ЗАВДАННЯ

Варіант №9

Функція

X=10\*(cos(t)-t\*sin(t))+5\*$cos^{3}$(t)-50\*cos(t/5)

Y=10\*(cos(t)+t\*sin(t))+5\*$sin^{3}$(t)-50\*sin(t/5)

Інтервал T=[1;18]

Крок h=0.1

Блок Схема

Початок

Задаємо початкові значення : xmax=0 , xmin =0, ymax=0 , ymin=0 , t=1

t=1 while (t$<$=18) t=t+0.1

Визначення максимальних і мінімальних значень функції

X=10\*(cos(t)-t\*sin(t))+5\*$cos^{3}$(t)-50\*cos(t/5)

Y=10\*(cos(t)+t\*sin(t))+5\*$sin^{3}$(t)-50\*sin(t/5)

Визначення центру , обчислення коефіцієнтів перетворення :

xmd=rect.right-rect.left-150;

ymd=rect.bottom-rect.top-150;

xmmin=rect.left;

xmmax=rect.right;

ymmin=rect.top;

ymmax=rect.bottom;

xc=(rect.right-rect.left)/2;

yc=(rect.bottom-rect.top)/2;

int xd=xmax-xmin;

int yd=ymax-ymin;

kx=xmd/xd;

ky=ymd/yd;

Будуємо осі координат :

 dc.MoveTo(xc,10);

 dc.LineTo(xc,yc\*2-10);

 dc.MoveTo(10,yc);

 dc.LineTo(xc\*2-10,yc);

 dc.MoveTo(xc-5,15);

 dc.LineTo(xc,10);

 dc.LineTo(xc+5,15);

 dc.MoveTo(xc\*2-15,yc-5);

 dc.LineTo(xc\*2-10,yc);

 dc.LineTo(xc\*2-15,yc+5);

Задаємо проміжок розмітки осі Х

xs=(-1)\*m; xs=xs+20

int xx=xc+(xs\*kx);

if (xs==m)

Проставляємо позначку осі Х : CString xchar("x"); dc.TextOutW (xx,yc-25,xchar);

Проставляємо розмір осі Х:

swprintf\_s(str,50,L"%1.1f",xs);

dc.TextOut(xx-20,yc+10,str);

dc.MoveTo(xx,yc-2);

dc.LineTo(xx,yc+2);

Задаємо проміжок розмітки осі Y

(xs<=m); ys=ys+20

int yy=yc-(ys\*ky);

if (ys==n)

Проставляємо позначку осі Y : CString ychar("y"); dc.TextOutW (xc-15,yy,ychar);

Проставляємо розмір осі Y:

dc.TextOutW (xc-15,yy,ychar);

swprintf\_s(str,50,L"%5.1f",ys);

dc.TextOut(xc+10,yy-7,str);

dc.MoveTo(xc-2,yy);

 dc.LineTo(xc+2,yy);

створюємо перо для побудови графіка

HPEN hPen = CreatePen(PS\_SOLID, 2, RGB(0, 255, 255));

HGDIOBJ hOldPen = SelectObject(dc, hPen);

Будуємо графік : t=1;

while (t <= 18)

Проводимо розрахунок для побудови : xp=10\*(cos(t)-t\*sin(t))+5\*cos(t)\*cos(t)\*cos(t)-50\*cos(t/5);

yp=10\*(sin(t)+t\*cos(t))+5\*sin(t)\*sin(t)\*sin(t)-50\*sin(t/5);

x=xc+(xp\*kx);

y=yc-(yp\*ky);

if(t==1)

Переносимо поточну вершину в першу точку графіку : dc.MoveTo(x,y);

dc.LineTo(x,y);

Кінець

Список ідентифікаторів констант, змінних, процедур і функцій, використаних в програмі, та їх пояснення

rect – структура, в якій записані розміри екрана.

GetClientRect(&rect) – команда для запису розмірів екрана в структуру rect.

str[192] – массив символів.

kx,ky – коефініент перетворення по осях x та y відповідно.

x0,y0 – мінімальне значення графіка в машинних координатах по осях x та y відповідно.

x – змінна, з інтервалом зміни від x1 до x2 з кроком h.

y – значення функції.

lx,ly – довжина осі по x та y відповідно.

sx,sy – ціле значення від розбиття осі на 10 частин по x та y відповідно.

tx,ty – змінна, значення якої відповідає довжині додатної осі по x та від’ємної по y відповідно.

xc,yc – центр графіка в точці x=0 по осях x та y відповідно.

xx,yy – змінна, яка відповідає за розбиття графіка на 10 частин по осях x та y відповідно.

t – тимчасова змінна.

xmax,xmin,ymax,ymin – мінімальне та максимальне значення функції по осях x та y відповідно.

txp,txm,typ,tym – змінні, які визначають положення тексту розмітки осей у додатному та від’мному напрямках x та y відповідно.

rect.bottom,rect.right – нижня та права сторона вікна, відповідає за розміри клієнтського вікна.

rect.top,rect.left – верхня та ліва сторона вікна, рівна 0.

Остаточно відлагоджений текст програми відповідно до отриманого завдання

// ChildView.cpp : implementation of the CChildView class

//

#include "stdafx.h"

#include "Graf\_lab\_2.h"

#include "ChildView.h"

#include <math.h>

#ifdef \_DEBUG

#define new DEBUG\_NEW

#endif

// CChildView

CChildView::CChildView()

{

}

CChildView::~CChildView()

{

}

BEGIN\_MESSAGE\_MAP(CChildView, CWnd)

 ON\_WM\_PAINT()

END\_MESSAGE\_MAP()

// CChildView message handlers

BOOL CChildView::PreCreateWindow(CREATESTRUCT& cs)

{

 if (!CWnd::PreCreateWindow(cs))

 return FALSE;

 cs.dwExStyle |= WS\_EX\_CLIENTEDGE;

 cs.style &= ~WS\_BORDER;

 cs.lpszClass = AfxRegisterWndClass(CS\_HREDRAW|CS\_VREDRAW|CS\_DBLCLKS,

 ::LoadCursor(NULL, IDC\_ARROW), reinterpret\_cast<HBRUSH>(COLOR\_WINDOW+1), NULL);

 return TRUE;

}

void CChildView::OnPaint()

{

 CPaintDC dc(this); // device context for painting

 // TODO: Add your message handler code here

 // Do not call CWnd::OnPaint() for painting messages

 double y,x,xp,yp,kx,ky,h=0.1,t=1;

 wchar\_t str[192]={0};

 double xs,ys,n,m;

 double xmax=0,xmin=0,ymax=0,ymin=0,xmd,ymd,xmmin,xmmax,ymmin,ymmax,xc,yc;

 //візуальні розширення xmax,xmin,ymax,ymin

 RECT rect;

 GetClientRect(&rect);

 while (t <= 18)

 {

 x=10\*(cos(t)-t\*sin(t))+5\*cos(t)\*cos(t)\*cos(t)-50\*cos(t/5);

 if (x>xmax)

 {

 xmax=x;

 }

 else if (t==1 || x<xmin )

 {

 xmin=x;

 }

 y=10\*(sin(t)+t\*cos(t))+5\*sin(t)\*sin(t)\*sin(t)-50\*sin(t/5);

 if (y>ymax)

 {

 ymax=y;

 }

 else if (t==1 || y<ymin )

 {

 ymin=y;

 }

 t=t+0.1;

 }

 //znahodzhennya koeficientiv i centra grafika

 xmd=rect.right-rect.left-150;

 ymd=rect.bottom-rect.top-150;

 xmmin=rect.left;

 xmmax=rect.right;

 ymmin=rect.top;

 ymmax=rect.bottom;

 xc=(rect.right-rect.left)/2;

 yc=(rect.bottom-rect.top)/2;

 int xd=xmax-xmin;

 int yd=ymax-ymin;

 kx=xmd/xd;

 ky=ymd/yd;

 //будуємо еліпс

 dc.MoveTo(xc,10);

 dc.LineTo(xc,yc\*2-10);

 dc.MoveTo(10,yc);

 dc.LineTo(xc\*2-10,yc);

 dc.MoveTo(xc-5,15);

 dc.LineTo(xc,10);

 dc.LineTo(xc+5,15);

 dc.MoveTo(xc\*2-15,yc-5);

 dc.LineTo(xc\*2-10,yc);

 dc.LineTo(xc\*2-15,yc+5);

 //позначаємо осі

 n=yc\*2-20;

 m=xc\*2-20;

 n=n/ky;

 m=m/kx;

 n=n/2;

 m=m/2;

 n=floor(n/10)\*10;

 m=floor(m/10)\*10;

 xs=(-1)\*m;

 ys=(-1)\*n;

 do

 {

 int xx=xc+(xs\*kx);

 if (xs==m)

 {

 CString xchar("x");

 dc.TextOutW (xx,yc-25,xchar);

 swprintf\_s (str,50,L"%1.1f",xs);

 dc.TextOut(xx-20,yc+10,str);

 dc.MoveTo(xx,yc-2);

 dc.LineTo(xx,yc+2);

 }

 else

 {

 swprintf\_s (str,50,L"%1.1f",xs);

 dc.TextOut(xx-20,yc+10,str);

 dc.MoveTo(xx,yc-2);

 dc.LineTo(xx,yc+2);

 }

 //по х міняє числа

 xs=xs+20;

 }

 while (xs<=m);

 do

 {

 int yy=yc-(ys\*ky);

 if (ys==n)

 {

 CString ychar("y");

 dc.TextOutW (xc-15,yy,ychar);

 swprintf\_s (str,50,L"%5.1f",ys);

 dc.TextOut(xc+10,yy-7,str);

 dc.MoveTo(xc-2,yy);

 dc.LineTo(xc+2,yy);

 }

 else

 {

 swprintf\_s (str,50,L"%5.1f",ys);

 dc.TextOut(xc+10,yy-7,str);

 dc.MoveTo(xc-2,yy);

 dc.LineTo(xc+2,yy);

 }

 //міняє у відстані між точками

 ys=ys+20;

 }

 while (ys<=n);

 //створюємо перо для побудови графіка

 HPEN hPen = CreatePen(PS\_SOLID, 2, RGB(0, 255, 255));

 HGDIOBJ hOldPen = SelectObject(dc, hPen);

 t=1;

 while (t <= 18)

 {

 xp=10\*(cos(t)-t\*sin(t))+5\*cos(t)\*cos(t)\*cos(t)-50\*cos(t/5);

 yp=10\*(sin(t)+t\*cos(t))+5\*sin(t)\*sin(t)\*sin(t)-50\*sin(t/5);

 x=xc+(xp\*kx);

 y=yc-(yp\*ky);

 if(t==1)

 {

 dc.MoveTo(x,y);

 }

 dc.LineTo(x,y);

 t=t+h;

 }

}

Результати виконання програми