raíz. Su única ventaja es que funciona aun para funciones no analíticas.

Ejemplo de simple de uso de la biblioteca

```

#include <iostream>
#include "nr.h"
using namespace std;

// -- función
static float fx(float x)
{ return(x*x-4); }

int main(void)
{ float x0,x1,tolerancia,raiz;

  x0=1.0;
  x1=3.0;
  tolerancia=1.0e-3;

  raiz=rtbis(fx,x0,x1,tolerancia);

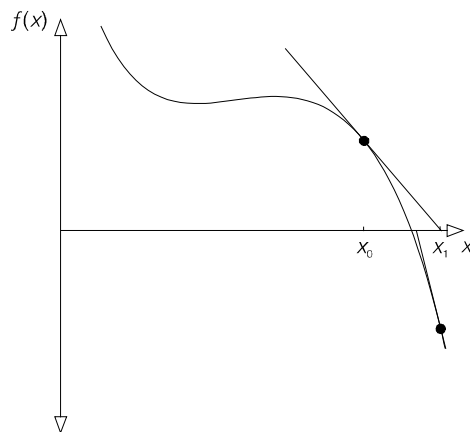
  cout << "La raiz es: " << raiz << endl;
  cout << "En valor de la función en la raiz encontrada: "
    << fx(raiz) << endl;

  return(0);
}

```

## Método de Newton

Este método (también llamado método de Newton-Raphson) encuentra una raíz, siempre y cuando se conozca una estimación inicial para la raíz deseada. Utiliza las rectas tangentes que se evalúan analíticamente.



## Ejemplo de simple de uso de la biblioteca

```

#include <iostream>
#include "nr.h"
using namespace std;

// -- función
static float fx(float x)
{ return(x*x-4); }

```

```
// -- derivada
static float dfx(float x)
{ return(2*x); }

// -- formato para NRC
static void funcd(float x, float *fn, float *df)
{ *fn=fx(x); *df=dfx(x); }

// -----
int main(void)
{ float x0,x1,tolerancia,raiz;

  x0=1.0;
  x1=3.0;
  tolerancia=1.0e-3;

  raiz=rtnewt(funcd,x0,x1,tolerancia);

  cout << "La raiz es: " << raiz << endl;
  cout << "En valor de la función en la raiz encontrada: "
        << fx(raiz) << endl;

  return(0);
}
```

## PARTE III ANEXOS

# Introducción a HTML

Esta guía ha sido creada para dar una visión inicial en la creación de documentos en HTML (HyperText Markup Language). Es una forma fácil de publicar resultados de la ejecución de programas en la World Wide Web<sup>3</sup>.

## Directivas

Las *directiva* (en inglés *tags*) en HTML proporcionan información adicional al cliente que visualiza el documento. Un documento sin ninguna directiva no resaltará ninguna parte del texto. Éstas son códigos especiales que están contenidos en un documento y se usan caracteres especiales para delimitarlas. Por ejemplo, la directiva para iniciar un párrafo es: `<P>`

Por lo general, se agrupan por pares: uno inicia la acción, mientras su par la finaliza. La directiva para finalizar el párrafo es: `</P>`. Como se puede ver, el par de final de acción es el mismo que el de inicio incluyendo el caracter /

HTML no es sensible a mayúsculas; esto es, la directiva puede escribirse como `<P>` o `<p>`. Los caracteres `<` y `>` son códigos reservados. Si se quieren mostrar dichos caracteres dentro del texto, se debe utilizar el concepto de entidades que se verá en el siguiente punto.

## Caracteres Especiales

Son especificados utilizando el siguiente formato: al inicio de la entidad se marca con el caracter "&", mientras que el final de la entidad se marca con el caracter ";". Entre los dos caracteres citados se especificará el nombre de una entidad.

Los más comunes son:

CARÁCTER	CODIGO
<	<code>&amp;lt;</code>
>	<code>&amp;gt;</code>
&	<code>&amp;amp;</code>
á	<code>&amp;aacute;</code>
Á	<code>&amp;Aacute;</code>
ñ	<code>&amp;ntilde;</code>
Ñ	<code>&amp;Ntilde;</code>

---

<sup>3</sup> De internet se puede obtener una gran variedad de documentación referida a la codificación de archivos en formato html. Este capítulo ha sido desarrollado a partir del publicado en <http://www.eis.uva.es/GuiaHTML/introHTML.html>

## Estructura de un documento HTML

Los documentos HTML se dividen en las siguientes partes como se puede ver en el siguiente código:

```
<HTML>
  <HEAD>
  </HEAD>

  <BODY>
  </BODY>
</HTML>
```

La primera línea indica que es un documento HTML (no visible).	<HTML>
La segunda y tercera línea delimitan la cabecera (no visible).	<HEAD> </HEAD>
La cuarta y quinta línea delimitan la parte principal, o cuerpo, y es la que el usuario ve.	<BODY> </BODY>
La última línea indica que finaliza el documento HTML.	</HTML>

## Un documento inicial

### Título

No es visible dentro de la página web . Se hará dentro de la directiva HEAD y utilizando la directiva <TITLE>...</TITLE>

```
...
  <HEAD>
  <TITLE>Introduccion a HTML</TITLE>
  <HEAD>
  ...
```

### Cabeceras

Las cabeceras permiten dividir lógicamente el documento en secciones, subsecciones, etc. HTML prevee hasta seis niveles de cabeceras.

Las cabeceras aparecen solamente en el cuerpo del documento. Para especificar un texto cabecera de nivel requerido se puede escribir:

```
...
<H1>Cabecera de nivel 1</H1>      Más general
<H2>Cabecera de nivel 2</H2>      .
<H3>Cabecera de nivel 3</H3>      .
...
<H6>Cabecera de nivel 6</H6>      Más específico
```

El usuario verá:

Cabecera de nivel 1

Cabecera de nivel 2

Cabecera de nivel 3

Cabecera de nivel 6

## Párrafos

La mayor parte del texto que se escribe forma parte de un párrafo de texto. Para crear un párrafo, basta con escribir el texto que lo forma dentro de la directiva `<P>...</P>`. Además, en numerosas ocasiones es necesario forzar un final de línea en el documento formateado (un simple retorno de carro no es suficiente). Para ello, es suficiente con el uso de la directiva simple `<BR>` (no necesita la directiva homóloga de cierre)

```
...
<P>Este es mi primer párrafo.
  aunque escriba este texto en otra línea,
  seguirá perteneciendo al mismo párrafo.</P>
<P>Este es un segundo párrafo que origina una separación.
  Luego sigue en la<BR>
  otra línea.</P>
```

El usuario verá:

Este es mi primer párrafo. aunque escriba este texto en otra línea, seguirá perteneciendo al mismo párrafo.

Este es un segundo párrafo que origina una separación. Luego sigue en la otra línea.

## Texto con formato

Dentro de un documento HTML se puede indicar que un texto tenga un estilo o tipografía especial. Se utiliza `<B>` para negrita y `<I>` para itálica. El siguiente ejemplo, muestra una forma de enfatizar texto:

Por ejemplo:

```
<P>A continuación <B>negrita</B> y luego <I>italica</I>.</P>
```

Se verá:

A continuación **negrita** y luego *italica*.

Para centrar un texto (u otro objeto, imagen, tabla, ...) se utiliza la directiva `<CENTER> ... </CENTER>`

Por ejemplo:

```
<CENTER>Esta línea la hemos centrado </CENTER>
```

Se verá:

Esta línea la hemos centrado

## Divisiones horizontales

Las divisiones horizontales son usadas para formatear un documento en secciones.

Se debe evitar el uso de imágenes para crear el efecto de divisiones horizontales porque pueden no verse correctamente. La inserción de una división horizontal es muy simple. Se realizará insertando la directiva `<HR>` en el lugar donde se desee que aparezca la línea horizontal (esta directiva no necesita su homóloga de cierre).

## Primer ejemplo completo

El siguiente listado es un ejemplo completo de lo visto hasta aquí. Copie el siguiente código a un archivo de texto y grabándolo con la extensión "html" se podrá ver con un navegador. Desde un programa en C, puede generar estas líneas a un archivo de texto.

```
<HTML>

  <HEAD>
    <TITLE>Código del primer ejemplo</TITLE>
  </HEAD>

  <BODY>
    <H1>PRIMER EJEMPLO DE HTML</H1>
    <H2>Cabecera de nivel 2</H2>

    <P>Este documentos presenta ejemplos de directivas.</P>
    <HR>
    <P>A continuación <B>negrita</B> y luego <I>itálica</I>.</P>

    <CENTER>Esta línea está centrada. </CENTER>
  </BODY>
</HTML>
```

Se verá:

# PRIMER EJEMPLO DE HTML

## Cabecera de nivel 2

Este documentos presenta ejemplos de directivas.

---

A continuación **negrita** y luego *itálica*.

Esta línea está centrada.

Pruebe modificarlo para observar los efectos producidos.

## Listas

Una lista para HTML no es más que una agrupación o enumeración de elementos de información. Estas listas pueden anidarse.

Los miembros de la mesa de trabajo son:

- Sr. Chema Pamundi
- Sr. German Tequilla
- Sr. Carmelo Coton

Para hacer una lista como la anterior, se debe utilizar la directiva `<UL>...</UL>`; mientras que, para cada elemento de la lista debe utilizarse la directiva `<LI>...</LI>`:

```
<UL>
<LI>Sr. Chema Pamundi</LI>
<LI>Sr. German Tequilla</LI>
<LI>Sr. Carmelo Coton</LI>
</UL>
```

Para que la lista se vea enumerada, en vez de la directiva `<UL>...</UL>` utilizar `<OL>...</OL>`.

## Imágenes

Una de las funcionalidades más llamativas en HTML es la posibilidad de incluir imágenes dentro de un documento. Algunos formatos gráficos tienen soporte en modo nativo (son visualizados directamente por el navegador), mientras que otros requieren del concurso de programas externos.

No todos los archivos que contienen gráficos siguen la misma convención de almacenamiento. Existen varios formatos que permiten, entre otras cosas, comprimir en distinto grado la información. Los formatos más extendidos son: GIF (Graphics Interchange Format), JPEG (Joint Photographic Experts Group bitmap) y sus variantes (JPG, BMP, MP, XBM), TIFF (Tagged Image File Format), EPS (Encapsulated PostScript), o PCX (de Paintbrush).

Solo el formato GIF es soportado directamente por todos los visualizadores, mientras que el JPEG lo es por la mayoría. El formato GIF se basa en el sistema de compresión LZW, un algoritmo muy simple y que, por ello, no alcanza unas altas cotas de reducción. Este formato trabaja con un máximo de 256 colores (8 bits); para simular colores fuera de la paleta utiliza la técnica de dithering (aproximación del color por los vecinos más próximos).

El formato JPEG utiliza un algoritmo de compresión muchos más complicado que el utilizado por el GIF: los archivos resultantes son más pequeños, pero necesitan más tiempo para su descompresión. A diferencia del anterior formato, JPEG trabaja con 16.7 millones de colores.

Como norma general, diremos que se utilizará el formato GIF para iconos e imágenes pequeñas y JPEG para imágenes grandes o de calidad.

Para insertar una imagen en un documento HTML se utilizará la directiva simple

```
<IMG>
<IMG src="/icons/network.gif">
```

El usuario verá una imagen:



## Tablas

Al igual que las listas, las tablas son componentes dedicados a mejorar la visualización de datos tabulados.

Las tablas se especificarán siempre por filas; es decir, primero se escribirá la fila 1, después la fila 2, etc. La directiva que se utiliza para delimitar una tabla es <TABLE>...</TABLE>.

Cada fila se especifica con la directiva <TR>...</TR> y, dentro de ella, cada celda se especifica con la directiva <TD>...</TD>

La presencia de bordes en la tabla se especifica con el atributo **border** en la directiva <TABLE>. Con ello se logrará un borde de dimensión la unidad; si deseamos hacer el borde más grueso deberemos dar un valor numérico al atributo: **border=espesor**.

El título de la tabla es una cadena de caracteres delimitado por la directiva <CAPTION>...</CAPTION>.

Por último, cada cabecera de columna se especifica con la directiva <TH>...</TH>

Las directivas TR, TD y TH admiten dos atributos de centrado: VALIGN para el centrado vertical y ALIGN para el horizontal; donde los valores que pueden tomar son, TOP (superior), BOTTOM (inferior), MIDDLE (centrado vertical), RIGHT (derecha), LEFT (izquierda) y CENTER (centrado horizontal). La directiva TD admite WIDTH=ancho de la columna.

Por ejemplo:

```
<TABLE border>
<CAPTION> Ejemplo de tabla</CAPTION>
<TR><TH>Primera columna</TH><TH>Segunda columna</TH><TH>Tercera columna</TH></TR>
<TR><TD WIDTH=215>100, 3</TD><TD>1, 8</TD><TD>313, 1</TD></TR>
<TR ALIGN=RIGHT><TD>22, 7</TD><TD>200, 8</TD><TD>23, 1</TD></TR>
<TR ALIGN=CENTER><TD>8100, 3</TD><TD>1300, 5</TD><TD>4100, 1</TD></TR>
</TABLE>
```

El usuario verá:

Ejemplo de tabla

Primera columna	Segunda columna	Tercera columna
100,3	1,8	313,1
22,7	200,8	23,1
8100,3	1300,5	4100,1

## Colores

Además de las imágenes, el color hace a las páginas más vistosas. Para especificar colores dentro de una página, se utiliza el código RGB (rojo-verde-azul) con el cual podemos especificar distintas tonalidades de colores. Para especificar un color se utiliza el caracter # seguido de un valor hexadecimal de 6 dígitos; los dos primeros para el rojo, los dos centrales para el verde y los dos finales para el azul.

Podemos ver algunos ejemplos:

Código	#FF0000	#0000FF	#00FF00	#FFFF00	#9933CC	#FFFFFF	#666666	#000000
Color	Rojo	Azul	Verde	Amarillo	Morado	Blanco	Gris	Negro

Existen tres características a las que podemos aplicar color: el fondo de un documento, al texto de un documento y a texto específico dentro del documento:

Lugar	Ejemplos	Resultado
Fondo de un documento	<BODY BGCOLOR=#9933CC>	Fondo de color morado
Texto de un documento	<BODY TEXT=#00FF00>	Texto de color verde
Texto específico	<FONT COLOR=#FF0000>	Texto en rojo

## Segundo ejemplo completo

Ahora se ejemplifica un código en C++ que genera un archivo html.

```
#include<iostream>
#include<fstream>
#include <iomanip>

int main() {
    ofstream salida("salida.html");

    salida << "<HTML>" << endl;
    salida << "<BODY>" << endl;
    salida << "<H1>FUNCION SENO</H1>" << endl;
    salida << "<HR>" << endl;

    salida << "<TABLE border>" << endl;
    salida << "<CAPTION> <B>Resultado</B></CAPTION>" << endl;
    salida << "<TR><TH>X</TH><TH>Seno (X)</TH></TR>" << endl;

    float x = 0;
    while (x < 3.14) {
        salida << "<TR TR ALIGN=CENTER><TD WIDTH=100>" << x
            << "</TD><TD WIDTH=215>" << setprecision(4)
```

```
        << sin(x) << "</TD></TR>" << endl;
    x = x + 0.15;
}

salida << "</TABLE>" << endl;
salida << "</BODY>" << endl;
salida << "</HTML>" << endl;

salida.close();
}
```

# Directivas del preprocesador

Las directivas del preprocesador son órdenes que se incluyen dentro del código de los programas que no son instrucciones para el programa en sí, sino que son para el preprocesador. El preprocesador es ejecutado automáticamente por el compilador cuando se compila un programa en C++ y está a cargo de realizar las primeras verificaciones del código del programa.

Cada una de estas directivas debe ser especificada en una sola línea de código y no deben incluir punto y coma ; al final.

## #define

Al principio se trató acerca de una directiva del preprocesador: **#define**, que sirve para generar constantes definidas o macros, y cuya forma es la siguiente:

```
#define nombre valor
```

Su función es definir una macro llamada nombre que cuando sea encontrada en algún punto del código sea reemplazada por valor. Por ejemplo:

```
#define ANCHO_MAX 100
char str1[ANCHO_MAX];
char str2[ANCHO_MAX];
```

define dos cadenas para almacenar hasta 100 caracteres.

**#define** puede ser usado también para generar funciones macro:

```
#define mayor(a,b) a>b?a:b
int x=5, y;
y = mayor(x,2);
```

luego de la ejecución de este código **y** contendría **5**.

## #undef

**#undef** realiza la función inversa que **#define**. Lo que hace es eliminar de la lista de constantes definidas la que posea el nombre pasado como parámetro a **#undef**:

```
#define ANCHO_MAX 100
char str1[ANCHO_MAX];
#undef ANCHO_MAX
#define ANCHO_MAX 200
char str2[ANCHO_MAX];
```

## #ifdef, #ifndef, #if, #endif, #else y #elif

Estas directivas permiten descargar parte del código de un programa si una cierta condición no es cumplida.

### #ifdef - #ifndef

#### #ifdef

Permite que una sección de un programa sea compilada sólo si la constante definida que se especifica como parámetro ha sido definida, independientemente de su valor. Su uso es:

```
#ifdef nombre
// código
#endif
```

Por ejemplo:

```
#ifdef ANCHO_MAX
char str[ANCHO_MAX];
#endif
```

En este caso, la línea **char str[ANCHO\_MAX];** es sólo considerada por el compilador si la constante definida **ANCHO\_MAX** ha sido previamente definida, independientemente de su valor. Si no ha sido definida, el código encerrado no será incluido en el programa.

#### #ifndef

Reactualiza lo opuesto: el código entre la directiva **#ifndef** y la directiva **#endif** es compilado sólo si el nombre constante que ha sido especificado no ha sido definido. Por ejemplo:

```
#ifndef ANCHO_MAX
#define ANCHO_MAX 100
#endif
char str[ANCHO_MAX];
```

En este caso, si al llegar a esta parte del código la *constante definida* **ANCHO\_MAX** no ha sido definida, será definida con un valor de **100**. Si ya existe, mantendrá el valor que actualmente posee (debido a que la directiva **#define** no sería ejecutada).

Las directivas **#if**, **#else** y **#elif** (elif = else if) sirven para que la porción de código que las sigue sea compilada sólo si se cumple la condición específica. La condición únicamente sirve para evaluar expresiones constantes. Por ejemplo:

```
#if ANCHO_MAX>200
#undef ANCHO_MAX
#define ANCHO_MAX 200
```

```
#elif ANCHO_MAX<50
#undef ANCHO_MAX
#define ANCHO_MAX 50

#else
#undef ANCHO_MAX
#define ANCHO_MAX 100
#endif

char str[ANCHO_MAX];
```

observar como la estructura de directivas encadenadas **#if**, **#elif** y **#else** terminan con **#endif**.

## #line

Cuando se compila un programa y ocurre un error durante el proceso de compilación, el compilador muestra el error que ha ocurrido precedido por el nombre del archivo y la línea dentro de ese archivo donde el error ha ocurrido.

La directiva **#line** permite controlar ambas cosas, los números de línea dentro de los archivos y el nombre del archivo que se quiere que aparezca cuando un error ocurre. Se utiliza de la siguiente manera:

```
#line numero "nombre_archivo"
```

donde *numero* es el nuevo número de línea que será asignado a la siguiente línea de código. El número de línea de las líneas sucesivas será incrementado en uno a partir de ésta.

*Nombre\_archivo* es un parámetro opcional que sirve para reemplazar el nombre de archivo que será mostrado en caso de error hasta que otra directiva lo cambie nuevamente o hasta llegar al final del archivo. Por ejemplo:

```
#line 1 "asignando variables"
int a?;
```

este código generará un error que será mostrado como error en el archivo "asignando variables", línea **1**.

## #error

Esta directiva interrumpe el proceso de compilación cuando es encontrada devolviendo el error especificado como parámetro:

```
#ifndef __constanteDeError
#error Se requiere compilador C++
#endif
```

Este ejemplo interrumpe el proceso de compilación si la constante definida **\_\_constanteDeError** no está definida.

## #include

Esta directiva también ha sido usada asiduamente en otras secciones. Cuando el preprocesador encuentra una directiva **#include** la reemplaza por el contenido total del archivo especificado. Existen dos maneras de especificar un archivo para ser incluido:

```
#include "archivo"  
#include <archivo>
```

La única diferencia entre ambas expresiones es en las carpetas en que el compilador va a buscar los archivos. En el primer caso, en el que el archivo es especificado entre comillas, el compilador buscará por los archivos en la misma carpeta en la que se encuentra el archivo que se está utilizando, y sólo en caso de que no esté allí el compilador buscará en las carpetas por defecto (donde se encuentran los archivos de encabezamiento estándar).

En caso que se encierre el nombre del archivo entre signos de <> se busca al archivo directamente en las carpetas por defecto.

## #pragma

Esta directiva se usa para especificar diversas opciones para el compilador. Estas opciones son específicas de la plataforma y el compilador que se está utilizando. Se deberá consultar el manual o la referencia del compilador para más información acerca de los posibles parámetros a definir con **#pragma**.

# Bases numéricas

Estamos acostumbrados a usar números decimales para expresar cantidades. Esta nomenclatura que parece tan lógica puede no serlo para un romano de la Roma clásica. Para ellos cada símbolo que ponían para expresar un número siempre representaba el mismo valor:

I	1
II	2
III	3
IV	4
V	5

Si se presta atención todos los signos I siempre representan el valor 1 (uno) dondequiera que se sitúen, como el signo V siempre representa nuestro 5 (cinco). Sin embargo esto no ocurre en nuestro sistema decimal. Cuando escribimos el símbolo decimal 1 no siempre estamos hablando acerca del valor 1 (I en números romanos).

Por ejemplo:

1	I
10	X
100	C

En estos casos nuestro símbolo 1 no siempre posee el valor 1. Por ejemplo, en el segundo caso, el símbolo 1 representa el valor 10 y en el tercero, 1 representa el valor 100.

Otro ejemplo:

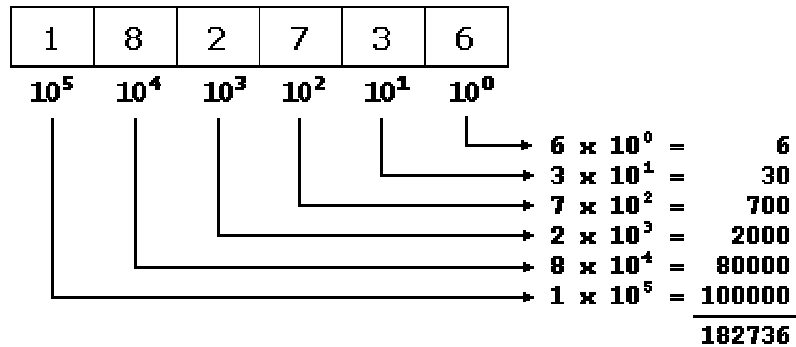
275

no es equivalente a  $2+7+5$ , sino que puede ser descompuesto como  $200+70+5$ :

$$\begin{array}{r} 200 \\ + 70 \\ 5 \\ \hline 275 \end{array}$$

por lo tanto, el primer símbolo 2 es equivalente a 200 ( $2 \times 100$ ), el segundo símbolo, 7 es equivalente a 70 ( $7 \times 10$ ) mientras que el último signo corresponde al valor 5 ( $5 \times 1$ ).

Toda la introducción previa puede ser representada matemáticamente en una forma muy simple. Por ejemplo, para representar el valor 182736 podemos asumir que cada dígito es el producto de sí mismo multiplicado por 10 elevado a la potencia que corresponde a su ubicación, comenzando desde la derecha con  $10^0$ , siguiendo con  $10^1$ ,  $10^2$ , y así:



## Números octales (base 8)

Como nuestros números "normales" son en base 10 porque tenemos 10 dígitos diferentes (del 0 al 9):

0123456789

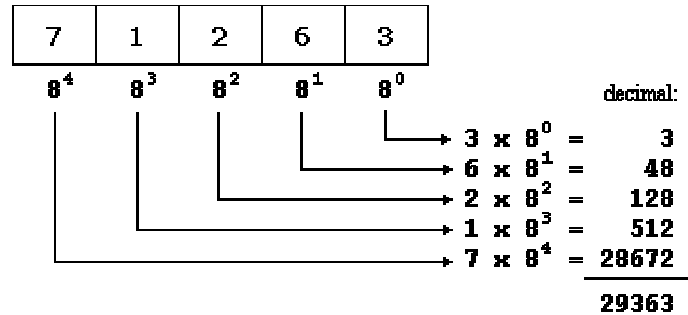
los números octales incluyen sólo dígitos del 0 al 7:

01234567

y por lo tanto, su base matemática es 8. En C++ los números octales tienen la peculiaridad de que siempre comienzan con un dígito 0. Veamos como escribiríamos los primeros números en base octal:

octal	decimal	
0	0	(cero)
01	1	(uno)
02	2	(dos)
03	3	(tres)
04	4	(cuatro)
05	5	(cinco)
06	6	(seis)
07	7	(siete)
010	8	(ocho)
011	9	(nueve)
012	10	(diez)
013	11	(once)
014	12	(doce)
015	13	(trece)
016	14	(catorce)
017	15	(quince)
020	16	(dieciseis)
021	17	(diecisiete)

podemos aplicar el esquema que vimos previamente con los números decimales simplemente considerando a 8 como base. Por ejemplo tomando el número octal 071263:



## Representaciones binarias

Los números en base octal y hexadecimal tienen una considerable ventaja sobre los números en base decimal en el mundo de los bits, y es que sus bases (8 y 16) son potencias de 2 ( $2^3$  y  $2^4$ ) lo que permite hacer conversiones más fáciles a números binarios que desde números decimales (cuya base es  $2 \times 5$ ). Por ejemplo, si quisiéramos pasar la siguiente secuencia binaria a otras bases:

```
1100111111010010100
```

para pasar esta secuencia a un número decimal necesitaríamos realizar una operación matemática similar a las hechas anteriormente para convertir desde hexadecimal o octal, lo cual nos daría el número decimal 212628.

Sin embargo, para pasar esta secuencia a un número octal sólo nos llevaría algunos segundos y lo podemos hacer sólo mirando: dado que 8 es  $2^3$ , separaremos el valor binario en grupos de 3 números:

```
110 011 111 010 010 100
```

y ahora sólo tenemos que traducir a un número en base octal cada grupo por separado:

```
110 011 111 010 010 100
 6   3   7   2   2   4
```

resultando el número 637224. el mismo proceso puede realizarse a la inversa para pasar desde octal a binario.

Para realizar esta operación con números hexadecimales debemos realizar el mismo proceso pero separando el valor binario en grupos de 4 números ( $16 = 2^4$ ):

```
11 0011 1110 1001 0100
 3   3   E   9   4
```

por lo tanto, la expresión binaria 1100111111010010100 puede ser representada en C++ como 212628 (decimal), 0637224 (octal) o 0x33e94 (hexadecimal).

El código hexadecimal es especialmente interesante en computación dado que hoy en día las computadoras están basadas en bytes compuestos de 8 bits binarios y por lo tanto cada byte es igual al rango que 2 números hexadecimales pueden representar. Por esta razón es el tipo más usado al representar valores traducidos desde binario

# Código ASCII

Como probablemente sabrás, al nivel más bajo las computadoras sólo manejan 0s (ceros) y 1s (unos). Usando secuencias de 0s y 1s una computadora puede manejar números en formato binario. Sin embargo no existe una forma evidente para representar letras con 0s y 1s. Para este propósito se usa el código ASCII (*Código Standard Americano para Intercambio de Información*).

El código ASCII es una tabla o lista que contiene todas las letras del alfabeto más una variedad de caracteres adicionales. En este código, cada caracter se representa con un número, que siempre es el mismo. Por ejemplo, el código ASCII para representar la letra A es siempre representada por el número 65, que es fácilmente representable usando 0s y 1s en notación binaria (1000001).

El código ASCII standard define 128 códigos de caracteres (de 0 a 127). Los primeros 32 son códigos de control (no imprimibles), y los otros 96 son caracteres representables

*	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	TAB	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	A	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	

\* Este panel está organizado para ser leído fácilmente en hexadecimal: los número de las filas representan el primer dígito y los números de las columnas representan el segundo dígito. Por ejemplo, el caracter A se localiza en la 4<sup>o</sup> fila y la 1<sup>o</sup> columna, así que sería representado como un número hexadecimal como 0x41 (65).

Además de los 128 códigos ASCII standard (los listados arriba en el rango de 0 a 127), muchas máquinas tienen otros 128 códigos extra cuyo formato se conoce como código ASCII extendido (en el rango de 129 a 255). Este conjunto de caracteres ASCII extendido depende de la plataforma, lo que significa que puede variar de una máquina a otra, o entre sistemas operativos.

Los conjuntos de caracteres extendidos ASCII más usados son OEM y ANSI.

El conjunto de caracteres OEM se incluye en todas las computadoras PC-compatibles como el conjunto de caracteres por defecto cuando el sistema bootea antes de cargar cualquier sistema operativo y bajo MS-DOS. Incluye algunos signos extranjeros,

algunos caracteres acentuados y también piezas para dibujar paneles simples. Desafortunadamente es redefinido usualmente para configuraciones regionales específicas para incluir símbolos locales.

### OEM Extended ASCII

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8	ç	ü	é	â	ä	à		ç	ê	ë	è	ï	î	ì	ñ	
9	é	æ		ô	ö	ò	û	ù	ÿ	ö	Ü	ç	£	¥		f
A	á	í	ó	ú	ñ											
B																
C																
D																
E																
F																

El conjunto de caracteres estándares de ANSI lo incorporan los sistemas como Windows, algunas plataformas UNIX y varias otras aplicaciones. Incluye otros simbolos y letras acentuada que pueden ser usadas sin necesidad de ser redefinidas en otros lenguajes:

### ANSI Extended ASCII (Windows)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8			,	f	//	...	†	‡	ˆ		Š	<				
9												>				
A																
B																
C																
D																
E																
F																

# Lógica booleana

## Operaciones AND, OR, XOR y NOT

Un bit es la mínima cantidad de información que podemos imaginar, dado que sólo almacena valores de 1 o 0, que representan SI o NO, activado o desactivado, verdadero o falso, etc... esto es: dos estados posibles, cada uno opuesto al otro, sin posibilidad de intermedios.

Muchas operaciones pueden ser realizadas con bits, tanto en conjunto con otros bits o sin ellos. Estas operaciones reciben el nombre de **operaciones booleanas**, una palabra que viene del nombre de uno de los matemáticos que más contribuyó en este campo:: George Boole (1815-1864).

Todas estas operaciones tienen un comportamiento predeterminado y todas ellas pueden ser aplicadas a cualquier bit cualquiera sea el valor que contenga (**0** o **1**). A continuación tienes una lista de las operaciones booleanas básicas y una tabla con el comportamiento de esa operación con todas las posibles combinaciones de bits.

### AND (&)

Esta operación se realiza entre dos bits distintos (a y b). El resultado es 1 si ambos a y b son iguales a 1, si alguno de ellos es igual a 0 el resultado es 0

a	b	a&b
0	0	0
0	1	0
1	0	0
1	1	1

### OR (|)

Esta operación se realiza entre dos bits distintos (a y b). El resultado es 1 si alguno de ellos a o b es 1. Si ninguno es 1 el resultado es 0

a	b	a b
0	0	0
0	1	1
1	0	1
1	1	1

## XOR (Or exclusivo: ^)

Esta operación se realiza entre dos bits distintos (a y b). El resultado es 1 si alguno de ellos a o b es 1, excepto si ambos son 1. Si ninguno de los dos o ambos son iguales a 1 el resultado es 0

a	b	a^b
0	0	0
0	1	1
1	0	1
1	1	0

## NOT (~)

Esta operación se realiza en un sólo bit. Su resultado es la inversión del valor actual del bit: si contenía 1 pasa a contener 0 y si contenía 0 pasa a contener 1

a	~a
0	1
1	0

Estas son las cuatro operaciones básicas (**AND**, **OR**, **XOR** y **NOT**). Combinando estas operaciones cualquier resultado deseado puede ser obtenido.

En C++, estas operaciones pueden ser usadas entre dos variables de cualquier tipo entero; la operación lógica se realiza entre los bits de las dos variables. Por ejemplo, suponiendo dos variables: a y b, ambas de tipo char, a conteniendo 195 (11000011 en binario) y b conteniendo 87 (o 01010111 en binario). Si escribimos el siguiente código:

```
char a = 195;
char b = 87;
char c;
c = a&b;
```

Lo que hicimos fue una operación de bits **AND** entre **a** y **b**. El resultado (el contenido de **c**) sería el siguiente:

```

a :   1 1 0 0 0 0 1 1
b :   0 1 0 1 0 1 1 1
-----
c :   0 1 0 0 0 0 1 1
```

&

01000011, que es 67.

# Archivos de encabezamiento estándar

En ANSI-C++ la forma de incluir archivos de encabezamiento desde la biblioteca standard ha cambiado.

El standard especifica la siguiente modificación a la manera C de incluir archivos de encabezamiento standard:

Los archivos de encabezamiento no mantienen la extensión **.h** típica del lenguaje C y de los compiladores C++ previos al standard, como en el caso de **stdio.h**, **stdlib.h**, **iostream.h**, etc. Esta extensión **.h** simplemente desaparece y los archivos antes conocidos como **iostream.h** se convierten en **iostream** (sin **.h**).

Los archivos de encabezamiento que vienen del lenguaje C ahora deben ser precedidos por un caracter **c** para distinguirlos de los nuevos archivos de encabezamiento exclusivos de C++ que tienen el mismo nombre. Por ejemplo **stdio.h** se convierte en **cstdio**.

Todas las clases y funciones definidas en las bibliotecas standard están bajo el namespace **std** en vez de ser globales. Estos no se aplican a las macros C que permanecen igual.

La siguiente lista corresponde a los archivos de encabezamiento standard de C++:

```
<algorithm> <bitset> <deque> <exception> <fstream> <functional>
<iomanip> <ios> <iosfwd> <iostream> <istream> <iterator> <limits>
<list> <locale> <map> <memory> <new> <numeric> <ostream> <queue>
<set> <sstream> <stack> <stdexcept> <streambuf> <string> <typeinfo>
<utility> <valarray> <vector>
```

Y aquí hay una lista de los archivos de encabezamiento standard de C incluidos en el ANSI-C++ con el nuevo nombre y sus equivalentes en ANSI-C:

ANSI-C++	ANSI-C
<cassert>	<assert.h>
<cctype>	<ctype.h>
<cerrno>	<errno.h>
<cfloating>	<float.h>
<ciso646>	<iso646.h>
<climits>	<limits.h>
<clocale>	<locale.h>
<cmath>	<math.h>
<csetjmp>	<setjmp.h>
<csignal>	<signal.h>
<cstdarg>	<stdarg.h>
<cstddef>	<stddef.h>

<cstdio>	<stdio.h>
<cstdlib>	<stdlib.h>
<cstring>	<string.h>
<ctime>	<time.h>
<wchar>	<wchar.h>
<ctype>	<ctype.h>

Desde ahora las clases y funciones de las bibliotecas se localizan dentro del namespace `std` y debemos usar la directiva C++ `using` para que estos se vuelvan útiles en la misma manera que lo eran antes. Por ejemplo, para poder utilizar todas las clases standard de `iostream` tendríamos que incluir algo similar a esto:

```
#include <iostream>
using namespace std;
```

que sería equivalente a la vieja expresión

```
#include <iostream>
```

previa al estándar.

Sin embargo para compatibilidad con ANSI-C, el uso de la forma `name.h` para incluir archivos de encabezamiento C está permitida. Por lo tanto, los siguientes ejemplos son válidos y equivalentes en un compilador que cumple con las especificaciones ANSI-C++:

```
// Ejemplo ANSI C++
#include <cstdio>
using namespace std;
int main ()
{
    printf ("Hola Mundo!");
    return (0);
}
```

```
// Ejemplo pre-ANSI C++
// también válido bajo ANSI C++
#include <stdio.h>
int main ()
{
    printf ("Hola Mundo!");
    return (0);
}
```

# Listado de páginas del lenguaje en Internet

Este capítulo recopila algunos links a páginas de internet. Estos links se pueden realizar automáticamente a partir de la versión electrónica de este documento.

## Los más recomendados

[Thinking in C++](#), (Bruce Eckel)

<http://www.cplusplus.com/doc/tutorial/> (Es el tutorial de C++ que se utilizó como base para varios capítulos de este documento).

## Sitios en español

<http://www.esi.unav.es/asignaturas/Informat1/AyudaInf/aprendainf.htm> (Muchos manuales de informática en español)

[C++ \(665 Kbytes\)](#) (Manual en español de C++)

[ANSI C \(451 Kbytes\)](#) (Manual en español de C)

<http://c.conclase.net/curso/indice.html> (Esta en español)

<http://delta.cs.cinvestav.mx/~oolmedo/CBR/CCPP.PDF> (El lenguaje C++ Concurrente).

<http://www.zator.com/Cpp/index.htm> (Curso de C++)

<http://www.geocities.com/dreamlfp315/dll/cpp.htm> (Lenguaje C++)

<http://www.abcdatos.com/tutoriales/programacion/c.html> (Varios Tutoriales)

[Area 52 Escom](#) - Tutorial de básico con datos adicionales sobre las redes y Linux.

[Club Builder](#) - Noticias y tutoriales sobre Firebird y Borland C++ Builder.

[Con Clase](#) - Para aprender C y C++: con un grupo de correo y tutoriales.

[Curso de Introducción al lenguaje C](#)

[Curso de Programación en C](#) - Temario sobre C.

[Introducción al lenguaje C](#) - Por José Miguel Santos Espino.

[Lenguaje C](#) - Pequeños programas para familiarizarse con él.

[Lenguaje C, El](#) - Conceptos básicos, instrucciones de control, parametrización, gestión de ficheros y más.

[Lenguaje C/C++](#) - Prácticas y ejemplos para aprender a programar en este lenguaje.

[Lenguaje C: Curso de Iniciación](#)

[Manual de C++](#)

[Programación en C](#) - Variables y constantes, operadores, funciones y punteros.

[Programación en C bajo Dos](#) - Compiladores, tutoriales, programas en desarrollo, codigos fuente, etc...

[Rincón del C, El](#) - Cursos de C, código fuente y compiladores.

[Rincón del programador, El](#) - Códigos fuente, ejemplos, consejos y trucos en Visual C++ para desarrolladores en Windows, tanto iniciados como expertos.

[Solucionario Al Libro de Jorge Villalobos](#) - Informes dirigidos a estudiantes de ingeniería de sistemas con ejercicios resueltos de programación en lenguaje C.

[Usenet - es.comp.lenguajes.c](#)

[Usenet - es.comp.lenguajes.c++](#)

## Sitios en inglés listado por temas

- **Para comenzar**

- [An Introduction to C++](#) (Saveen Reddy & G. Bowden Wise)
- [Objective View Point](#) (ACM)
- [The C++ Object Model](#) (NCITS H7)
- [Understanding C++: An Accelerated Introduction](#) (Marshall Brain & Kelly Campbell)

- **Ejemplos**

- [Callbacks in C++ - Simple Solution](#) (Paul Jakubik)
- [C++ Pitfalls](#) (Cay S. Horstmann)
- [C++ Standard Library Examples](#) (Nicolai M. Josuttis.)
- [The Generic Algorithms](#) (Stanley Lippman)

- **Cosas interesantes**

- [Guru of the Week](#) (Herb Sutter)
- [99 Bottles of Beer](#) (Craig Schroeder)

- **Tutoriales**

- [Advanced C++ Tutorials](#) (About.com)
- [Beginning C++ Tutorials](#) (About.com)
- [C++ Tutorials](#) (1001tutorials.com)
- [C++ Tutorials](#) (Glen McCluskey & Associates)
- [DevCentral Tutorials](#) (Interface Technologies)
- [The Bits Tutorials](#) (Jon Jenkinson et al.)
- [Tutorials](#) (Open Directory)
- [Tutorials and References](#) (Tina I. Seawell)
- [An Introduction to C++ Programming](#) (Björn Fahller)
- [C++ Annotations](#) (Frank B. Brokken & Karel Kubat)
- [C++ in Hypertext](#) (Curtis Solloway)
- [C++ Programming](#) (Valencia Community College)
- [C++ Programming Language Tutorials](#) (Douglas C. Schmidt)
- [Crash Course on STL](#) (Osvaldo Pinali Doederlein)
- [Course on OO Software Construction](#) (Joe Sant)
- [Crashproof C++](#) (Steve Litt)
- [C++ tutorial for C users](#) (Eric Brasseur)
- [Einführung in C++ \[German\]](#) (Universität Siegen)
- [From The Ground Up: A Guide to C++](#) (?)
- [Learn C/C++ Today](#) (Vinit Carpenter)
- [Neil's C++ Stuff](#) (Neil C. Obremski)
- [Online C++ Tutorial](#) (David Wegman)
- [OWL HOW](#) (Tenermerx Software)
- [Pointers](#) (Todd A. Gibson)
- [STL Tutorial](#) (Phil Ottewell)
- [Understanding C++: An Accelerated Introduction](#) (Marshall Brain & Kelly Campbell)
- [Well-mannered oo design in C++](#) (Taligent)

- **Preguntas frecuentes**

- [comp.lang.c++. FAQs List](#) (MIT)
- [comp.lang.c++. FAQs List](#) (Ohio State University)

- [comp.lang.c++.newsgroup.faq](#) (The Internet FAQ Consortium)
- [FAQs](#) (Miller Freeman)
- [FAQs](#) (Open Directory)
- [\[alt.comp.lang.learn.c-c++\] - FAQ list](#) (Sunil Rao, Rich Churcher)
- [comp.std.c++.faq](#) (Matt Austern et al.)
- [C/C++ FAQs](#) (C/C++ Users Journal)
- [C++ FAQ Lite](#) (Marshall Cline)
- [C++ Language Topic Area](#) (Experts Exchange)
- [C/C++ Users Group FAQ](#) (CUG)
- [Bjarne Stroustrup's FAQs](#) (Bjarne Stroustrup)
- [DevCentral Q & A](#) (Interface Technologies)
- [Embedded C++ FAQs](#) (Embedded C++ Technical Committee)
- [Numerical Analysis & Associated Fields Resource Guide FAQs](#) (S. J. Sullivan)
- [Platform-Independent Graphical User Interface Development Kits FAQs](#) (Ross McKay & Wade Guthrie)
- **Glosario**
  - [C++ Glossary](#) (Glen McCluskey & Associates)
  - [VisualAge Developer Domain Glossary](#) (IBM)
- **Soporte**
  - [Programming advices for C++](#) (Valentin Bonnard)
  - [Ask the C++Pros](#) (Fawcette Technical Publications)
  - [Tech Tips](#) (DevX)
- **Referencias**
  - [The Dinkum C++ Library Reference](#) (Dinkumware)
- **Standards**
  - [Standards](#) (Open Directory)
  - [J16 of National Committee for Information Technology Standards](#) (NCITS)
  - [JTC1/SC22/WG21 - C++ of ISO](#) (International Organization for Standardization) / [IEC](#) (International Electrotechnical Commission) (ISO/IEC)
  - [C++ - Beyond the ARM](#) (OCS)
  - [C++ DIN Arbeitskreis \[German\]](#) (Nicolai M. Josuttis)
  - [C++ Draft - Hypertext Summary of the Syntax](#) (Dick Botting)
  - [C++ Standards - What are they? Why are they important?](#) (Microsoft)
  - [The ISO/ANSI C++ Draft](#) (Cygnus Solutions)
  - [Embedded C++](#) (Embedded C++ Technical Committee)
  - [Embedded C++](#) (Embedded C++ Technical Committee)
  - [The Embedded C++ Programming Guide Lines](#) (Embedded C++ Technical Committee)
  - [The Language Specification & Libraries](#) (Embedded C++ Technical Committee)
  - [C and C++ Style Guides](#) (Christopher Lott)
  - [C ++ Coding Conventions](#) (Will Pitt)
  - [C++ Coding Standard](#) (Todd Hoff)
  - [C++ Coding Standards](#) (CoreLinux Consortium)
  - [Coding Standards for C, C++, and Java](#) (Jeff Johnson)
  - [C++ programming conventions](#) (Taligent)
  - [C++ Programming Style](#) (Wildfire Communications)

- [Programming in C++, Rules and Recommendations](#) (Ellemtel Telecommunication Systems Laboratories)
- [Ottinger's Rules for Variable and Class Naming](#) (Tim Ottinger)
- **Newsgroups generales**
  - [Newsgroups about C++](#) (Niklas Olsson)
  - [comp.lang.c++](#)
  - [comp.lang.c++ Resources](#) (PHOAKS)
  - [comp.lang.c++.moderated](#)
  - [comp.lang.c++.moderated Resources](#) (PHOAKS)
  - [comp.std.c++](#)
  - [comp.std.c++ Resources](#) (PHOAKS)
  - [Deja News](#) (Deja News)
  - [Reference.COM](#) (InReference)
- **Newsgroups especiales**
  - [comp.lang.c++.leda](#)
  - [comp.lang.c++.leda Resources](#) (PHOAKS)
  - [comp.os.ms-windows.programmer.tools.owl](#)
  - [gnu.g++](#)
  - [gnu.g++.announce](#)
  - [gnu.g++.bug](#)
  - [gnu.g++.help](#)
  - [gnu.g++.lib.bug](#)
  - [gnu.gdb.bug](#)
- **Newsgroups Naciones**
  - [de.comp.lang.iso-c++](#)
  - [Offizielle FAQ zu de.comp.lang.iso-c++ \[German\]](#) (LogicTools)
  - [es.comp.lenguajes.c++](#)
  - [fj.lang.c++](#)
  - [fr.comp.lang.c++](#)
  - [han.comp.lang.c++](#)
  - [it.comp.lang.c++](#)
- **Newsgroups de Compañías**
  - DevX
    - [c++.announcements](#)
    - [c++.general](#)
    - [c++.getting.started](#)
    - [c++.windows.development](#)
  - IBM
    - [VisualAge C++ Newsgroups](#) (IBM)
    - [ibm.software.vagen](#)
    - [ibm.software.vacpp.acm](#)
    - [ibm.software.vacpp.beta](#)
    - [ibm.software.vacpp.builders](#)
    - [ibm.software.vacpp.compiler](#)
    - [ibm.software.vacpp.debugger](#)
    - [ibm.software.vacpp.ide](#)
    - [ibm.software.vacpp.misc](#)
    - [ibm.software.vacpp.non-technical](#)
    - [ibm.software.vacpp.openclass](#)
    - [ibm.software.vacpp.os390](#)

- [ibm.software.vacpp.os390.compiler](http://ibm.software.vacpp.os390.compiler)
- [ibm.software.vacpp.os390.toolsbeta](http://ibm.software.vacpp.os390.toolsbeta)
- [ibm.software.vacpp.tools](http://ibm.software.vacpp.tools)
- Inprise / Borland
  - [Borland / Inprise Newsgroups \(Inprise\)](#)
  - [borland.public.announce](http://borland.public.announce)
  - [inprise.public.announce](http://inprise.public.announce)
  - [inprise.public.as400.cppbuilder](http://inprise.public.as400.cppbuilder)
  - [inprise.public.corba.cppbuilder](http://inprise.public.corba.cppbuilder)
  - [inprise.public.visibroker](http://inprise.public.visibroker)
  - [borland.public.cpp](http://borland.public.cpp)
  - [borland.public.cpp.announce](http://borland.public.cpp.announce)
  - [borland.public.cpp.commandlinetools](http://borland.public.cpp.commandlinetools)
  - [borland.public.cpp.ide](http://borland.public.cpp.ide)
  - [borland.public.cpp.jobs](http://borland.public.cpp.jobs)
  - [borland.public.cpp.language](http://borland.public.cpp.language)
  - [borland.public.cpp.non-technical](http://borland.public.cpp.non-technical)
  - [borland.public.cpp.owl](http://borland.public.cpp.owl)
  - [borland.public.cpp.thirdpartytools](http://borland.public.cpp.thirdpartytools)
  - [borland.public.cpp.winapi](http://borland.public.cpp.winapi)
  - [borland.public.install.bcphp](http://borland.public.install.bcphp)
  - [Borland C++Builder Newsgroups \(Inprise\)](#)
  - [borland.public.cppbuilder](http://borland.public.cppbuilder)
  - [borland.public.cppbuilder.activex](http://borland.public.cppbuilder.activex)
  - [borland.public.cppbuilder.announce](http://borland.public.cppbuilder.announce)
  - [borland.public.cppbuilder.commandlinetools](http://borland.public.cppbuilder.commandlinetools)
  - [borland.public.cppbuilder.database](http://borland.public.cppbuilder.database)
  - [borland.public.cppbuilder.database.desktop](http://borland.public.cppbuilder.database.desktop)
  - [borland.public.cppbuilder.database.multi-tier](http://borland.public.cppbuilder.database.multi-tier)
  - [borland.public.cppbuilder.database.sqlservers](http://borland.public.cppbuilder.database.sqlservers)
  - [borland.public.cppbuilder.graphics](http://borland.public.cppbuilder.graphics)
  - [borland.public.cppbuilder.ide](http://borland.public.cppbuilder.ide)
  - [borland.public.cppbuilder.internet](http://borland.public.cppbuilder.internet)
  - [borland.public.cppbuilder.jobs](http://borland.public.cppbuilder.jobs)
  - [borland.public.cppbuilder.language](http://borland.public.cppbuilder.language)
  - [borland.public.cppbuilder.non-technical](http://borland.public.cppbuilder.non-technical)
  - [borland.public.cppbuilder.thirdpartytools](http://borland.public.cppbuilder.thirdpartytools)
  - [borland.public.cppbuilder.vcl](http://borland.public.cppbuilder.vcl)
  - [borland.public.cppbuilder.winapi](http://borland.public.cppbuilder.winapi)
  - [borland.public.install.cppbuilder](http://borland.public.install.cppbuilder)
  - [inprise.public.as400.cppbuilder](http://inprise.public.as400.cppbuilder)
  - [inprise.public.corba.cppbuilder](http://inprise.public.corba.cppbuilder)
- Powersoft
  - [End of Life Notice for Power++ Enterprise \(Sybase\)](#)
  - [powersoft.public.power++.database](http://powersoft.public.power++.database)
  - [powersoft.public.power++.datawindow](http://powersoft.public.power++.datawindow)
  - [powersoft.public.power++.docs](http://powersoft.public.power++.docs)
  - [powersoft.public.power++.general](http://powersoft.public.power++.general)
  - [powersoft.public.power++.ole-activex](http://powersoft.public.power++.ole-activex)
  - [powersoft.public.watcom\\_c\\_c++.general](http://powersoft.public.watcom_c_c++.general)

- **Listas de correo**

- [CodeCraft](#) (Darren Collins)
- [Discussion List](#) (The Bits Editors)
- [IBM Application Development e-News](#) (IBM)
- [Inprise & Borland International Listserver](#) (Inprise)
- [Inprise Newsletters](#) (Inprise)
- [MFC@LISTSERV.MSN.COM](#) (L-Soft international)
- [The Object Windows Library Mailing List](#) (G. Bowden Wise)

- **Foros**

- [EarthWeb Discussions: earthweb.c\\_plus\\_plus.general](#) (EarthWeb)
- [Items in c++.announcements](#) (DevX)
- [Items in c++.general](#) (DevX)
- [Items in c++.getting.started](#) (DevX)
- [Items in c++.windows.development](#) (DevX)

- **Chats**

- [#C++ Website Webchat](#) (AFE)

- **Artículos en general**

- Articles Collections
  - [Magazine Publications](#) (Scott Meyers)
  - [publications::articles](#) (Object Mentor)
- Articles
  - [C++ Annotations](#) (Frank B. Brokken / Karel Kubat)
  - [C++ Critique /3rd ed.](#) (Ian Joyner)
  - [C++ Gets Faster for Scientific Computing](#) (Kuck & Associates)
  - [C++ in the Real World: Advice from the Trenches](#) (Nathan C. Myers)
  - [Posting to comp.lang.c++](#) (Bjarne Stroustrup)

- **Artículos especiales**

- [A First Look at C++ Program Analyzers](#) (Scott Meyers & Martin Klaus)
- [An Introduction to the C++ Standard Library](#) (ObjectCraft)
- [Automating Design-Pattern Identification](#) (Dr. Dobb's Journal)
- [Casting in C++: Bringing Safety and Smartness to Your Programs](#) (G. Bowden Wise)
- [C++Builder 3 - A Customer and Orders Master Detail Web Server Application](#) (Inprise)
- [C++ Interfaces](#) (Dr. Dobb's Journal)
- [Combining OO Design and Generic Programming](#) (Angelika Langer & Klaus Kreft)
- [Connective C++](#) (Quintessent Technologies)
- [Contracts: From Analysis to C++ Implementation](#) (Plösch, Reinhold; Pichler, Josef)
- [Copy Constructors](#) (About.com)
- [C++ Programmers - Two Faces, One Language - Building a Web Server Application](#) (Inprise)
- [Data Abstraction in C++](#) (About.com)
- [Dynamische Allokation von Ressourcen](#) (Angelika Langer)
- [Extending the iostream library](#) (Cay S. Horstmann)
- [Handling Exceptions in C and C++: Part 1, Part 2, Part 3, Part 4, Part 5, Part 6, Part 7, Part 9, Part 9, Part 10](#) (Microsoft)

- [Integrating Structured Exception Handling in the C++ exception mechanism](#) (Ruurd Pels)
- [Internationalization Using Standard C++](#) (Angelika Langer & Klaus Kreft)
- [Iterators in the Standard C++ Library](#) (Angelika Langer & Klaus Kreft)
- [Keeping interfaces and implementations separate](#) (Dr. Dobb's Journal)
- [Linked Lists](#) (Fawcette Technical Publications)
- [Makefile References on the Web](#) (Tina I. Seawell)
- [MVP: Model-Viewer-Presenter - The Taligent Programming Model For C++ and Java](#) (IBM/Taligent)
- [Program for Change - Use interfaces in your C++ programs to simplify maintenance](#) (Visual C++ Developers Journal)
- [Reengineering Legacy C and C++ Applications](#) (Peter L. Bird & F. Andy Seidl)
- [References - Part I of III](#) (About.com)
- [References - Part II of III](#) (About.com)
- [References - Part III of III](#) (About.com)
- [Techniques For Cross-Platform Development](#) (Miller Freeman, Inc.)
- [The Beauty and Power of C++](#) (Elliott Coates)
- [The Object Ownership Method - Efficient C++ Memory Management](#) (Applied Microsystems)
- [Thread-Recycling in C++ \[German\]](#) (SIGS)
- **Interoperabilidad**
  - [C++ and JAVA](#) (About.com)
  - [Integrating Java with C++](#) (Bill Foote)
  - [Java Cookbook: Porting C++ to Java](#) (IBM)
- **Sitios relacionados**
  - [Absolute Power++](#) (Bill Klein)
  - [Ask the C++ Pro](#) (Fawcette Technical Publications)
  - [C++](#) (About.com)
  - [C++ Zone](#) (DevX)
  - [CodeGuru](#) (CodeGuru)
  - [Developer Resources](#) (Genitor)
  - [Dr. Bob's C++Builder Gate](#) (Bob Swart)
  - [Intel Architecture Performance Training Center](#) (Intel)
  - [Internet Parallel Computing Archive](#) (University of Kent at Canterbury)
  - [Neil's C++ Stuff](#) (Neil C. Obremski)
  - [Parallel Programming with C++](#) (Bernd Mohr)
  - [The Bits ... The C++Builder Information and Tutorial Site](#) (The Bits Editors)
  - [The C++ Virtual Library](#) (DESY)
  - [The OO Numerics Page](#) (Todd Veldhuizen.)
  - [The Unofficial Borland C++Builder Home Page](#) (Brian Sturk)
  - [This is #C++....](#) (Morph)
  - [VisualAge Developer Domain](#) (ibm)
- **Colección de links**
  - [Borland C++ Builder 4](#) (About.com)
  - [C/C++](#) (developer.com)
  - [C/C++](#) (Karim Ratib)
  - [C++](#) (Mecklermedia)
  - [C++ and related topics](#) (Angelika Langer)

- [C++ Archive](#) (Quadralay Corporation)
- [C++Builder](#) (Open Directory)
- [C++Builder Sites](#) (Inprise)
- [C++Builder Sites](#) (The Bits Editors)
- [C++ Documentation](#) (Martin Brown)
- [C++ Links](#) (Brad Appleton)
- [C++ Links and Reviews](#) (DevX)
- [Cross Platform Development Internet Resources](#) (R2M Software)
- [C / C++ Zone](#) (Programmers' Heaven)
- [Embedded Systems](#) (EG3)
- [Internet Sites and Files of Interest to C++ Users](#) (Robert Davies)
- [Links](#) (EarthWeb)
- [Oli's C/C++-Links \[German\]](#) (Oliver Böhm)
- [Open Directory](#)
- [Other STL Web Sites](#) (Silicon Graphics)
- [Powersoft Power++ Resources](#) (Bill Klein)
- [Programmer's Oasis](#) (Simo Salminen)
- [The C++ Builder Programmer's Ring](#) (Mark Cashman)
- [The C++Builder Web Ring](#) (ZBuilder Software)
- [Viper's C/C++ Web Page](#) (?)
- [WWW C++ Information](#) (Forschungszentrum Juelich)
- [YAHOO I](#)
- [YAHOO II](#)
- [YAHOO III](#)
- [YAHOO IV](#)
- **Bibliografías**
  - [C++ Report Columns on Distributed Object Computing](#) (Douglas C. Schmidt)
  - [OO Papers and Articles](#) (Markus Frank)
  - [Publications by Bjarne Stroustrup](#) (Bjarne Stroustrup)
- **Libros**
  - [Best Sellers](#) (Computer Literacy)
  - [Books by Bjarne Stroustrup](#) (Bjarne Stroustrup)
  - [Books on C++](#) (CERN)
  - [C++](#) (Addison Wesley Longman)
  - [C/C++ Books](#) (Computer Literacy)
  - [C & C++ / General](#) (Amazon.com)
  - [C & C++ / Language](#) (Amazon.com)
  - [C & C++ / Objects](#) (Amazon.com)
  - [C & C++ / Programming](#) (Amazon.com)
  - [C/C++ Programming Center](#) (O'Reilly)
  - [C & C++ / Templates](#) (Amazon.com)
  - [C & C++ / Tutorials](#) (Amazon.com)
  - [C/C++ Users Group Bookstore](#) (CUG)
  - [Free Books](#) (About.com)
  - [Inprise C++Builder Books](#) (Inprise)
  - [Reviews Section](#) (The Association of C & C++ Users)
  - [The Bits Books Index](#) (The Bits Editors)
  - [The Essential C/C++ Reading List](#) (C/C++ Users Journal)
  - [Our Recommendations: C++](#) (Computer Literacy)

- [Thinking in C++](#), (Bruce Eckel)
- **Revistas**
  - [C++Builder Developer's Journal](#) (The Cobb Group)
  - [C++ Newsletters](#) (Glen McCluskey & Associates)
  - [C++ Report Online](#) (SIGS)
  - [C/C++ Users Journal](#) (Miller Freeman)
  - [ISDF Newsletter - International Standards Development Forum](#) (ACCU)
  - [The Development Exchange](#) (FTP)
  - [The Intel Software Performance Newsletter](#) (Intel)
  - [Visual Systems Journal](#) (Bearpark Publishing)
- **Organizaciones**
  - [Association of C and C++ Users](#)
  - [C++ and C SIG](#) (New York PC Users Group)
  - [C/C++ Users Group](#)
  - [C++ Industrial Seminar Group](#) (Marian Corcoran)
  - [CPG Homepage](#) (United Programmers)
  - [Embedded C++ Technical Committee](#) (?)
- **Proyectos**
  - [CoreLinux++](#) (CoreLinux Consortium)
- **Personajes especiales**
  - [Gurus](#) (Genitor)
  - [Jim Coplien](#)
  - [Cay Horstmann](#)
  - [Andrew Koenig](#)
  - [Angelika Langer](#)
  - [Scott Meyers](#)
  - [Ira Pohl](#)
  - [Bjarne Stroustrup](#)
- **Ambientes de desarrollo**
  - [C++ Compiler Comparison Chart](#) (The Internet Group)
  - [Borland C++](#) (Inprise / Borland)
  - [Borland C++Builder](#)
    - [Books](#) (Inprise)
    - [Developer Support](#) (Inprise)
    - [Feature Matrix](#) (Inprise)
    - [Features & Benefits](#) (Inprise)
    - [FTP Sites for Inprise, Borland, and VisiBroker Products](#) (Inprise)
    - [Previous Versions](#) (Inprise)
    - [Tool and Component Builders](#) (Inprise)
    - [White Papers](#) (Inprise)
    - [ZD Net Tips & Articles](#) (Inprise)
    - [Application Development with C++Builder and Delphi](#) (Inprise)
    - [Bold for C++Builder](#) (BoldSoft)
    - [Borland C++Builder for Borland Delphi Users](#) (The Delphi Magazine)
    - [Borland C++Builder 4 is here!](#) (Ruurd Pels)
    - [C++Builder and VCL](#) (Inprise)
    - [C++Builder Developer's Journal](#) (Inprise)
    - [C++Builder 4 Enterprise Tour](#) (Inprise)
    - [Changes made to ATL 2.0 in C++Builder 4.0](#) (Inprise)
    - [COM Automation in BCB4](#) (Inprise)

- [Introduction to the Database Components in Borland C++Builder](#) (Inprise)
- [Lost keys in the IE5 ActiveX controls](#) (Inprise)
- [New Features Matrix](#) (Inprise)
- [Simple CORBA Servers in C++Builder 4.0](#) (Inprise)
- [SQL Access in C++Builder](#) (Inprise)
- [Cray C++](#) (Silicon Graphics / Cray Research)
- [Genitor](#)
  - [Genitor FAQs](#) (Genitor)
  - [White Papers](#) (Genitor)
- [IBM VisualAge for C++](#)
  - [Download](#) (IBM)
  - [Library](#) (IBM)
  - [Support](#) (IBM)
  - [VisualAge Developer Domain](#) (ibm)
  - [VisualAge for C++ Library](#) (IBM)
  - [Getting Started](#) (IBM)
  - [VisualAge C++ Goes Incremental](#) (IBM)
- [Metrowerks CodeWarrior](#)
  - [CodeWarrior IDE Tour](#) (Metrowerks)
  - [CodeWarrior Professional](#) (Metrowerks)
- [SNiFF+](#) (TakeFive Software)
- [Sybase Power++](#) (Sybase)
- [Symantec C++](#) (Symantec)
- [Turbo C++](#) (Inprise / Borland)
- [Visual WorkShop C++ / Forte C++](#)
  - [Product Overview](#) (Sun Microsystems)
  - [Sun Visual WorkShop Evaluation Guide](#) (Sun Microsystems)
- [Watcom C/C++](#) (Sybase)
- **Compiladores / Interpreters**
  - [Compilers](#) (Open Directory)
  - <http://sources.redhat.com/cygwin/cygwin-api/cygwin-api.html> (Cygwin API Reference)
  - <http://sources.redhat.com/cygwin/cygwin-ug-net/cygwin-ug-net.html> (Cygwin User's Guide)
  - [Apogee C++](#) (Apogee Software)
  - [Comeau C++ Front End](#) (Comeau Computing)
  - [Edison C++ Front End](#) (Edison Design Group)
  - [GNU Compiler Collection](#)
    - [GNU G++ Documentation \(?\)](#)
    - [GNU G++ FAQs](#) (Joe Buck)
    - [GNU G++ MS-DOS Version](#) (DJ Delorie)
    - [GNU G++ Win32 Version](#) (Cygnum Solutions)
    - [RHIDE / GNU G++ IDE](#) (Robert Höhne)
  - [Intel C/C++ Compiler](#)
    - [The Intel C/C++ Compiler](#) (Intel)
  - [KAI C++](#)
    - [Kai C++ Documentation Set](#) (Kuck & Associates)
    - [Kai C++ FAQs](#) (Kuck & Associates)
    - [Kai C++ Free Trial Period](#) (Kuck & Associates)
  - [MetaWare High C++](#) (MetaWare)

- [Portland Group PGCC Workstation](#) (The Portland Group)
- [TenDRA](#)
  - [Rob's Unofficial TenDRA Page](#) (Rob Andrews)
  - [TenDRA](#) (OSSG)
- [Open source port of Sybase's Watcom C/C++](#) (Watcom)
- **Utilidades / Herramientas**
  - [Freeware/Shareware](#) (About.com)
  - [Static Source Code Analysis Tools](#) (About.com) (Lint)
  - [Ccdoc - Documentation Generator](#) (Joe Linoff)
  - [CC-RIDER - C/C++ Code Visualization](#) (Western Wares)
  - [ClassBuilder](#) (Jimmy Venema)
  - [ClassExplorer Pro](#) (toolsfactory)
  - [Code Crusader](#) (John Lindal)
  - [Code Navigator for C++](#) (Quintessoft Engineering)
  - [Create HTML Pages from C++ Header Files](#) (Darrell Schiebel)
  - [ctoohtml: a c/C++ to HTML filter](#) (Andrew H. Fagg)
  - [DDD - Data Display Debugger](#) (Andreas Zeller)
  - [Development Tools](#) (C/C++ Users Journal)
  - [IBM Classes for Unicode](#) (IBM) (ICU)
  - [Look!](#) (Objective Software Technology)
  - [V - Integrated Development Environment](#) (Object Central)
- **GUIs**
  - [Graphics Libraries](#) (About.com)
  - [OpenAmulet](#) (OpenIP)
  - [Qt](#) (Troll Tech)
  - [The GUI Toolkit, Framework Page](#) (Li-Cheng Tai)
  - [V - A Freeware Portable C++ GUI Framework for Windows and X](#) (Bruce E. Wampler)
- **Librerías**
  - [Available C++ Libraries FAQ](#) (Nikki Locke)
  - [C++ Class Libraries](#) (University of Darmstadt)
  - [C++ Class Library Review](#) (Paul Kent)
  - [C++ Class Libraries FAQ](#) (Yahoo)
  - [C++ Libraries](#) (boost.org)
  - [Class Libraries](#) (Marshall Cline)
  - [Class Libraries](#) (Open Directory)
  - [Cross-Platform GUI](#) (Miller Freeman)
  - [ACE](#) (Douglas C. Schmidt)
  - [Blitz++](#) (Todd Veldhuizen et al.)
  - [Carrick Encryption](#) (Azalea Software)
  - [C++ Data Object Library](#) (Code Farms)
  - [ChartFolio](#) (DPC Technology)
  - [CIDLib Class Library System](#) (Charmed Quark Software)
  - [COOOL](#) (Lydia Deng & Wences Gouveia)
  - [C++SIM](#) (M.C.Little)
  - [DEGUI: C++ Objects for Allegros GUI](#) (D.J.Eleveld)
  - [Diffpack](#) (Numerical Objects)
  - [Dinkum C++ Libraries](#) (Dinkumware)
  - [ET++ Application Framework Distribution](#) (GUP Linz)
  - [Generic++](#) (OOTec)

- [Geophile - an extensible spatial index](#) (Geophile)
- [GNU Nana: assertions and logging](#) (P.J.Maker)
- [Go++](#) (Northwoods Software)
- [Hush](#) (Anton Eliens)
- [ILOG Views](#) (ILOG)
- [ILOG Vision](#) (ILOG)
- [IMSL](#) (Visual Numerics)
- [International Classes for Unicode](#) (IBM)
- [JThreads/C++](#) (Object-Oriented Concepts)
- [JX - Graphical User Interfaces](#) (?)
- [LEDA](#) (MPI)
- [OptSolve++](#) (Tech-X)
- [MET++](#) (University of Zurich)
- [MetaKit](#) (Equi4 Software)
- [MorphCGI C++ class library](#) (MORPH.COM)
- [MWA GIF Component Library](#) (MWA Software)
- [MWA JPEG Component Library](#) (MWA Software)
- [ONC RPC/XDR Toolkit](#) (Distinct)
- [OSE](#) (Dumpleton Software Consulting)
- [POOMA](#) (POOMA Team)
- [ProHelp](#) (Igor Glukhov)
- [PTL Pattern Template Library](#) (Code Farms)
- [RCS - Real-Time Control Systems Library](#) (Will Shackelford.)
- [SFL - Standard Function Library](#) (iMatix)
- [Tech-X Class Libraries](#) (Tech-X)
- [The Database Access Library](#) (?)
- [The ROOT System](#) (Rene Brun, Fons Rademakers)
- [The Visualization Toolkit](#) (Will Schroeder et al.)
- [Undoable C++ Object Library](#) (Notation Software)
- [V](#) (Bruce E. Wampler)
- [VWCL - Virtual Windows Class Library](#) (The VWCL Alliance )
- [wxWindows](#) (Anthemion Software)
- [XML for C++](#) (IBM)
- [Yacc++ and the Language Objects Library](#) (Compiler Resources)
- [YACL](#) (M. A. Sridhar)
- [Zinc Application Framework](#) (Zinc Software)
- Intel Performance Library Suite (Intel)
  - [Intel Performance Library Suite](#) (Intel)
  - [Image Processing Library](#)
  - [JPEG Library](#)
  - [Math Kernel Library](#)
  - [Recognition Primitives Library](#)
  - [Signal Processing Library](#)
- ObjectSpace C++ <ToolKit> (ObjectSpace)
  - [ObjectSpace C++ <ToolKit>](#) (ObjectSpace)
  - [Communications<ToolKit>](#)
  - [Foundations<ToolKit>](#)
  - [Internet<ToolKit>](#)
  - [Standards<ToolKit>](#)
  - [Systems<ToolKit>](#)

- [Web<ToolKit>](#)
- [Rogue Wave Libraries](#) (Rogue Wave Software)
- [Rogue Wave Libraries](#) (Rogue Wave Software)
- [Analytics.h++](#)
- [DBTools.h++](#)
- [Money.h++](#)
- [Standard C++ Library](#)
- [Threads.h++](#)
- [Tools.h++](#)
- [Tools.h++ Professional](#)
- [Standard Template Library \(STL\)](#)
- [A Modest STL Tutorial](#) (Jak Kirman)
- [A Tiny STL Primer](#) (David Harvey)
- [Effective STL](#) (David Harvey)
- [STL Newbie Guide](#) (Mumit Khan)
- [STL Online Reference Home Page](#) (Kenny Zalewski)
- [STL Programmer's Guide](#) (Silicon Graphics & Hewlett-Packard)
- [Standard Template Library Programmer's Guide](#) (EarthWeb)
- [The SGI Standard Template Library](#) (Dr. Dobb's Journal)
- [The STL](#) (David R. Musser)
- [The STL Made Simple](#) (Bruce Eckel)
- [The STL Tutorial](#) (Johannes Weidl)
- Others
- [CodeWeb: Data Mining Software Development Experience](#) (Amir Michail)
- **Otros programas**
- [Dynace Object Oriented Extension To C](#) (Blake McBride )
- **Otros Recursos**
- [C++Builder 1.0 Jump Center](#) (The Delphi Super Page)
- [C++](#) (René Eng)
- [C/C++ Snippets](#) (niklas.olsson)
- [C++ Resources Collection](#) (Code Beach)
- [C++ Code Collection](#) (Sacred.dk)
- [DevX Product Guide](#) (DevX)
- [Favorite Source Code Links](#) (CUJ)
- [Free Source Code](#) (About.com)
- [G.L.A.D. Components for Borland Delphi and C++Builder developers](#) (Greg Lief)
- [Glg Toolkit](#) (Generic Logic)
- [OpenC++](#) (Shigeru Chiba)
- [Our Favorite Source Code Sites](#) (Miller Freeman)
- [Powersoft Power++ Code](#) (Bill Klein)
- [Powersoft Power++ Components](#) (Bill Klein)
- [Source code](#) (Robert Davies)