***Зміст***

Вступ………………………………………………………………………….......3

1. Аналіз проблеми та постановки задачі……..……………………………….4

2.Синтез зрівноважених блок-схем …..……………………………………...... 6

3.Синтез частково зрівноважених блок-схем…. ..…………………………...14

Висновок………………………………………………………………………..17

Список використаної літератури…………………………………………….. 18

Додаток 1………………………………………………………………………19

**Вступ**

Комбінаторні моделі та системи широко застосовуються в технічній кібернетиці, інформаційно – вимірювальній техніці, радіотехніці, зв’язку, електротехніці, машинобудуванні, а комбінаторні методи використовуються в теорії кодування, математичній логіці, програмуванні, теорії планування експерименту, технічній та статистичній фізиці, економіці, кристалографії, біології. Тому актуальними є дослідження властивостей існуючих і пошук нових комбінаторних моделей, способів їхньої побудови, класифікація, визначення умов існування, виявлення взаємних зв’язків та інтерпретацій.

Для синтезу комбінаторних систем застосовують здебільшого класичні методи комбінаторного аналізу, основані на положеннях комбінаторного аналізу з використанням табличних, теоретико – числових та матричних інтерпретацій, методи апарату скінченних груп перестановок, теорії чисел, схеми відношень, а також методи, пов’язані з використанням апарату теорії полів Галуа [1].

Методи побудови комбінаторних моделей із застосуванням впорядкованих сукупностей елементів – чисел, векторів чи будь – яких інших математичних величин або функцій, зв’язаних між собою деякою операцією, яка здійснюється над відповідними елементами даної сукупності, приводить до загального поняття «в’язанки» як математичного об’єкта, що відіграє роль базової структури у побудові оптимальних комбінаторних систем [2]. Математичні конструкції на впорядкованих елементах, організовані у вигляді в’язанок, виявилися зручним інструментом для синтезу та дослідження комбінаторних конфігурацій. Відношення інцидентності, що існують у більшості класичних комбінаторних конфігурацій, не завжди піддаються відносно простому математичному опису, тоді як у в’язанках ці відношення часто постають у вигляді звичайних числових послідовностей. Тому послідовності – в’язанки дають змогу спрощувати синтез комбінаторних конфігурацій, а відтак побудову комбінаторних систем і розширити теоретичні дослідження в області комбінаторики.

**Аналіз проблеми та постановки задачі**

Найпростішою структурною організацією елементів, що утворює математичний об’єкт під назвою «в’язанка», є одновимірна упорядкована послідовність чисел

З погляду постановки задачі синтезу та дослідження оптимальних комбінаторних моделей особливий інтерес викликають в’язанки з кільцевою структурою, елементами яких є цілі додатні числа, а суми на кільцевій послідовності вичерпують значення чисел натурального ряду від 1 до суми усіх чисел цієї послідовності. Така числова конструкція називається «ідеальною кільцевою в’язанкою», або скорочено ІКВ.

Наприклад, якщо брати упорядкований ряд (1,4,2) як кільцеву послідовність чисел, то легко перевірити, що всі кільцеві суми цієї послідовності вичерпують натуральний ряд чисел від 1 до :

1=1, 2=2, 3=2+1, 4=4, 5=1+4,6=4+2,7=1+4+2.

За допомогою в’язанок можна будувати блок – схеми. Під блок – схемою розуміють розміщення елементів множини v в підмножинах які називаються блоками, з однаковою кількістю елементів у кожному блоці причому елемент різних блоків, а кожна , повторюється в λ

До основного виду блок – схем належать зрівноважені неповні блок – схеми (*balanced incomplete block design*), або ВІВ схеми [1]. ВІВ – схеми утворені на множині у кожному блоці, причому елементи у блоках розміщуються так:

1. всі елементи одного блоку різні;
2. кожен елемент розміщується точно в
3. ,

Між параметрами

(1)

Числову в’язанку можна розглядати як структурований код, за допомогою якого зручно синтезувати відповідну комбінаторну систему. Методи побудови ВІВ – схем за допомогою одновимірних числових в’язанок розглядаються в [1,2].

Перехід від одновимірних до багатовимірних числових конструкцій є логічним продовженням дослідження організацій комбінаторних систем. У багатовимірних числових конструкціях – в’язанках відповідні ланцюжки операцій здійснюють над векторами багатовимірного простору. При перенесенні дій на послідовності векторів аналогами одновимірних послідовностей постають багатовимірні числові послідовності.

Багатовимірні в’язанки ще мало вивчені. На практиці особливий інтерес становлять методи побудови як самих двовимірних в’язанок, так і похідних конфігурацій за допомогою двовимірних в’язанок. Один із методів побудови двовимірних в’язанок ґрунтується на використанні одновимірних в’язанок[2].

**Синтез зрівноважених блок-схем**

Нехай задана одновимірна ідеальна кільцева в’язанка з параметрами Алгоритм побудови одновимірної ІКВ з такими самими параметрами на прямокутнику містить виконання таких операцій:

* на послідовності знайти елементи першого рядка таблиці кільцевих сум одновимірної ІКВ:

(2)

* на множині елементів знайти пари чисел

(3)

* упорядкувати пари чисел у зростаючому порядку числових значень їх елементів, починаючи з , а потім з , причому (0,0) записати в кінці упорядкованої послідовності; утворена послідовність з відповідно перейменованими індексами становить перший рядок таблиці кільцевих вектор – сум на n – послідовності елементів шуканої ІКВ,
* визначити елементи двовимірної ІКВ за формулами:

(4)

Наприклад, одновимірна (21,5,1) – ІКВ (2,5,1,3,10) перетворюється у двовимірну на прямокутній матриці після здійснення передбачених розглянутим алгоритмом операцій;

* на послідовності знаходять елементи першого рядка таблиці кільцевих сум одновимірної ІКВ – {2,7,8,11,21};
* на множині елементів записаного рядка за допомогою співвідношень (3) знаходять послідовність ((2,2),(1,0),(2,1),(2,4),(0,0));
* після упорядкування отримують перший рядок таблиці кільцевих сум двовимірної ІКВ:

((1,0),(2,1),(2,2),(2,4),(0,0)); (5)

* за допомогою залежностей (4) визначають елементи двовимірної ІКВ:

((1,0),(1,1),(0,1),(0,2),(1,3)); (6)

Побудована за цією послідовністю таблиця кільцевих вектор – сум має такий вигляд:

1,0 2,1 2,2 2,4 0,0

0,0 1,1 1,2 1,4 2,0

2,6 0,0 0,1 0,3 1,6

2,5 0,6 0,0 0,2 1,5

2,3 0,4 0,5 0,0 1,3

Тут кожен із упорядкованих 2 – наборів елементів, крім (0,0), трапляється точкою по одному разу, де перший елемент набуває значення числового ряду {0,1,2}, другий - {0,1,2,3,4,5,6}. Отже, знайдена послідовність утворює двовимірну ІКВ п’ятого (n=5) порядку на прямокутнику .

Синтез ВІВ – схем за допомогою двовимірної в’язанки здійснюють у два етапи.

На першому етапі будується ВІВ – схема за алгоритмом, який викладено в [1]. Для цього двовимірним елементам ІКВ ставиться у відповідність послідовностей:

;

;

… … … … …

;

… … … … …

;

елементи яких визначаються за формулами:

(7)

(8)

Де параметри двовимірної ІКВ.

Наприклад, з двовимірної ІКВ ((1,0),(1,1),(0,1),(0,2),(1,3)) за допомогою формул (7),(8) легко отримати таку конфігурацію: (9)

У цьому випадку побудована конфігурація з двовимірними елементами на множинах . Ця конструкція є двовимірною блок – схемою з параметрами , причому,вона є циклічною, бо кожний її блок можна отримати з попереднього блоку при збільшенні його двовимірних елементів на одиницю в полі чисел за відповідним модулем

На другому етапі здійснюється перетворення конфігурацій (9) в класичну блок – схему. Оскільки елементами конфігурації на множині елементів одновимірної блок – схеми з параметрами достатньо замінити двовимірні елементи одновимірними відповідниками і так отримати блок – схему з одновимірними елементами. Така заміна є правомірною, бо комбінаторні властивості блок – схем з фіксованими параметрами, описані залежностями (1), для обох конфігурацій залишаються незмінними. Тому, існує взаємно однозначна відповідність між класичними (одновимірними) та новоутвореними (двовимірними) комбінаторними конфігураціями. Це дає змогу легко перетворювати класичні блок – схеми на багатовимірні блок – схеми на векторних ІКВ, або навпаки шляхом простої заміни елементів однієї конфігурації відповідними елементами іншої. Взаємно однозначну відповідність елементів класичної та векторної конфігурацій зручно ілюструвати таблицею відповідності. Нижче наведена таблиця відповідності двовимірних елементів схеми (9) одновимірним елементам (натуральним числам).

*Таблиця 1*

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |

Після заміни двовимірних елементів схеми (9) натуральними числами згідно до таблиці 1 та їх упорядкування всередині блоків у зростаючому порядку отримаємо:

Упорядкувавши блоки в лексикографічному порядку, отримуємо завершальну блок – схема:

Синтезована блок – схема є зрівноваженою симетричною блок – схемою з параметрами:

(10)

Порівняно з одновимірними ідеальними кільцевими в’язанками багатовимірним ІКВ властива значно більша різновидність ізоморфних перетворень за допомогою ряду операцій зміщення, інверсії, перестановки, доповнення, множення на вектор – коефіцієнти [1]. Це дає можливість будувати нові конфігурації, використовуючи вищезгадані операції, зокрема операцію «доповнення», суть якої зводиться до заміни всіх елементів у кожному із наборів t – вимірної ІКВ, заданої на t – вимірному паралелепіпеді [1].

Наприклад, двовимірна ІКВ ((1,0),(1,1),(0,1),(0,2),(1,3)) при після операції доповнення перетворюється на послідовність

((2,0),(2,6),(0,6),(0,5),(2,4)), (11)

елементи яких обчислюються за вищевказаними правилами.

Легко перевірити, що послідовність (11) є одним із варіантів двовимірної ІКВ п’ятого (n=5) порядку з параметрами базової ІКВ (6). За цією послідовністю за допомогою формул (7) і (8) знаходимо двовимірну блок – схему з параметрами

Виконуючи операцію множення двовимірних елементів цієї схеми на вектор – коефіцієнт (1,3) в полі чисел за відповідними модулями та розмістити блоки в лексикографічному порядку, отримаємо кінцеву блок – схему: (12)

Блок – схема (12) є ще одним варіантом ВІВ – схеми з параметрами що утворена за допомогою двовимірної ІКВ. Ця схема є зрівноваженою симетричною ВІВ – схемою. На відміну від циклічної блок – схеми, яка відповідає одновимірній ІКВ (2,5,1,3,10) з параметрами , блок – схема (12) позбавлена циклічного автоморфізму. Описаний метод можна застосовувати не лише для синтезу зрівноважених комбінаторних конфігурацій, але й для побудови частково зрівноважених ВІВ – схем [1].

**Синтез частково зрівноважених блок-схем**

На відміну від зрівноважених блок – схем частково зрівноважені схеми дають змогу утворювати конфігурації з параметрами, що відрізняються від параметрів базових блок – схем. У таких конфігураціях кількість пар елементів у різних блоках може відрізнятися. Однак порушення рівноваги між кількістю окремих пар, що трапляються у блоках схеми не зменшує їхнього практичного значення, а у деяких випадках має свої переваги. Цю властивість можна застосовувати, наприклад, для побудови частково зрівноважених планів експерименту з двома асоціативними класами, коли деякі пари умов експерименту будуть порівнюватися точніше, ніж інші [3].

Для побудови частково зрівноваженої блок – схеми оберемо за основу послідовність двовимірних елементів (1.1) – (1.4), яка утворює ІКВ з параметрами на прямокутнику [1]. Перетворюючи цю послідовність на блок – схему за допомогою формул (7) і (8), знаходимо: (13)

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Система (13) є частково зрівноважена конфігурація з двовимірними елементами на множині причому вона є циклічною, бо кожний її блок можна отримати з попереднього блоку при збільшенні його двовимірних елементів на одиницю в полі чисел за модулем

Перехід від двовимірних елементів до одновимірних зручно здійснити за допомогою таблиці відповідності (таблиця 2).

*Таблиця 2*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |

Після заміни елементів з таблиці 2 і розміщення блоків в лексикографічному порядку отримаємо таку конфігурацію: (14)

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Схема (14) є частково зрівноваженою блок – схемою з параметрами що відрізняються від параметрів відповідної зрівноваженої блок – схеми , для якої, як відомо, Дробове значення параметра вказує на те, що не всі пари елементів в блоках цієї схеми представлені в однаковій кількості [4]. У побудованій конфігурації різні пари елементів трапляються у блоках не більше одного разу, але деякі пари не трапляються жодного разу. Різниця між кількістю усіх присутніх та відсутніх пар становить мінімальну можливу величину, завдяки чому, цю блок – схему доцільно використати для синтезу частково зрівноважених планів експерименту [3].

**Висновок**

Використання багатовимірних кільцевих в’язанок (ІКВ) для синтезу класичних комбінаторних конфігурацій розширює сферу вживання нетрадиційних методів дослідження систем різного призначення та їх практичного застосування у багатьох галузях науки та народного господарства. Описані методи та алгоритми демонструють можливість побудови математичних моделей систем з відповідними комбінаторними властивостями, що можуть знайти застосування в теорії та практиці планування експерименту. Ідеальні в’язанки дають змогу спрощувати синтез комбінаторних конфігурацій, а відтак побудову комбінаторних систем і розширити теоретичні дослідження в області комбінаторики.

**Список використаної літератури**

1. Холл М., Комбінаторика. – М, 1970. – 470 с.

2. Різник В.В. Синтез оптимальних комбінаторних систем. — Львів, 1989. – 168 с.

3. Бродський В.З.,Голикова Т.І. Таблиці планів експерименту для факторних і поліноміальних моделей – М, : Металургія, 1982. – 752 с.

4. Соломко М.Т. Синтез комбінаторних конфігурацій на рівні блок – схем за допомогою числових в’язанок : Дисертація … канд. техн. наук. – Львів,2000.

**Додаток 1**

**Код програми**

#include <vcl.h>

#pragma hdrstop

#include <math.h>

#include <vector.h>

#include <string.h>

#include <dstring.h>

#include <IniFiles.hpp>

#include "UnMain.h"

#include "UnLog.h"

#include "UnAbout.h"

#include "UnCl\_ikv.h"

#include "UnExplr.h"

#pragma package(smart\_init)

#pragma link "CSPIN"

#pragma link "Trayicon"

#pragma link "CGAUGES"

#pragma resource "\*.dfm"

TFm1 \*Fm1;

int PorPolinoma=3,

nmin,nmax, nmint,nmaxt,

a2min,a2max,

a1min,a1max,

a0min,a0max,

logtop=0,

prgs=0;

bool minimized=false, ThreadDone=true, Binder=false,

blShowResWindow=true,

OnlyOne, FindUniq, UseDB, AutoSave;

TDateTime calc0,

calc1;

TThdCalc \*ThdCalc;

AnsiString sPause="Пауза",sContinue="Продовжувати",

sFind1,sFind2="Шукати по одній ІКВ кожного порядку";

const HOTKEYS (20);

\_\_fastcall TFm1::TFm1(TComponent\* Owner): TForm(Owner)

{

}

bool \_\_fastcall isPrime(int n)

{

if (n<0) n=-n;

if (n!=2)

for(int i=2; i<=((n / 2)+1); i++)

if ((n % i)==0) return false;

return true;

}

int TFm1::ChangeToPrime(int n, bool fctp)

{

if (isPrime(n-1)) return n;

if (fctp)

do n++;

while (!isPrime(n-1));

else

do n--;

while (!isPrime(n-1));

return n;

}

void \_\_fastcall TFm1::SpEdnminChange(TObject \*Sender)

{

if (SpEdnmin->Value < SpEdnmin->MinValue)

{SpEdnmin->Value = SpEdnmin->MinValue;

return;

}

SpEdnmin->MaxValue = SpEdnmax->Value;

if (SpEdnmin->Value > SpEdnmin->MaxValue)

{SpEdnmin->Value = SpEdnmin->MaxValue;

return;

}

SpEdnmax->MinValue = SpEdnmin->Value;

bool fl=false;

if (SpEdnmin->Value > nmint) fl=true;

SpEdnmin->Value = ChangeToPrime(SpEdnmin->Value, fl);

nmint = SpEdnmin->Value;

if (SpEdnmax->Value != SpEdnmin->Value) ChBxOnlyOne->Caption = sFind2;

else ChBxOnlyOne->Caption = sFind1;

if (Binder)

SpEdnmax->Value = SpEdnmin->Value;

}

void \_\_fastcall TFm1::SpEdnmaxChange(TObject \*Sender)

{

if (SpEdnmax->Value > SpEdnmax->MaxValue)

{SpEdnmax->Value = SpEdnmax->MaxValue;

return;

}

bool fl=false;

if (SpEdnmax->Value > nmaxt) fl=true;

SpEdnmax->Value = ChangeToPrime(SpEdnmax->Value, fl);

nmaxt = SpEdnmax->Value;

SpEdnmin->MaxValue = SpEdnmax->Value;

SpEdnmax->MinValue = SpEdnmin->Value;

if (SpEdnmax->Value != SpEdnmin->Value) ChBxOnlyOne->Caption = sFind2;

else ChBxOnlyOne->Caption = sFind1;

if (Binder)

SpEdnmin->Value = SpEdnmax->Value;

}

void \_\_fastcall TFm1::SetWidthDef()

{

int i,w=0;

SGridSM->ColWidths[0] = 38;

for(i=1; i<SGridSM->ColCount-1; i++)

SGridSM->ColWidths[i] = 33;

for(i=0; i<SGridSM->ColCount-1; i++) w += SGridSM->ColWidths[i];

SGridSM->ColWidths[SGridSM->ColCount-1] = ;

SGridSM->Width-w-11;

for(i=0; i<HdrControl->Sections->Count; i++)

HdrControl->Sections->Items[i]->Width = SGridSM->ColWidths[i]+1;

HdrControl->Sections->Items[0]->Width +=2;

{

void \_\_fastcall TFm1::FormShow(TObject \*Sender)

{

if (Top>Screen->Height-Height-27) Top=0;

Application->Title = "ІКВ";

SpEdnmin->SetFocus();

StatusBar->Align = alClient;

SetWidthDef();

Binder = SpEdnmin->Value == SpEdnmax->Value;

DrawBinder();

}

void \_\_fastcall TFm1::NRestoreClick(TObject \*Sender)

{ TrayIcon1->Restore();

}

void \_\_fastcall TFm1::BitBtnCalcMdlClick(TObject \*Sender)

{

int i,j;

ThreadDone = false;

TrayIcon1->IconIndex=1; Update();

TimerLog->Enabled=false;

PopupMenuMdl->AutoPopup = true;

if (ChBxOnlyOne->Checked && SpEdnmin->Value==SpEdnmax->Value)

Application->Title = "IKB";

else

Application->Title = "0 - ІКВ";

BitBtnCalcMdl->Enabled = false;

MStart->Enabled = false;

NStart->Enabled = false;

BitBtnPause->Enabled = true; BitBtnPause->SetFocus();

MPause->Enabled = true;

NPause->Enabled = true;

BitBtnStop->Enabled = true;

MStop->Enabled = true;

NStop->Enabled = true;

SGridSM->Enabled = false;

for(i=0; i<SGridSM->RowCount; i++)

for(j=0; j<SGridSM->ColCount; j++)

SGridSM->Cells[j][i]="";

SGridSM->RowCount=1;

nmin = SpEdnmin->Value;

nmax = SpEdnmax->Value;

a2min = SpEda2min->Value;

a2max = SpEda2max->Value;

a1min = SpEda1min->Value;

a1max = SpEda1max->Value;

a0min = SpEda0min->Value;

a0max = SpEda0max->Value;

OnlyOne = ChBxOnlyOne->Checked;

FindUniq = ChBxFindUniq->Checked;

UseDB = ChBxUseDB->Checked;

AutoSave = ChBxAutoSave->Checked;

ThdCalc = new TThdCalc(true);

ThdCalc->OnTerminate = AfterCalc;

switch (CbBxPrior->ItemIndex){

case 0: ThdCalc->Priority = tpIdle; break;

case 1: ThdCalc->Priority = tpLower; break;

case 3: ThdCalc->Priority = tpHigher; break;

default: ThdCalc->Priority = tpNormal;

}

ThdCalc->Resume();

}

void \_\_fastcall TFm1::BitBtnPauseClick(TObject \*Sender)

{

if (SGridSM->Cells[0][0]!="")

{ MSave->Enabled = true;

MSave->ImageIndex = 2;

}

else

{ MSave->Enabled = false;

MSave->ImageIndex = 3;

}

if (BitBtnPause->Caption == sPause)

{

calc1 = Time();

BitBtnPause->Caption = sContinue;

ThdCalc->Suspend();

TimerMdl->Enabled = false;

}

else

{

calc0 += Time() - calc1;

BitBtnPause->Caption = sPause;

switch (CbBxPrior->ItemIndex){

case 0: ThdCalc->Priority = tpIdle; break;

case 1: ThdCalc->Priority = tpLower; break;

case 3: ThdCalc->Priority = tpHigher; break;

default: ThdCalc->Priority = tpNormal;

}

ThdCalc->Resume();

TimerMdl->Enabled = true;

}

}

void \_\_fastcall TFm1::BitBtnStopClick(TObject \*Sender)

{

if (Application->MessageBox("Зупинити процес синтезу ІКВ?",

"Увага!", MB\_ICONINFORMATION+MB\_YESNO)==IDYES)

{if (BitBtnPause->Caption == sContinue)

BitBtnPauseClick(Sender);

ThdCalc->Terminate();

}

}

void \_\_fastcall TFm1::AfterCalc(TObject \*)

{

int i,j,t,max,l;

CGauge->Visible = false;

for(i=0; i<SGridSM->ColCount; i++)

{ max=0;

for(j=0; j<SGridSM->RowCount; j++)

{t = SGridSM->Canvas->TextWidth(SGridSM->Cells[i][j]);

if (t>max) max=t;

}

SGridSM->ColWidths[i] = max + SGridSM->GridLineWidth +20;

}

SGridSM->ColWidths[0] = SGridSM->ColWidths[0] - 5;

for(i=0; i<HdrControl->Sections->Count; i++)

HdrControl->Sections->Items[i]->Width = SGridSM->ColWidths[i]+1;

HdrControl->Sections->Items[0]->Width +=2;

TGridRect Rect1;

Rect1.Left = 0; Rect1.Top = SGridSM->RowCount-1;

Rect1.Right = 0; Rect1.Bottom = SGridSM->RowCount-1;

SGridSM->Selection = Rect1;

if (SpBtnOptions->Down) t=12; else t=16;

if (SGridSM->RowCount>t) SGridSM->TopRow = SGridSM->RowCount-t;

SGridSM->Enabled=true;

SGridSM->SetFocus();

MessageBeep(4294967295);

TrayIcon1->IconIndex=0; Update();

TimerMdl->Enabled = false;

TimerMdlTimer(this);

BitBtnCalcMdl->Enabled = true;

MStart->Enabled = true;

NStart->Enabled = true;

BitBtnPause->Enabled = false; BitBtnPause->Caption = sPause;

MPause->Enabled = false;

NPause->Enabled = false;

BitBtnStop->Enabled = false;

MStop->Enabled = false;

NStop->Enabled = false;

if (!blShowResWindow) blShowResWindow = true;

else

{

if (SGridSM->Cells[0][0]!="")

{ MSave->Enabled = true;

MSave->ImageIndex = 2;

}

else

{ MSave->Enabled = false;

MSave->ImageIndex = 3;

}

if (SGridSM->Cells[0][0]=="")

{ FormLog->Img1->Visible = false;

FormLog->Img2->Visible = true;

FormLog->LblRes->Caption= "Не знайдено жодної ІКВ";

}

else

{ FormLog->Img1->Visible = true;

FormLog->Img2->Visible = false;

FormLog->LblRes->Caption="Знайдено "+ IntToStr(SGridSM->RowCount) +" ІКВ";

}

if (minimized)

{ RECT RWork;

SystemParametersInfo(SPI\_GETWORKAREA, 0, &RWork, 0);

FormLog->Position = poDesigned;

FormLog->BorderStyle = bsToolWindow;

FormLog->Left = RWork.right - FormLog->Width;

FormLog->Top = RWork.bottom-2;

}

else

{FormLog->Position = poMainFormCenter;

FormLog->BorderStyle = bsNone;

}

FormLog->Show();

if (minimized)

{logtop = 0;

do

{FormLog->Top = FormLog->Top - (FormLog->Height)/10;

FormLog->Repaint();

Sleep(100);

}while(++logtop<10);

FormLog->Top = FormLog->Top - 5;

TimerLog->Enabled=true;

}

else TimerLog->Enabled=true;

}

}

void \_\_fastcall TFm1::TimerLogTimer(TObject \*Sender)

{

FormLog->Hide();

TimerLog->Enabled=false;

}

void \_\_fastcall TFm1::HdrControlSectionResize(

THeaderControl \*HeaderControl, THeaderSection \*Section)

{

SGridSM->ColWidths[Section->Index] =

HdrControl->Sections->Items[Section->Index]->Width-1;

}

void \_\_fastcall TFm1::SpEda0minChange(TObject \*Sender)

{

if (SpEda0min->Value>SpEda0max->Value) SpEda0min->Value=SpEda0max->Value;

if (SpEda0max->Value<SpEda0min->Value) SpEda0max->Value=SpEda0min->Value;

SpEda0min->MaxValue = SpEda0max->Value;

SpEda0max->MinValue = SpEda0min->Value;

}

void \_\_fastcall TFm1::SpEda1minChange(TObject \*Sender)

{

if (SpEda1min->Value>SpEda1max->Value) SpEda1min->Value=SpEda1max->Value;

if (SpEda1max->Value<SpEda1min->Value) SpEda1max->Value=SpEda1min->Value;

SpEda1min->MaxValue = SpEda1max->Value;

SpEda1max->MinValue = SpEda1min->Value;

}

void \_\_fastcall TFm1::SpEda2minChange(TObject \*Sender)

{

if (SpEda2min->Value>SpEda2max->Value) SpEda2min->Value=SpEda2max->Value;

if (SpEda2max->Value<SpEda2min->Value) SpEda2max->Value=SpEda2min->Value;

SpEda2min->MaxValue = SpEda2max->Value;

SpEda2max->MinValue = SpEda2min->Value;

}

AnsiString FileExt(AnsiString fn)

{

int i=fn.Length()-1;

while(i>0){ if (fn[i]=='.') break; i--; }

if (i==0) return "";

else return fn.SubString(i+1, fn.Length());

}

void \_\_fastcall TFm1::TimerMdlTimer(TObject \*Sender)

{

if (SGridSM->Cells[0][0]=="")

{StatusBar->Panels->Items[1]->Text = " Не знайдено жодної ІКВ";

if (!(ChBxOnlyOne->Checked && SpEdnmin->Value==SpEdnmax->Value))

Application->Title = "0 - IKB";

}

else{

StatusBar->Panels->Items[1]->Text = " Знайдено " + IntToStr(SGridSM->RowCount) + " ІКВ";

if (!(ChBxOnlyOne->Checked && SpEdnmin->Value==SpEdnmax->Value))

Application->Title = IntToStr(SGridSM->RowCount)+" - IKB";

}

StatusBar->Panels->Items[0]->Text = "Минуло " + TimeToStr(Time()-calc0);

CGauge->Progress = prgs;

CGauge->Hint =

Format("Оброблено %d%s даних",ARRAYOFCONST((int(CGauge->Progress\*100/CGauge->MaxValue),"%")));

}

void \_\_fastcall TFm1::NClearTblClick(TObject \*Sender)

{

int i,j;

for(i=0; i<SGridSM->RowCount; i++)

for(j=0; j<SGridSM->ColCount; j++)

SGridSM->Cells[j][i]="";

SGridSM->RowCount = 1;

SetWidthDef();

PopupMenuMdl->AutoPopup = false;

StatusBar->Panels->Items[0]->Text = "";

StatusBar->Panels->Items[1]->Text = "";

Application->Title = "ІКВ";

MSave->Enabled = false;

}

void \_\_fastcall TFm1::NSaveRowClick(TObject \*Sender)

{

int i; FILE \*outfile;

SaveDlg->FileName = "ikv\_N"+SGridSM->Cells[1][SGridSM->Row];

SaveDlg->InitialDir = ExtractFilePath(ParamStr(0));

if (SaveDlg->Execute())

{if (FileExt(SaveDlg->FileName).Length()==0)

SaveDlg->FileName = SaveDlg->FileName + ".txt";

if (FileExists(SaveDlg->FileName))

{ if (Application->MessageBox("Файл з ім’ям, яке ви вказали вже існує. Перезаписати?",

"Увага!", MB\_ICONINFORMATION+MB\_YESNO) == IDNO) return;

}

outfile = fopen(SaveDlg->FileName.c\_str(), "wt");

if (outfile)

{fprintf(outfile, "n\ta2\ta1\ta0\tІКВ\n");

AnsiString str;

str="";

for(i=1; i<SGridSM->ColCount; i++)

str += SGridSM->Cells[i][SGridSM->Row]+"\t";

fprintf(outfile, "%s\n", str);

}

fclose(outfile);

}

}

void \_\_fastcall TFm1::ApplicationEvents1Minimize(TObject \*Sender)

{

minimized = true;

}

void \_\_fastcall TFm1::ApplicationEvents1Restore(TObject \*Sender)

{

minimized = false;

if (FormLog->Active) TimerLogTimer(Sender);

}

bool StrToIntConvert(AnsiString s)

{

int i; bool fl=true;

if (s=="") return false;

for(i=1; i<=s.Length(); i++)

if (!((s[i]>='0') && (s[i]<='9')) && (s[i]!='+') && (s[i]!='-'))

{ fl=false; break; }

try{ StrToInt(s); }

catch(...){ fl=false; }

return fl;

}

void \_\_fastcall TFm1::SpEdpKeyDown(TObject \*Sender, WORD &Key,

TShiftState Shift)

{

if (Key==VK\_RETURN) ((TCSpinEdit\*)Sender)->ReadOnly=true;

}

void \_\_fastcall TFm1::SpEdpKeyUp(TObject \*Sender, WORD &Key,

TShiftState Shift)

{

if (Key==VK\_RETURN) ((TCSpinEdit\*)Sender)->ReadOnly=false;

}

void \_\_fastcall TFm1::FormCreate(TObject \*Sender)

{

sFind1 = ChBxOnlyOne->Caption;

AnsiString fname=ChangeFileExt(Application->ExeName, ".INI");

nmint = SpEdnmin->Value;

nmaxt = SpEdnmax->Value;

if (!FileExists(fname)) return;

TIniFile \*ini;

ini = new TIniFile(fname);

try

{Top = ini->ReadInteger( "Form", "Top", 0 );

Left = ini->ReadInteger( "Form", "Left", 0 );

SpEdnmax->Value = ini->ReadInteger( "InitParam", "nmax", 8 );

SpEdnmin->Value = ini->ReadInteger( "InitParam", "nmin", 8 );

SpEda0min->Value = ini->ReadInteger( "InitParam", "a0min", -2 );

SpEda0max->Value = ini->ReadInteger( "InitParam", "a0max", 2 );

SpEda1min->Value = ini->ReadInteger( "InitParam", "a1min", -2 );

SpEda1max->Value = ini->ReadInteger( "InitParam", "a1max", 2 );

SpEda2min->Value = ini->ReadInteger( "InitParam", "a2min", -2 );

SpEda2max->Value = ini->ReadInteger( "InitParam", "a2max", 2 );

ChBxOnlyOne->Checked = ini->ReadBool("Options","OnlyOne",false);

ChBxFindUniq->Checked = ini->ReadBool("Options","FindUniq",false);

CbBxPrior->ItemIndex = ini->ReadInteger("Options","Priority",2);

ChBxUseDB->Checked = ini->ReadBool("Options","UseDB",true);

ChBxAutoSave->Checked = ini->ReadBool("Options","AutoSave",true);

EdLocateDB->Text = ini->ReadString("Options","LocateDB","ikv.dbf");

}

\_\_finally { delete ini; }

}

void \_\_fastcall TFm1::FormClose(TObject \*Sender, TCloseAction &Action)

{

TIniFile \*ini;

ini = new TIniFile(ChangeFileExt( Application->ExeName, ".INI" ) );

try

{ini->WriteInteger("Form", "Top", Top);

ini->WriteInteger("Form", "Left", Left);

ini->WriteInteger("InitParam", "nmin", SpEdnmin->Value);

ini->WriteInteger("InitParam", "nmax", SpEdnmax->Value);

ini->WriteInteger("InitParam", "a0min", SpEda0min->Value);

ini->WriteInteger("InitParam", "a0max", SpEda0max->Value);

ini->WriteInteger("InitParam", "a1min", SpEda1min->Value);

ini->WriteInteger("InitParam", "a1max", SpEda1max->Value);

ini->WriteInteger("InitParam", "a2min", SpEda2min->Value);

ini->WriteInteger("InitParam", "a2max", SpEda2max->Value);

ini->WriteBool("Options", "OnlyOne", ChBxOnlyOne->Checked);

ini->WriteBool("Options", "FindUniq", ChBxFindUniq->Checked);

ini->WriteInteger("Options", "Priority", CbBxPrior->ItemIndex);

ini->WriteBool("Options", "UseDB", ChBxUseDB->Checked);

ini->WriteBool("Options", "AutoSave", ChBxAutoSave->Checked);

ini->WriteString("Options", "LocateDB", EdLocateDB->Text);

}

\_\_finally { delete ini; }

}

int \_\_fastcall PowerN(int n)

{

if (n==2) return 1;

int i,j,t,S,fi,m;

S = n\*(n-1)+1;

if (isPrime(S)) fi=S-1;

else

{fi=1; t=S;

for(i=3; t>1 && i<=S/2+1; i++)

{if (!(i%2)) continue;

m=0;

while (!(t % i))

{ m++; t/=i;

}

if (m>0) fi \*= pow(i,m)-pow(i,m-1);

}

}

return fi/6;

}

void \_\_fastcall TFm1::CreateDB(AnsiString fnameDB)

{

Tbl->FieldDefs->Clear();

TFieldDef \*pNewDef = Tbl->FieldDefs->AddFieldDef();

pNewDef->Name="N";

pNewDef->DataType = ftInteger;

pNewDef=Tbl->FieldDefs->AddFieldDef();

pNewDef->Name="K2";

pNewDef->DataType = ftInteger;

pNewDef=Tbl->FieldDefs->AddFieldDef();

pNewDef->Name="K3";

pNewDef->DataType = ftInteger;

pNewDef=Tbl->FieldDefs->AddFieldDef();

pNewDef->Name="A2";

pNewDef->DataType = ftInteger;

pNewDef=Tbl->FieldDefs->AddFieldDef();

pNewDef->Name="A1";

pNewDef->DataType = ftInteger;

pNewDef=Tbl->FieldDefs->AddFieldDef();

pNewDef->Name="A0";

pNewDef->DataType = ftInteger;

pNewDef=Tbl->FieldDefs->AddFieldDef();

pNewDef->Name="IKB";

pNewDef->DataType = ftMemo;

pNewDef->Size = 255;

Tbl->IndexDefs->Clear();

Tbl->IndexDefs->Add("N","N",TIndexOptions()<<ixPrimary);

Tbl->CreateTable();

}

\_\_fastcall TThdCalc::TThdCalc(bool CreateSuspended): TThread(CreateSuspended)

{

}

void \_\_fastcall TThdCalc::Execute()

{

FreeOnTerminate = true;

AnsiString ikv\_str;

int i,j,a0,a1,a2, t, kt,S,sm[3][3], \*ikv, maxdln,k1,sz,powerN,k\_ikvN, kn;

short M, b[3], bt[3];

int \*tbl, \*mt,\*mti, N,l, \*ap1,\*ap2,\*ap3;

bool ideal,fc,CheckDBF,IsInDBF=false,IsInODB=false;

vector<int> dln,Z; unsigned iu;

vector<ikv1> ikvDB;

Screen->Cursor = crAppStart;

try{

if (UseDB)

{//перевіряємо чи є такий файл

AnsiString fnameDB;

if (Fm1->EdLocateDB->Text==ExtractFileName(Fm1->EdLocateDB->Text))

fnameDB = ExtractFilePath(Application->ExeName) + Fm1->EdLocateDB->Text;

else fnameDB = Fm1->EdLocateDB->Text;

Fm1->Tbl->DatabaseName = ExtractFilePath(fnameDB);

Fm1->Tbl->TableName = fnameDB;

Fm1->Tbl->TableType = ttFoxPro;

if (AutoSave && !FileExists(fnameDB))

Fm1->CreateDB(fnameDB);

Fm1->Tbl->Active = false;

try

{ Fm1->Tbl->Exclusive = true;

Fm1->Tbl->Active = true;

}

catch(EDatabaseError &E)

{ if (FileExists(fnameDB))

{ Application->MessageBox("БД зайнята! Закрийте програму, яка працює з цією БД.",

"Увага!", MB\_ICONWARNING+MB\_OK);

blShowResWindow = false;

goto lbl\_stop;

}

else

if (Application->MessageBox("Файл бази даних не знайдено! \n Відключити використання БД?",

"Увага!", MB\_ICONWARNING+MB\_YESNO)==IDYES)

Fm1->ChBxUseDB->Checked = UseDB = false;

else

{blShowResWindow = false;

goto lbl\_stop;

}

}

}

Fm1->TimerMdl->Enabled = true;

calc0 = Time();

Fm1->StatusBar->Panels->Items[0]->Text = "Минуло 0:00:00";

Fm1->StatusBar->Panels->Items[1]->Text = " ІКВ не знайдено";

Fm1->CGauge->MaxValue = (nmax-nmin+1)\*

(a2max-a2min+1)\*(a1max-a1min+1)\*(a0max-a0min+1);

prgs=0;

Fm1->CGauge->Visible = true;

for(N=nmin; N<=nmax; N++)

if (!isPrime(N-1))

prgs += (a2max-a2min+1)\*(a1max-a1min+1)\*(a0max-a0min+1);

else

{if (FindUniq)

{k\_ikvN=0;

powerN = PowerN(N);

}

if (UseDB)

{ //накладаємо фільтр

Fm1->Tbl->Filtered = false;

Fm1->Tbl->Filter = "N='"+IntToStr(N)+"'";

Fm1->Tbl->Filtered = true;

}

for(a0=a0min; a0<=a0max; a0++)

for(a1=a1min; a1<=a1max; a1++)

for(a2=a2min; a2<=a2max; a2++)

{prgs++;

if (!a0) { kn++; continue; }

if (Terminated) goto lbl\_stop;

dln.clear();

maxdln = abs(a0)/2;

for(i=2; i<=maxdln; i++)

if (!(abs(a0) % i)) dln.push\_back(i);

dln.push\_back(1);

if (abs(a0)!=1) dln.push\_back(abs(a0));

fc=false;

for(iu=0; iu<dln.size(); iu++)

if (!(dln[iu]\*dln[iu]\*(a2+dln[iu])+a1\*dln[iu]+a0) ||

!(dln[iu]\*dln[iu]\*(a2-dln[iu])-a1\*dln[iu]+a0))

{ fc=true; break; }

if (fc) { kn++; continue; }

S = N\*(N-1)+1; M = N-1;

sm[0][2] = -a0; sm[1][2] = -a1; sm[2][2] = -a2;

for(i=0;i<3;i++)

for(j=0;j<2;j++)

if (i-1==j) sm[i][j]=1; else sm[i][j]=0;

for(i=0;i<3;i++) \*(b+i) = 0;

\*b = 1;

Z.clear(); Z.push\_back(1);

for(l=1;l<S;l++)

{for(i=0;i<3;i++)

{for(t=0,j=0;j<3;j++) t+=sm[i][j] \* \*(b+j);

\*(bt+i) = t % M;

}

for(i=0;i<3;i++) \*(b+i) = \*(bt+i);

if (!\*(b+2)) Z.push\_back(l+1);

}

ikv=new int[N];

for(i=1; i<N; i++)

\*(ikv+i-1) = Z[i]-Z[i-1];

\*(ikv+N-1) = S-Z[N-1]+1;

AnsiString st1;

for(i=0; i<N; i++)

st1 += IntToStr(\*(ikv+i))+" ";

st1.Delete(st1.Length(),1);

fc=true;

for(i=1; i<N; i++)

if (\*(ikv+i)==2) { fc=false; break; }

if (fc) continue;

fc=false;

for(i=0; i<N; i++)

for(j=0; j<N; j++)

if (i!=j && \*(ikv+i)==\*(ikv+j))

{ fc = true; goto lbl\_st2;

}

lbl\_st2: if (fc) continue;

if (Terminated) goto lbl\_stop;

if (\*ikv!=1)

{

int t=1;

while((\*(ikv+t) != 1) && (t<N)) t++;

int \*atmp = new int[t];

try

{

ap1=atmp; ap2=ikv; ap3=ikv+t;

while(ap2<ap3) \*(ap1++) = \*(ap2++);

ap1=ikv; ap2=ikv+t; ap3=ikv+N;

while(ap2<ap3) \*(ap1++) = \*(ap2++);

ap1=ikv+N-t; ap2=atmp;

while(ap1<ap3) \*(ap1++) = \*(ap2++);

}

\_\_finally

{

delete[] atmp;

atmp = NULL;

}

}

if (\*(ikv+1) > \*(ikv+N-1))

{ap1=ikv+1; ap2=ikv+N-1; ap3=ikv+N/2;

if (N%2) ap3++;

while(ap1<ap3)

{ t = \*ap1; \*ap1++ = \*ap2; \*ap2-- = t;

}

}

ideal = true;

if (FindUniq)

for(iu=0; iu<ikvDB.size(); iu++)

{if (ikvDB[iu].N == N)

{IsInODB = true;

for(i=0;i<N;i++)

if (ikvDB[iu].v[i] != \*(ikv+i))

{ IsInODB = false; break;

}

}

else IsInODB = false;

if (IsInODB) break;

}

CheckDBF = false;

ikv\_str.Delete(1,ikv\_str.Length());

for(i=0; i<N; i++) ikv\_str += IntToStr(\*(ikv+i))+" ";

ikv\_str.Delete(ikv\_str.Length(),1);

if (((FindUniq && !IsInODB) || !FindUniq) && UseDB)

{CheckDBF = true;

IsInDBF = false;

Fm1->Tbl->First();

while (!Fm1->Tbl->Eof)

{if ( ((Fm1->Tbl->FieldByName("A2")->AsInteger == a2) &&

(Fm1->Tbl->FieldByName("A1")->AsInteger == a1) &&

(Fm1->Tbl->FieldByName("A0")->AsInteger == a0)) ||

((Fm1->Tbl->FieldByName("K2")->AsInteger == ikv[1]) &&

(Fm1->Tbl->FieldByName("K3")->AsInteger == ikv[2]) &&

(!ikv\_str.AnsiCompare(Fm1->Tbl->FieldByName("IKB")->AsString) )))

{IsInDBF = true;

break;

}

Fm1->Tbl->Next();

}

}

if (!IsInODB && !IsInDBF)

{

sz=S/2+N;

tbl = new int[sz];

for(i=0; i<N; i++) \*(tbl+i)= ikv[i];

for(mt=tbl, mti=tbl+N, i=N; i<sz; i++)

\*mti++ = \*mt++ + \*(tbl+(i/N+i) % N);

N++;

mt = tbl+sz;

for(l=3; l<N; l++)

{for(kt=0, mti=tbl; mti<mt;)

if (\*mti++==l)

if (++kt>1)

{ ideal=false; break; }

if (!ideal) break;

if (!kt)

{ ideal=false; break; }

}

delete[]tbl;

N--;

}

if (!ideal)

{ delete[]ikv;

continue;

}

if ((!FindUniq) || (FindUniq && !IsInODB))

{

if (Fm1->SGridSM->Cells[0][0]!="") Fm1->SGridSM->RowCount++;

k1= Fm1->SGridSM->RowCount-1;

Fm1->SGridSM->Cells[0][k1]= IntToStr(k1+1);

Fm1->SGridSM->Cells[1][k1]= IntToStr(N);

Fm1->SGridSM->Cells[2][k1] = IntToStr(a2);

Fm1->SGridSM->Cells[3][k1] = IntToStr(a1);

Fm1->SGridSM->Cells[4][k1] = IntToStr(a0);

Fm1->SGridSM->Cells[5][k1] = ikv\_str;

}

if (FindUniq && !IsInODB)

{

ikvDB.push\_back(ikv1(ikv,N));

k\_ikvN++;

}

if (UseDB && AutoSave)

{if (!CheckDBF)

{

IsInDBF = false;

Fm1->Tbl->First();

while (!Fm1->Tbl->Eof)

{if ( ((Fm1->Tbl->FieldByName("A2")->AsInteger == a2) &&

(Fm1->Tbl->FieldByName("A1")->AsInteger == a1) &&

(Fm1->Tbl->FieldByName("A0")->AsInteger == a0)) ||

((Fm1->Tbl->FieldByName("K2")->AsInteger == ikv[1]) &&

(Fm1->Tbl->FieldByName("K3")->AsInteger == ikv[2]) &&

(!ikv\_str.AnsiCompare(Fm1->Tbl->FieldByName("IKB")->AsString) )))

{IsInDBF = true;

break;

}

Fm1->Tbl->Next();

}

}

if (!IsInDBF)

{

Fm1->Tbl->Append();

Fm1->Tbl->FieldValues["N"] = N;

Fm1->Tbl->FieldValues["K2"] = ikv[1];

Fm1->Tbl->FieldValues["K3"] = ikv[2];

Fm1->Tbl->FieldValues["A2"] = a2;

Fm1->Tbl->FieldValues["A1"] = a1;

Fm1->Tbl->FieldValues["A0"] = a0;

Fm1->Tbl->FieldValues["IKB"] = ikv\_str;

Fm1->Tbl->Post();

}

}

delete[]ikv;

if ((FindUniq && k\_ikvN==powerN) || OnlyOne)

if (nmin == nmax) goto lbl\_stop;

else {a0=a0max; a1=a1max; break;}

}

}

}\_\_finally

{ Screen->Cursor = crDefault;

ikvDB.clear();

if (Fm1->ChBxUseDB->Checked)

Fm1->Tbl->Active = false;

}

lbl\_stop:

ThreadDone = true;

}

void \_\_fastcall TFm1::MExitClick(TObject \*Sender)

{

Close();

}

void \_\_fastcall TFm1::MAboutClick(TObject \*Sender)

{

TFmAbout \*a = NULL;

\_\_try

{ a = new TFmAbout(Application);

a->ShowModal();

}

\_\_finally

{ if (a)

delete a;

}

}

void \_\_fastcall TFm1::MSaveClick(TObject \*Sender)

{

int i,j; FILE \*outfile;

SaveDlg->FileName = "Result";

SaveDlg->InitialDir = ExtractFilePath(ParamStr(0));

if (SaveDlg->Execute())

{if (FileExt(SaveDlg->FileName).Length()==0)

SaveDlg->FileName = SaveDlg->FileName + ".txt";

if (FileExists(SaveDlg->FileName))

{ if (Application->MessageBox("Файл з ім’ям, яке ви вказали вже існує. Перезаписати?",

"Увага!", MB\_ICONINFORMATION+MB\_YESNO) == IDNO) return;

}

outfile = fopen(SaveDlg->FileName.c\_str(), "wt");

if (outfile)

{fprintf(outfile, "#\tn\ta2\ta1\ta0\tІКВ\n");

AnsiString str;

for(i=0; i<SGridSM->RowCount; i++)

{str="";

for(j=0; j<SGridSM->ColCount; j++)

str += SGridSM->Cells[j][i]+"\t";

fprintf(outfile, "%s\n", str);

}

}

fclosen(file);

void \_\_fastcall TFm1::FormKeyDown(TObject \*Sender, WORD &Key,

TShiftState Shift)

{

if (Key==VK\_PAUSE) BitBtnPauseClick(Sender);

if (Key==VK\_F9)

{SpBtnOptions->Down = !SpBtnOptions->Down;

SpBtnOptions->Click();

}

}

void \_\_fastcall TFm1::SpBtnOptionsClick(TObject \*Sender)

{

if (SpBtnOptions->Down) PanelMng->Height = 185;

else PanelMng->Height = 118;

}

void \_\_fastcall TFm1::ApplicationEvents1Deactivate(TObject \*Sender)

{

if (FormLog->Visible)

FormLog->Close();

}

void \_\_fastcall TFm1::BtnLocateDBClick(TObject \*Sender)

{

OpenDlg->FileName = EdLocateDB->Text;

OpenDlg->InitialDir = ExtractFilePath(Application->ExeName);

if (OpenDlg->Execute())

{ EdLocateDB->Text = OpenDlg->FileName;

}

}

void \_\_fastcall TFm1::ChBxUseDBClick(TObject \*Sender)

{

bool fl=ChBxUseDB->Checked;

ChBxAutoSave->Enabled = fl;

EdLocateDB->Enabled = fl;

BtnLocateDB->Enabled = fl;

}

void \_\_fastcall TFm1::DrawBinder()

{

int h,w;

w = ImgBind->Width;

h = ImgBind->Height;

ImgBind->Canvas->Brush->Color = clBtnFace;

ImgBind->Canvas->FillRect(Rect(0,0,w,h));

if (Binder)

{ImgBind->Canvas->Pen->Width = 2;

ImgBind->Canvas->RoundRect(0,1,w,h,w/2,h/2);

}

else

{ImgBind->Canvas->Pen->Width = 1;

ImgBind->Canvas->RoundRect(0,1,w,h,w/2-2,h/2);

ImgBind->Canvas->FillRect(Rect(w-2,h/2-4,w,h/2+4));

}

ImgBind->Canvas->FillRect(Rect(0,h/2-4,2,h/2+4));

}

void \_\_fastcall TFm1::ImgBinderClick(TObject \*Sender)

{

Binder = !Binder;

DrawBinder();

SpEdnmin->Value = SpEdnmax->Value;

}

AnsiString \_\_fastcall TimeToStrM(TDateTime Time)

{

Word Hour,Min,Sec,MSec;

DecodeTime(Time,Hour,Min,Sec,MSec);

return IntToStr(Hour)+":"+IntToStr(Min)+":"+IntToStr(Sec)+":"+IntToStr(MSec);

}

void \_\_fastcall TFm1::MExplClick(TObject \*Sender)

{

TFmExpl \*a = NULL;

\_\_try

{a = new TFmExpl(Application);

if (SGridSM->Cells[0][0]!="")

{a->SpEdN->Value = StrToInt(SGridSM->Cells[1][SGridSM->Row]);

for(int i=2; i<5; i++)

a->StrGrida->Cells[i-2][0] = SGridSM->Cells[i][SGridSM->Row];

}

a->ShowModal();

}

\_\_finally

{if (a)

delete a;

}

}

void \_\_fastcall TFm1::FormCloseQuery(TObject \*Sender, bool &CanClose)

{

if (!ThreadDone &&

Application->MessageBox("Виконується синтез ІКВ.\nЗупинити його і вийти з програми?",

"Увага!",MB\_ICONQUESTION+MB\_YESNO)==IDNO)

CanClose = false;

}

void \_\_fastcall TFm1::MContentClick(TObject \*Sender)

{

Application->HelpCommand(HELP\_CONTENTS, 0);

}

void \_\_fastcall TFm1::FormDestroy(TObject \*Sender)

{

Application->HelpCommand(HELP\_QUIT, 0);

}

void \_\_fastcall TFm1::MHotKeysClick(TObject \*Sender)

{

Application->HelpContext(HOTKEYS);

}