```
/*****************************************************************************
  PROGRAM: Mhello.c
  PURPOSE: "Pienin Windows-ohjelma".  Tulostaa näyttöön tekstin Hello World
  Editor:  Vesa Lappalainen  typistänyt malliohjelmista.
  Project: mhello.c, mhello.def
*****************************************************************************/
#include <windows.h>            /* Tarvitaan kaikissa Windows C-ohjelmissa  */

LONG CALLBACK _export MainWndProc(HWND hWnd, UINT message,
                                  WPARAM wParam, LPARAM lParam)
{
  PAINTSTRUCT ps;

  switch (message)
  {
    case WM_PAINT:            /* Viesti: Piirrä ikkuna uudelleen            */
      if ( BeginPaint(hWnd,&ps) )
        TextOut(ps.hdc, 10, 10, "Hello World!",12);
      EndPaint(hWnd,&ps);
      return 0;
    case WM_DESTROY:          /* Viesti: ikkuna hävitetään                  */
      PostQuitMessage(0);
      return 0;
    default:                  /* Antaa Windowsin käsitellä muut             */
      break;
  }
  return DefWindowProc(hWnd, message, wParam, lParam);
}

int PASCAL WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
                   LPSTR lpCmdLine, int nCmdShow)
{
  WNDCLASS  wc;   /* Ikkunaluokka                                           */
  HWND    hWnd;   /* Pääikkunan kahva                                       */
  MSG msg;        /* Viesti                                                 */
  (void)lpCmdLine; /* Hämäystä, jottei valitusta param. käytt.              */

  // if-lause ei ole tarpeen WIN32-sovelluksissa
  // (win95 ja myöhemmissä hPrevInstance on aina NULL),
  // koska jokaista sovellusta ajetaan omassa koodisegmentissään.
  // if-lause säilytetty yhteensopivuuden vuoksi. MN / 3.6.2004
  if (!hPrevInstance) {       // Onko muita esiintymiä käynnisssä?
    wc.style         = 0   ;           wc.lpfnWndProc    = MainWndProc;
    wc.cbClsExtra    = 0;              wc.cbWndExtra     = 0;
    wc.hInstance     = hInstance;
    wc.hIcon         = LoadIcon(NULL, IDI_APPLICATION);
    wc.hCursor       = LoadCursor(NULL, IDC_ARROW);
    wc.hbrBackground = GetStockObject(WHITE_BRUSH);
    wc.lpszMenuName  = NULL;           wc.lpszClassName  = "WHelloWClass";
    if (!RegisterClass(&wc)) return 1;
  }

  hWnd = CreateWindow("WHelloWClass","Windows Hello",WS_OVERLAPPEDWINDOW,
                      CW_USEDEFAULT,CW_USEDEFAULT,CW_USEDEFAULT,CW_USEDEFAULT,
                      NULL,NULL,hInstance,NULL);
  if ( !hWnd ) return 1;
  ShowWindow(hWnd, nCmdShow);     /* Näytetään ikkuna                       */
  UpdateWindow(hWnd);             /* Lähetetään WM_PAINT viesti             */

  while (GetMessage(&msg,NULL,0,0)) {
      TranslateMessage(&msg);     /* Tulkitaan virtuaaliset näp. koodit     */
      DispatchMessage(&msg);      /* Lähetetään viesti ikkunalle            */
  }
  return msg.wParam;           /* Palautetaan PostQuitMessage-funktion arvo */
}
```



```
/*************/
/* samplew.c  */
/*****************************************************************************
    PROGRAM: samplew.c
    PURPOSE: Runko yksinkertaisille Windows-ohjelmille.
               Kirjoitetaan tiedosto, jossa on funktiot
                 MyMove   (kun hiiri liikkunut)
                 MyDown   (kun hiiren näppäin alhaalla)
                 MyUp     (kun hiiren nappi päästetty ylös) ja
                 MyDraw   (vaikkapa tiedostoon myown.c)
               sekä merkkijono
                 WindowName
               ja tehdään projekti, jossa on tiedostot
                 samplew.c
                 myown.c
                 samplew.def
               ja käännetään!  Siinä se!

    Editor:  Vesa Lappalainen 15.8.1992
*****************************************************************************/
#include <windows.h>            /* Tarvitaan kaikissa Windows C-ohjelmissa  */

#define MAINWNAME "SampleWClass" /* Tunniste, jolla ikkunaluokka tunnetaan  */

extern char *WindowName;
extern void MyDraw(HWND hWnd, HDC hDC);
extern void MyDown(HWND hWnd, WORD wParam, LONG lParam);
extern void MyUp  (HWND hWnd, WORD wParam, LONG lParam);
extern void MyMove(HWND hWnd, WORD wParam, LONG lParam);

/*****************************************************************************/
LONG CALLBACK _export MainWndProc(HWND hWnd, UINT message,
                                  WPARAM wParam, LPARAM lParam)
{
    PAINTSTRUCT ps;

    switch (message)
    {
        case WM_PAINT:            /* Viesti: Piirrä ikkuna uudelleen        */
           if ( BeginPaint(hWnd,&ps) )
               MyDraw(hWnd,ps.hdc);
             EndPaint(hWnd,&ps);
           return (NULL);

        case WM_LBUTTONDOWN:      /* Hiiren vasen nappi alas:               */
           MyDown(hWnd,wParam,lParam);
           return (NULL);

        case WM_LBUTTONUP:        /* Hiiren vasen nappi ylös:               */
           MyUp(hWnd,wParam,lParam);
           return (NULL);

        case WM_MOUSEMOVE :       /* Hiirtä siirretty:                      */
           MyMove(hWnd,wParam,lParam);
           return (NULL);

        case WM_DESTROY:          /* Viesti: ikkuna hävitetään              */
            PostQuitMessage(0);
            return (NULL);

        default:                  /* Antaa Windowsin käsitellä muut         */
            break;
    }
    return (DefWindowProc(hWnd, message, wParam, lParam));
}
```

```
/*************************************************************************/
int PASCAL WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
                   LPSTR lpCmdLine, int nCmdShow)
{
    WNDCLASS  wc;   /* Ikkunaluokka                                      */
    HWND    hWnd;   /* Pääikkunan kahva                                  */
    MSG msg;        /* Viesti                                            */
    (void)lpCmdLine;/* Hämäystä, jottei valitusta param. käytt.          */

    if (!hPrevInstance) {       /* Onko muita esiintymiä käynnissä?      */
      wc.style         = NULL;           wc.lpfnWndProc    = MainWndProc;
      wc.cbClsExtra    = 0;              wc.cbWndExtra     = 0;
      wc.hInstance     = hInstance;
      wc.hIcon         = LoadIcon(NULL, IDI_APPLICATION);
      wc.hCursor       = LoadCursor(NULL, IDC_ARROW);
      wc.hbrBackground = GetStockObject(WHITE_BRUSH);
      wc.lpszMenuName  = NULL;           wc.lpszClassName  = MAINWNAME;
      if (!RegisterClass(&wc)) return (FALSE);
    }

    hWnd = CreateWindow(MAINWNAME,WindowName,WS_OVERLAPPEDWINDOW /* | CS_SAVEBITS */,
                        CW_USEDEFAULT,CW_USEDEFAULT,CW_USEDEFAULT,CW_USEDEFAULT,
                        NULL,NULL,hInstance,NULL);
    if (!hWnd) return (FALSE);
    ShowWindow(hWnd, nCmdShow);     /* Näytetään ikkuna                  */
    UpdateWindow(hWnd);             /* Lähetetään WM_PAINT viesti        */

    while (GetMessage(&msg,NULL,NULL,NULL)) {
        TranslateMessage(&msg);     /* Tulkitaan virtuaaliset näp. koodit*/
        DispatchMessage(&msg);      /* Lähetetään viesti ikkunalle       */
    }
    return (msg.wParam);      /* Palautetaan PostQuitMessage-funktion arvo */
}
```



```
/*************/
/* kolmio.c   */
/*************************************************************************/
/*
** Malliohjelma kauan kestävästä piirtämisestä.
** Piirretään rekursiivisesti joukko kolmioita.
**
** Vesa Lappalainen 16.8.1992 & 14.6.1994
**
** Kääntäminen:
**  Tee projektio kolmio.prj:
**     kolmio.c
**     simplet.def
**     ALI\timer.c
**     ALI\tabhand.c
**
** Tehtäviä 1)  Kokeile piirtämisen aikana valita ikkunan suurennus tms. tai
**              paina vaikkapa kahdesti (molemmat välillä irti) ALT-TAB.
**              Säädä ohjelman suoritusaikaa muuttelemalla vakiota
**              PIENIN_KOLMIO (pieni->kauan, iso-> nopea).
**              Mikä vika on ohjelmassa?
**
**              Käynnistä samanaikaisesti jokin toinen ohjelma ja säädä sen
**              ikkunan niin pieneksi, ettei se peitä kuin osaksi "Kolmiot"
**              ikkunaa.  Laita sitten ikkunoiden järjestys siten, että
**              "Kolmiot"-ikkuna on toiseksi ylin ja avaamasi pikku-ikkuna
**              on ylimpänä.  Siirtele pikkuikkunaa kolmiot ikkunan päällä!
**
**          2)  Ohjelmaikkunan otsikkoksi muuttuu aika, jonka ikkunan uudelleen
**              piirtäminen vei.  Kokeile peittää ikkunaa eri tavoin toisella
**              ikkunalla ja koeta keksiä syy, miksi uudelleen piirto vie
**              eri ajan peiton suuruudesta riippuen.
**
**          3)  Ohjelmaan liittyy myös "höpö-höpö"-piirtäminen.  Miten luulet
**              piirtämisen toimivan?  Käy välillä toisessa ikkunassa ja
**              palaa takaisin ohjelmaan.  Mitä käy piirroksille?
**
**          4)  Muuta "piirtäminen" sellaiseksi, että kun hiiren nappi
**              painetaan alas, aletaan piirtämään viivaa kunnes
**              nappi päästetään ylös.  Mitä tapahtuu kun nappi
**              päästetäänkin ylös "Kolmiot"-ikkunan ulkopuolella ja
**              sitten tuodaan hiiri takaisin ikkunan sisälle nappi
**              ylhäällä?
**
**          5)  Miten "Piirtäminen" pitäisi muuttaa, jotta kuva korjautuisi
**              toisessa ikkunassa käymisen jälkeen?
**
-------------------------------------------------------------------------*/
#include <windows.h>
#include <stdio.h>
#include <math.h>
#include "timer.h"
#include "tabhand.h"

/*************************************************************************/
TblClassSWindowMAIN("TriangleClass",0,"Kolmio",MsgTbl,0);
/*************************************************************************/

#define PIENIN_KOLMIO 0.2   /* Säädä tällä kauanko koko kuvan piirt. kest.  */

#define I(x,y) hdc,((int)(x)),((int)(y))

/*************************************************************************/
void kolmio(HDC hdc, double x, double y, double h)
{
  double s2 = h / (sqrt(3));
```

```
  MoveToEx(I(x,y),NULL);
  LineTo(I(x-s2,y-h));
  LineTo(I(x+s2,y-h));
  LineTo(I(x,y));

  if ( h < PIENIN_KOLMIO ) return;

  kolmio(hdc,x-s2,y,h/2);  /* Pienempi kolmio vasemmalle */
  kolmio(hdc,x+s2,y,h/2);  /* Pienempi kolmio oikealle   */
  kolmio(hdc,x,y-h,h/2);   /* Pienempi kolmio yläpuolelle*/
}


/*****************************************************************************/
static EVENT WM_paint(tMSGParam *msg) /* # MAKE_DC # */
{
  char s[100];

  SetWindowText(msg->hWnd,"Kolmio: Piirretään...");

  start_timer(1);

  kolmio(msg->hDC,300,400,200);

  sprintf(s,"Kolmio: %5.2lf s.",stop_timer(1));

  SetWindowText(msg->hWnd,s);

  return NULL;
}


/*****************************************************************************/
/*
** Seuraavissa aliohjelmissa on seuraavat parametrit:
**
**   hWnd     - ikkunan kahva
**   wParam   - SHIFT, CTRL ja ALT -näppäimen tile, voidaan testata esim.
**                  if ( wParam && MK_CONTROL ) ... hommat jos CTRL alhaalla
**   lParam   - x = LOWORD(lParam) ja y = HIWORD(lParam)
**              kätevämpi on kuitenkin selvittää hiren paikka:
**               POINTS pt = MAKEPOINTS(lParam)
**
------------------------------------------------------------------------------*/

static POINTS old = {0,0};
static pen_down  = 0;

/*****************************************************************************/
static EVENT WM_lbuttondown(tMSGParam *msg)
/* Aliohjelmaa kutsutaan kun hiiren nappa on painettu alas                   */
{
  old = MAKEPOINTS(msg->lParam);
  pen_down = 1;
  return 0;
}

/*****************************************************************************/
static EVENT WM_lbuttonup(tMSGParam *msg)
/* Aliohjelmaa kutsutaan kun hiiren nappi on päästetty ylös                  */
{
#pragma argsused
  pen_down = 0;
  return 0;
}
```



```
/*****************************************************************************/
static EVENT WM_mousemove(tMSGParam *msg) /* # MAKE_DC # */
/* Aliohjelmaa kutsutaan kun hiiri on liikkunut                              */
{
  POINTS pt = MAKEPOINTS(msg->lParam);

  if ( !pen_down ) return 0;

  MoveToEx(msg->hDC,old.x,old.y,NULL);
  LineTo(msg->hDC,pt.x,pt.y);
  return 0;
}

/*****************************************************************************/
static EVENT WM_create(tMSGParam *msg)
/* Aliohjelmaa kutsutaan kun ikkuna on luotu, muttei vielä näytössä.         */
{
  MoveWindow(msg->hWnd,10,10,600,500,FALSE);
  return 0;
}

/*****************************************************************************/
/* Viestien käsittelytaulukko                                                */
/*****************************************************************************/
tMSGEntry MsgTbl[] = {
  EV_HANDLE_WM_DESTROY,
  { WM_PAINT , DoC , DoC , WM_paint,  MAKE_DC  }, /*a*/
  { WM_LBUTTONDOWN , DoC , DoC , WM_lbuttondown }, /*a*/
  { WM_LBUTTONUP , DoC , DoC , WM_lbuttonup }, /*a*/
  { WM_MOUSEMOVE , DoC , DoC , WM_mousemove,  MAKE_DC  }, /*a*/
  { WM_CREATE , DoC , DoC , WM_create }, /*a*/
  { 0 }
};
/*****************************************************************************/
```

```
/*************************************************************************
    PROGRAM: Thello.c
    PURPOSE: Ohjelma tyypillisten kontrollien testaamiseen.
    Editor:  Vesa Lappalainen  typistänyt malliohjelmista.
    Project: thello.c, whello.def
*************************************************************************/
#include <windows.h>            /* Tarvitaan kaikissa Windows C-ohjelmissa */
#include <windowsx.h>

#define MY_BUTTON 175           /* Nappulan tunniste                       */
#define MY_EDIT   176           /* Tekstikentän (Edit) tunniste            */
#define MY_BAR    177           /* Oman likusäätimen tunniste              */
#define MY_STATIC 178           /* 1. vakiotekstin tunniste = liun arvo    */
#define MY_BARS   179           /*  apuvakio seuraaville                   */
#define MY_SVER   (MY_BARS+SB_VERT) /* 2. tekstikenän tunniste = pystyliun */
#define MY_SHOR   (MY_BARS+SB_HORZ) /* 3. tekstikentän tunniste = vaakaliun*/

#define MIN_BAR    0
#define MAX_BAR   50

static int value = 0;

LONG CALLBACK MainWndProc(HWND hWnd, UINT message,
                          WPARAM wParam, LPARAM lParam)
{
  PAINTSTRUCT ps;

  switch (message)
  {
    case WM_PAINT:            /* Viesti: Piirrä ikkuna uudelleen       */
      if ( BeginPaint(hWnd,&ps) )
        TextOut(ps.hdc, 10, 10, "Hello World!",12);
      EndPaint(hWnd,&ps);
      return NULL;

    case WM_COMMAND:
      switch ( GET_WM_COMMAND_ID(wParam,lParam) ) {
        case MY_BUTTON:     /* Painonappia painettu                  */
          SetDlgItemInt(hWnd,MY_STATIC,++value,TRUE);
          return NULL;
        case MY_EDIT:       /* Jos viesti teksti-ikkunalta           */
          switch ( GET_WM_COMMAND_CMD(wParam,lParam) ) {
            case EN_CHANGE: /* Aina kun sisältö muuttuu, lisät.lask  */
              SetDlgItemInt(hWnd,MY_STATIC,++value,TRUE);
              return NULL;
            case EN_KILLFOCUS: {/* Teksti-ikkunasta poistuttu        */
              char svalue[20];
              GetDlgItemText(hWnd,MY_EDIT,svalue,sizeof(svalue));
              SetWindowText(hWnd,svalue); /* Sama tulos pääotsikkoon */
              return NULL;
           }
          }
          return NULL;
        default: break;
      }
      break;

    case WM_VSCROLL: {        /* Johonkin pysytliukuun koskettu        */
      HWND shWnd = GET_WM_VSCROLL_HWND(wParam,lParam);
      int fnBar = SB_CTL;
      int id = GetDlgCtrlID(shWnd);
      if ( shWnd == NULL ) { shWnd = hWnd; id = SB_VERT; fnBar = SB_VERT; }
      switch ( GET_WM_VSCROLL_CODE(wParam,lParam) ) {
        case SB_BOTTOM       : value = MIN_BAR; break;
        case SB_TOP          : value = MAX_BAR; break;
        case SB_LINEDOWN     : value++;         break;
```



```
        case SB_PAGEDOWN     : value += 10;     break;
        case SB_LINEUP       : value--;         break;
        case SB_PAGEUP       : value -= 10;     break;
        case SB_THUMBPOSITION:
        case SB_THUMBTRACK   : value = GET_WM_VSCROLL_POS(wParam,lParam);
                                                break;
      }
      SetScrollPos(shWnd,fnBar,value,TRUE);
      SetDlgItemInt(hWnd,id == MY_BAR ? MY_STATIC : MY_SVER ,value, TRUE);
      return NULL;
    }

    case WM_DESTROY:          /* Viesti: ikkuna hävitetään             */
        PostQuitMessage(0);
        return NULL;
    default:                  /* Antaa Windowsin käsitellä muut        */
        break;
  }
  return DefWindowProc(hWnd, message, wParam, lParam);
}

int PASCAL WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
                   LPSTR lpCmdLine, int nCmdShow)
{
  HWND hWnd,hCtrl;/* Pääikkunan kahva ja apukahva kontrollien luontiin.  */
  WNDCLASS  wc;   /* Ikkunaluokka                                        */
  MSG msg;        /* Viesti                                              */
  (void)lpCmdLine; /* Hämäystä, jottei valitusta param. käytt.           */

  if (!hPrevInstance) {       /* Onko muita esiintymiä käynnisssä?       */
    wc.style         = NULL;           wc.lpfnWndProc    = MainWndProc;
    wc.cbClsExtra    = 0;              wc.cbWndExtra     = 0;
    wc.hInstance     = hInstance;
    wc.hIcon         = LoadIcon(NULL, IDI_APPLICATION);
    wc.hCursor       = LoadCursor(NULL, IDC_ARROW);
    wc.hbrBackground = GetStockObject(WHITE_BRUSH);
    wc.lpszMenuName  = NULL;           wc.lpszClassName  = "WHelloWClass";
    if ( !RegisterClass(&wc) ) return 1;
  }

  hWnd = CreateWindow("WHelloWClass","Windows Hello",
          WS_CAPTION | WS_BORDER | WS_VSCROLL | WS_HSCROLL | WS_SYSMENU |
          WS_THICKFRAME | WS_MINIMIZEBOX  |  WS_MAXIMIZEBOX,
          50,50,300,300,NULL,NULL,hInstance,NULL);
          if ( !hWnd ) return 1;

  ShowWindow(hWnd,nCmdShow);        /* Näytetään ikkuna                */
  hCtrl = CreateWindow("Button","Windows Hello",
                       WS_BORDER |WS_CHILD | WS_VISIBLE ,
                       10,30,100,30,hWnd,(HMENU)MY_BUTTON,hInstance,NULL);
                       if ( !hCtrl ) return 1;
  hCtrl = CreateWindow("Static","0",WS_CHILD | WS_VISIBLE | SS_RIGHT,
                       40,70,40,30,hWnd,(HMENU)MY_STATIC,hInstance,NULL);
                       if ( !hCtrl ) return 1;
  hCtrl = CreateWindow("Static","0",WS_CHILD | WS_VISIBLE | SS_RIGHT ,
                       80,70,40,30,hWnd,(HMENU)MY_SHOR,hInstance,NULL);
                       if ( !hCtrl ) return 1;
  hCtrl = CreateWindow("Static","0",WS_CHILD | WS_VISIBLE | SS_RIGHT,
                       120,70,40,30,hWnd,(HMENU)MY_SVER,hInstance,NULL);
                       if ( !hCtrl ) return 1;
  hCtrl = CreateWindow("Edit","Windows Hello",
                       WS_DLGFRAME | WS_CHILD | WS_VISIBLE,
                       10,110,100,40,hWnd,(HMENU)MY_EDIT,hInstance,NULL);
                       if ( !hCtrl ) return 1;
  hCtrl = CreateWindow("Scrollbar","Windows Hello",
                       WS_CHILD | WS_VISIBLE | SBS_VERT,
                       200,10,30,200,hWnd,(HMENU)MY_BAR,hInstance,NULL);
```

```
                    if ( !hCtrl ) return 1;

  SetScrollRange(hCtrl,SB_CTL,MIN_BAR,MAX_BAR,TRUE);
  SetScrollRange(hWnd,SB_VERT,MIN_BAR,2*MAX_BAR,TRUE);
  SetScrollRange(hWnd,SB_HORZ,MIN_BAR,3*MAX_BAR,TRUE);

  while ( GetMessage(&msg,NULL,NULL,NULL) ) {
      TranslateMessage(&msg);     /* Tulkitaan virtuaaliset näp. koodit  */
      DispatchMessage(&msg);      /* Lähetetään viesti ikkunalle         */
  }
  return msg.wParam;        /* Palautetaan PostQuitMessage-funktion arvo */
}
```



```
/*************/
/* laskurir.c */
/*****************************************************************************
**  PROGRAM: laskurir.c
**  Windows: Win 3.1 & WIN32
**  PURPOSE: Autolaskuri tehtynä mahd. pitkälle Windowsin resursseilla.
**  Editor:  Vesa Lappalainen
**    Projektiin tarvitaan
**      laskurir.c    - tämä tiedosto
**      laskurir.def  - moduulin määrittely
**      laskurir.rc   - resurssit
**
** Tehtäviä: 1)  Lisää polkupyörien laskeminen (käytä .rc tiedoston
**               korjailemiseen Resource WorkShopia)
**
**           2)  Lisää +/- toiminto näppäinten avulla
**
**           3)  Muuta .rc tiedostoa siten, että myös DEL-näppäin
**               tyhjentää näytön.
**
*****************************************************************************/
#include <windows.h>
#include <windowsx.h>           /* Tarvitaan 32-bit yhteensopivuudeksi     */
#include "laskurir.h"

static int laskurit[LASKUREITA] = {0,0};

BOOL CALLBACK _export MainWndProc(HWND hWnd, UINT message,
                                  WPARAM wParam, LPARAM lParam)
{
#pragma argsused
  switch (message) {
    case WM_COMMAND: {
      int id = GET_WM_COMMAND_ID(wParam,lParam); /* Jotta WIN32 toimisi */
      switch ( id ) {
        case HA:              /* Painonappia painettu                      */
        case KA:
          SetDlgItemInt(hWnd,id-HA+HAL,++laskurit[id-HA],TRUE);
          return TRUE;
        case NOLLAA: {
          int i;
          for (i=0; i<LASKUREITA; i++)
            SetDlgItemInt(hWnd,i+HAL,laskurit[i] = 0,TRUE);
          return TRUE;
         }  /* NOLLAA */
        case IDCANCEL:
        case EXIT:
          PostQuitMessage(0);
          return TRUE;
      } /* switch (id)        */
    }   /* WM_COMMAND         */
  }     /* switch ( message ) */
  return FALSE;
}

int PASCAL WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance,
                   LPSTR lpCmdLine, int nCmdShow)
{
#pragma argsused
  MSG     msg;
  FARPROC lProc  = MakeProcInstance((FARPROC)MainWndProc,hInstance);
  HWND    hWnd   = CreateDialog(hInstance,"LASKURI",NULL,(DLGPROC)lProc);
  HACCEL  hAccel = LoadAccelerators(hInstance,"PIKA");

  if ( !hWnd ) goto lopetus;
  while (GetMessage(&msg,NULL,NULL,NULL)) {
```

```
    if ( TranslateAccelerator(hWnd,hAccel,&msg) ) continue;
    if ( IsDialogMessage(hWnd,&msg)  )            continue;
    TranslateMessage(&msg);     /* Tulkitaan virtuaaliset näp. koodit      */
    DispatchMessage(&msg);      /* Lähetetään viesti ikkunalle             */
  }
lopetus:
  (void)FreeProcInstance(lProc); /* void, jottei 32-bit kääntäjä valita */
  return 0;
}
```



```
#include "laskurir.h"
LASKURI DIALOG 17, 25, 180, 88
STYLE WS_OVERLAPPED | WS_VISIBLE | WS_CAPTION | WS_SYSMENU | WS_MINIMIZEBOX
CAPTION "LASKURIR"
{
 CONTROL "&Henkilöautoja", HA, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP, 33, 10, 50, 20
 CONTROL "&Kuorma-autoja", KA, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP, 100, 10, 50, 20
 CONTROL "0 ", HAL, "STATIC", SS_RIGHT | WS_CHILD | WS_VISIBLE | WS_BORDER | WS_GROUP, 31, 40, 50, 10
 CONTROL "0 ", KAL, "STATIC", SS_RIGHT | WS_CHILD | WS_VISIBLE | WS_BORDER | WS_GROUP, 100, 40, 50, 10
 CONTROL "&Nollaa", NOLLAA, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP, 50, 60, 80, 20
 CONTROL "e&Xit", EXIT, "BUTTON", BS_PUSHBUTTON | WS_CHILD | WS_VISIBLE | WS_TABSTOP, 0, 0, 24, 14
}

PIKA ACCELERATORS
BEGIN
        "K", KA
        "k", KA
        "H", HA
        "h", HA
        "N", NOLLAA
        "n", NOLLAA
        "X", EXIT
        "x", EXIT
END
```

```
/****************/
/* kolmio.cpp   */
/****************/
// Ohjelma, joka aukaisee yhden ikkunan ja piirtää siihen rekursiivisen kolmion.
// Esimerkki kuvan päiväittämisestä ja piirtämisen kestosta   /vl-96
// Projektiin kolmio.cpp ja ALI\timer.c.                      MFC 4.1
// Tehtäviä:
//
#include <afxwin.h>
#include <math.h>
#include <stdio.h>
#include <math.h>
#include "timer.h"

#define PIENIN_KOLMIO 0.2   /* Säädä tällä kauanko koko kuvan piirt. kest.  */

#define I(x,y) hdc,((int)(x)),((int)(y))

/*************************************************************************/
void kolmio(HDC hdc, double x, double y, double h)
{
  double s2 = h / (sqrt(3));

  MoveToEx(I(x,y),NULL);
  LineTo(I(x-s2,y-h));
  LineTo(I(x+s2,y-h));
  LineTo(I(x,y));

  if ( h < PIENIN_KOLMIO ) return;

  kolmio(hdc,x-s2,y,h/2);  /* Pienempi kolmio vasemmalle */
  kolmio(hdc,x+s2,y,h/2);  /* Pienempi kolmio oikealle   */
  kolmio(hdc,x,y-h,h/2);   /* Pienempi kolmio yläpuolelle*/
}

//-------------------------------------------------------------------------
class CMainWindow : public CFrameWnd {
  CDC *pHoldDC;    // Siirron aikana käytössä oleva laiteyhteys
  CPoint old;
public:
  CMainWindow() {
    Create(NULL,"Kolmio");
    pHoldDC = NULL;
  }
  afx_msg void OnPaint() {
    char s[100];
    CPaintDC dc(this);
    SetWindowText("Piirretään...");
    start_timer(1);
    kolmio(dc,300,400,200);
    sprintf(s,"Kolmio: %5.2lf s.",stop_timer(1));
    SetWindowText(s);
  }
  afx_msg void OnLButtonDown(UINT modKeys, CPoint point) {
    if ( pHoldDC != NULL ) return;
    SetCapture();    // Direct all subsequent mouse input to this window
    pHoldDC = new CClientDC(this);
    old = point;
  }
  afx_msg void OnLButtonUp(UINT modKeys, CPoint point) {
    if ( pHoldDC == NULL ) return;
    ReleaseCapture();
    delete pHoldDC;
    pHoldDC = NULL;
  }
  afx_msg void OnMouseMove(UINT modKeys, CPoint point) {
```



```
    if ( pHoldDC == NULL ) return;
    pHoldDC->MoveTo(old);
    pHoldDC->LineTo(point);
  }
  DECLARE_MESSAGE_MAP()
};

BEGIN_MESSAGE_MAP( CMainWindow, CFrameWnd )
  ON_WM_PAINT()
  ON_WM_LBUTTONDOWN()
  ON_WM_LBUTTONUP()
  ON_WM_MOUSEMOVE()
END_MESSAGE_MAP()

//-------------------------------------------------------------------------
class CKolmioApp : public CWinApp {
public:
  virtual BOOL InitInstance() {
    m_pMainWnd = new CMainWindow();
          m_pMainWnd->ShowWindow(m_nCmdShow);
          m_pMainWnd->UpdateWindow();
          return TRUE;
  }
};

//-------------------------------------------------------------------------
CKolmioApp KolmioApp;  // constructor initializes and runs the app
```

```
/****************/
/* laskuri.cpp  */
/****************/
// Esimerkki Autolaskurista. Projektiin laskuri.cpp ja laskuri.rc

#include <afxwin.h>
#include "laskuri.rh"

//------------------------------------------------------------------------------
class TLaskuriDialog : public CDialog {
public:
  TLaskuriDialog(LPCSTR name,CWnd *parent=NULL) : CDialog(name,parent) {}
  void BNHAClicked()         { SetDlgItemInt(HAL,GetDlgItemInt(HAL)+1);        }
  void BNKAClicked()         { SetDlgItemInt(KAL,GetDlgItemInt(KAL)+1);        }
  void BNNollaaClicked()     { SetDlgItemInt(HAL,0); SetDlgItemInt(KAL,0);     }
  void BNExitClicked()       { EndDialog(0);                                   }
  DECLARE_MESSAGE_MAP()
};

BEGIN_MESSAGE_MAP(TLaskuriDialog, CDialog)
  ON_COMMAND(HA, BNHAClicked)
  ON_COMMAND(KA, BNKAClicked)
  ON_COMMAND(NOLLAA,BNNollaaClicked)
  ON_COMMAND(EXIT,BNExitClicked)
END_MESSAGE_MAP()


class TLaskuriApp : public CWinApp {
public:
  virtual BOOL InitInstance() {
    TLaskuriDialog dlg("LASKURI"); // Tyylinä WS_OVERLAPPED
    m_pMainWnd = &dlg;
    dlg.DoModal();
          return FALSE;  // Lopetetaan samalla koko ohjelma
  }
};

TLaskuriApp LaskuriApp;  //  constructor initializes and runs the app
```



```
/*************/
/* laskurir.h */
#define LASKUREITA 2

#define HA      1000
#define KA      HA+1
#define HAL     1010
#define KAL     HAL+1
#define EXIT    1027
#define NOLLAA  1028
```

```
//Microsoft Developer Studio generated resource script.
//
#include "resource.h"

#define APSTUDIO_READONLY_SYMBOLS
////////////////////////////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 2 resource.
//
#include "afxres.h"

////////////////////////////////////////////////////////////////////////////
#undef APSTUDIO_READONLY_SYMBOLS

////////////////////////////////////////////////////////////////////////////
// Finnish resources

#if !defined(AFX_RESOURCE_DLL) || defined(AFX_TARG_FIN)
#ifdef _WIN32
LANGUAGE LANG_FINNISH, SUBLANG_DEFAULT
#pragma code_page(1252)
#endif //_WIN32

////////////////////////////////////////////////////////////////////////////
//
// Dialog
//

LASKURI DIALOG DISCARDABLE  17, 25, 180, 88
STYLE WS_MINIMIZEBOX | WS_MAXIMIZEBOX | WS_VISIBLE | WS_CAPTION | WS_SYSMENU |
    WS_THICKFRAME
CAPTION "LASKURIR"
FONT 8, "MS Sans Serif"
BEGIN
    PUSHBUTTON      "&Henkilöautoja",HA,33,10,50,20
    PUSHBUTTON      "&Kuorma-autoja",KA,100,10,50,20
    RTEXT           "0 ",HAL,31,40,50,10,WS_BORDER
    RTEXT           "0 ",KAL,100,40,50,10,WS_BORDER
    PUSHBUTTON      "&Nollaa",NOLLAA,50,60,80,20
    PUSHBUTTON      "e&Xit",EXIT,0,0,24,14
END


////////////////////////////////////////////////////////////////////////////
//
// Accelerator
//

PIKA ACCELERATORS MOVEABLE PURE
BEGIN
    "K",            KA,                     ASCII
    "k",            KA,                     ASCII
    "H",            HA,                     ASCII
    "h",            HA,                     ASCII
    "N",            NOLLAA,                 ASCII
    "n",            NOLLAA,                 ASCII
    "X",            EXIT,                   ASCII
    "x",            EXIT,                   ASCII
END


#ifdef APSTUDIO_INVOKED
////////////////////////////////////////////////////////////////////////////
//
// TEXTINCLUDE
//
```



```
1 TEXTINCLUDE DISCARDABLE
BEGIN
    "resource.h\0"
END

2 TEXTINCLUDE DISCARDABLE
BEGIN
    "#include ""afxres.h""\r\n"
    "\0"
END

3 TEXTINCLUDE DISCARDABLE
BEGIN
    "\r\n"
    "\0"
END

#endif    // APSTUDIO_INVOKED

#endif    // Finnish resources
////////////////////////////////////////////////////////////////////////////


#ifndef APSTUDIO_INVOKED
////////////////////////////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 3 resource.
//


////////////////////////////////////////////////////////////////////////////
#endif    // not APSTUDIO_INVOKED
```