6 КӨПТЕРЕЗЕЛІ ҚОСЫМШАЛАР

6.1 Негізгі «батырмалық» форманы құру

Оқу құралының алдынғы бөлімдерінде бір терезелік қосымшалар қарастырылған болатын, онда барлық басқару элементтері және бағдарлама жұмысының нәтижелері негізгі форманың бір терезесінде орналасады. Осы бөлімде көптерезелі интерфейсті қосымшаларды жобалау технологиясын (Multiple-document interface, MDI) қарастырамыз.

Көптеген көптерезелі қосымшаларда негізгі «батырмалық» форма қолданылады, қосымшаның негізгі формасында меню түсініктемелері бар «үлкен» батырмалар түрінде орындалған.

Осындай жағдайларда негізгі форманы «батырмалық» негізгі форма түрінде жобалау керек пе? Осы жол кең қолданылады. Құрылатын жобада пайдаланушыға бірнеше әртүрлі сервистер ұсынылатын болса, пайдаланушы жұмысты бастаған уақытта өзіне керекті сервисті таңдайды. Негізгі формада пайдаланушыға керекті сервистері бар меню болуы мүмкін. Егер де әрбір сервис күрделі және осы сервиске жеке интерфейс керек болатын болса, онда бұл жағдайда стандартты менюдің орнына «батырмалық» негізгі форманы қолдану ыңғайлы болады. Меню командалары ретінде формадағы командалық батырмалар қолданылады, меню командасын таңдау және командалық батырманы басу бір-біріне балама болып келеді.

6.1-есеп. Қосымшада тіктөртбұрышты типіндегі 6 геометриялық фигуралардан тұратын массивті құру керек. Тіктөртбұрыштар (тікбұрыштар) екі қарама-қарсы тұрған төбелердің координаттарымен беріледі. Төбелердің координаттары кездейсоқ минус 100-ден 100-ге дейінгі аралығындағы бүтін сандардан құрылады. Пайдаланушы интерфейсінде 3 форма болуы керек: негізгі «батырмалық» форма терезесі: ақпаратты көрсету, редакциялау формасының терезесі; ақпаратты графикалық түрде көрсету терезесі.

Сонымен, негізгі терезеде бағдарлама жұмысының режімдеріне арналған 3 батырма болуы керек:

– тіктөртбұрыштар массивін құру;

– тіктөртбұрыштар төбелері координаттарының мәндерін көрсету мен редакциялауды кесте түрінде көрсететін терезеге көшу;

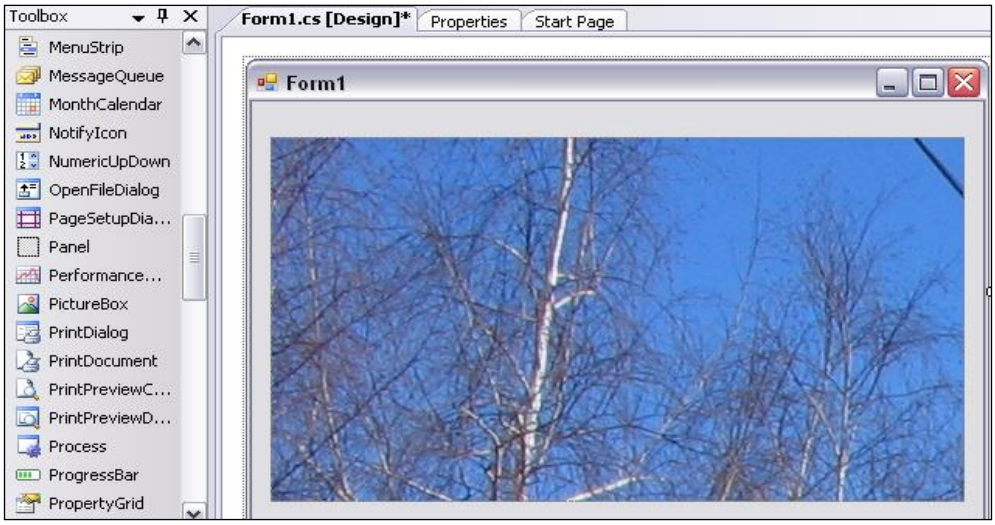
– тіктөртбұрыштарды графикалық түрде көрсететін терезеге көшу.

Негізгі формаға қосымшаның авторы туралы ақпаратты көрсететін қосымша батырманы қосуға болады.

Бағдарлама жұмысының режімдерін менюді пайдалануға болады, бірақ мысал ретінде біз «батырмалық» формаларды ұйымдастыруды қарастырамыз.

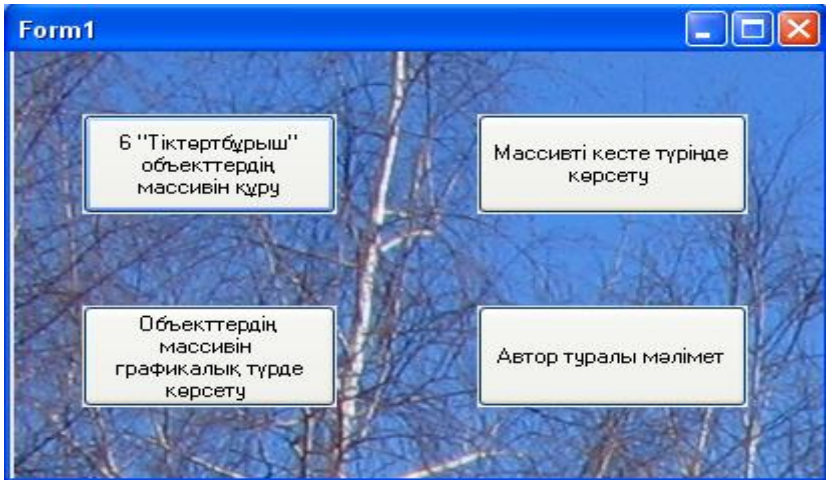
Бір ғана негізгі формасы бар жобаны дайындайық. Оған суретті немесе фотосуретті орналастырайық. Ол үшін басқару элементтерінің терезесінде PictureBox элементін таңдап, оны формаға орналастыруымыз керек. PictureBox элементінің қасиеттерінде Image қасиетін таңдап, оның оң

жағында кескін файлын таңдау бойынша диалогтық терезені ашу керек. Керекті кескінді тауып, оны таңдау керек. Осыдан кейін қосымшаның негізгі формасының терезесінде сурет пайда болады (6.1-сурет).



6.1-сурет – Форма терезесінде суретті орналастыру

Суреттің үстіне төрт батырманы бағдарлама жұмысының режімдеріне сәйкес орналастырамыз (6.2-сурет). Біздің қосымша үш режімде жұмыс істейді және терезеге автор туралы қосымша батырма енгізілді. Бағдарлама менюінде жұмыстың қосымша режімдері қосылады, мысалы, Help режимі – бағдарламамен жұмыс жасау бойынша нұсқау мен қосымша туралы барлық ақпаратты шығару.

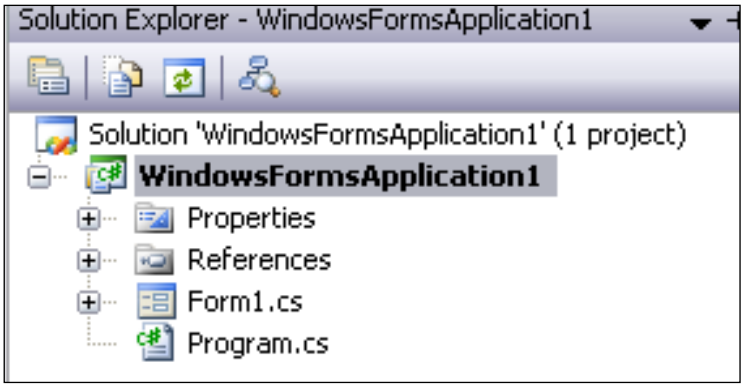


6.2-сурет – Қосымшаның негізгі «батырмалы формасы»

6.2 Қосымшаның жаңа түрлерін қосу

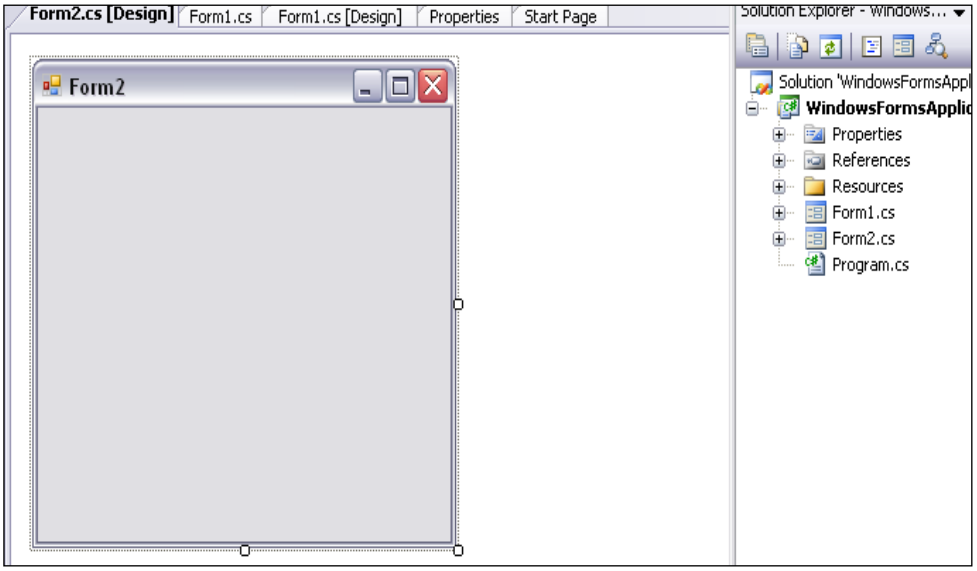
Жобаның қосымшасына жаңа формаларды қосудың бірнеше жолдары бар. Екі негізгі форманы қарастырайық.

Жобаға жаңа форманы қосу үшін Solution Explorer терезесінде жоба атауын білдіретін жолды – WindowsFormsAplication1 тышқанның оң жақ пернесімен шертіңіз (6.3-сурет).



6.3-сурет – Жобаға форманы қосудың бірінші нұсқасы

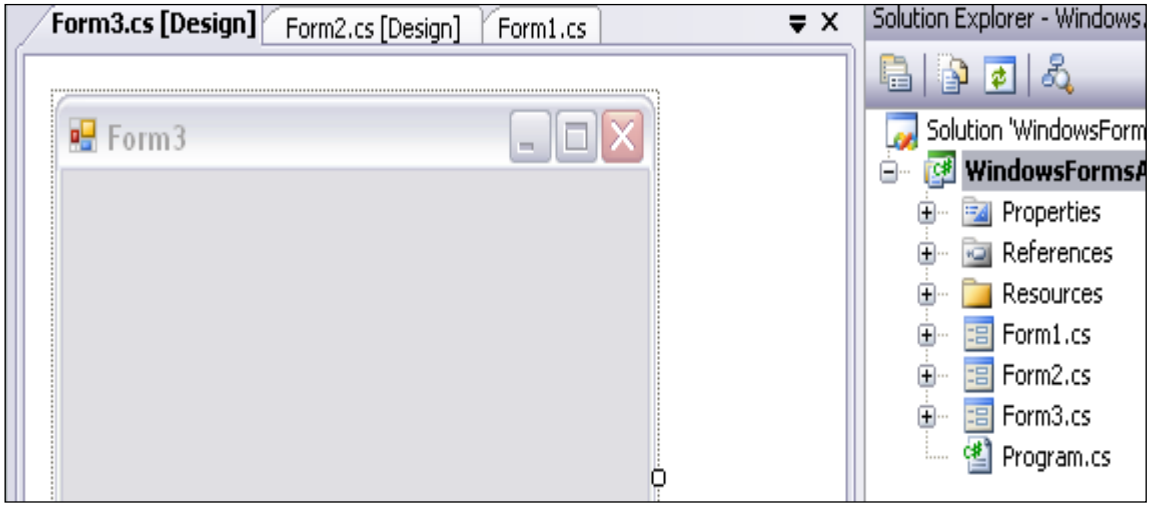
Пайда болған менюде Add режімін, оның ішінде Add Windows Form командасын таңдаңыз (6.4-сурет).



6.4-сурет – Жоба қосымшасына жаңа форманы қосу

WindowsFormsAplication1 жобасының Solution Explorer терезесінде Form2.cs жазуы пайда болды.

Екінші нұсқа бойынша Project режімін, оның ішінде Add Windows Form командасын таңдау керек. Одан кейін форма атын Add батырмасы арқылы растаңыз.



6.5-сурет – Жобаға үшінші форманы қосу

Формалардың қосылуын Solution Explorer терезесінде немесе формаларды редакциялау терезесінде тексеруге болады.

6.3 Негізгі форма оқиғаларының өңдеуіштері

6 тіктөртбұрышты массивті құру режімін негізгі формада орындауға болады, ал осы режім жұмысының нәтижелері 2 және 3-формаларда көрсетілген.

Негізгі форманың коды:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace WindowsFormsApplication1
{
 public partial class Form1 : Form
 {
 public static int[,] a = new int[6, 6];
 public Form1()
 {
 InitializeComponent();
 }
 private void button1_Click(object sender, EventArgs e)
 {
 Random rnd = new Random();
 for (int i = 0; i < 6; i++)
 {
 for (int j = 0; j < 4; j++)
```

```
{
a[i, j] = rnd.Next() % 61 - 100;
}
}
for (int i = 0; i < 6; i++)
{
if(Math.Abs(a[i,0]-a[i, 2]) == Math.Abs(a[i, 1] - a[i, 3]))
a[i, 4] = 1; else a[i, 4] = 0;
a[i,5]=Math.Abs(a[i,0]-a[i,2])*Math.Abs(a[i,1]-a[i,3]);
}
}
private void button2_Click(object sender, EventArgs e)
{
int y;
Form2 f2 = new Form2();
f2.ShowDialog();
}
private void button3_Click(object sender, EventArgs e)
{
Form3 f3 = new Form3();
f3.ShowDialog();
}
private void button4_Click(object sender, EventArgs e)
{
Form4 f4 = new Form4();
f4.Show();
}
}
}
```

Массивті құру кезінде 5-ші бағанаға тіктөртбұрыш, 0- квадрат, 1 – қарапайым тіктөртбұрыш туралы мәлімет, ал 6-шы бағанаға тіктөртбұрыш ауданының мәні жазылады.

Жоба бірнеше формадан тұрады. Сондықтан, «Бірнеше форманы бір уақытта ашуға және бір формадан келесі формаға қалай өтуге болады?» деген сұрақ туындайды. Бұл сұрақтың жауабы терезенің қалай ашылғанына байланысты.

Әрбір терезені (форманы) модальді ("диалогтық терезе") немесе модальді емес (қарапайым терезе) терезе түрінде ашуға болады.

Егер терезе Show() әдісімен ашылатын болса, онда ол қарапайым терезені ашады. Егер терезе ShowDialog() әдісімен ашылатын болса, онда ол диалогтық терезені ашады. Айырмашылығы неде? Диалогтық терезеден диалогты аяқтамай, форманы жаппай шығуға болмайды.

Диалогтық терезені ашып алғаннан кейін, басқа формамен жұмыс жасауға көшуге болмайды. Диалогтық терезені жабудың бірнеше жолы бар. Форманың жоғарғы оң жақ бұрышында орналасқан кресті немесе арнайы батырманы басуға болады.

Егер форма Show әдісімен ашылса, онда ашық формамен жұмысты аяқтамастан негізгі формаға немесе басқа модальді емес формаға көшіп, керекті мәліметті алғаннан кейін алғашқы формаға қайтып келуге болады.

Біз қосымшада терезені ашудың екі әдісін де қарастыратын боламыз.

Терезені жабу үшін екі әдісті қолдануға болады – Hide() және Close(). Осы әдістердің біріншісі форманы жасырады, екіншісі - жабады. Диалогтық терезе үшін Hide() немесе Close() әдістерін қолдануға болады: екі жағдайда да диалогтық терезе жабылады.

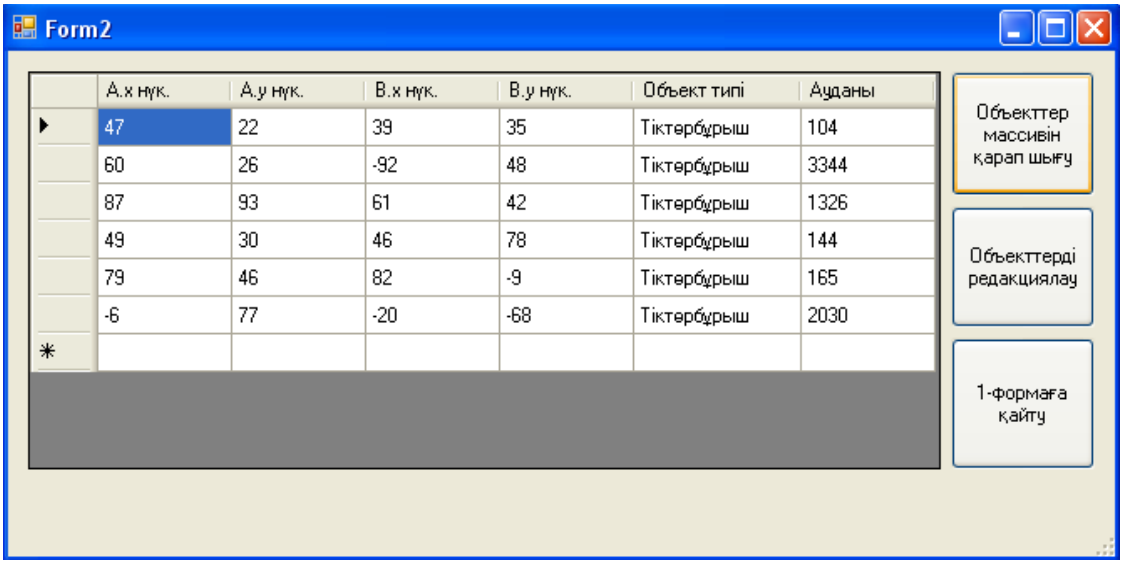
Hide() әдісін Show() әдісімен ашылған, модальді емес формаға да қолдануға болады. Hide() әдісі арқылы диалогтық емес терезені уақытша жасыруға, ал одан кейін Show() әдісін шақырып терезені ашуға болады.

Форманы жабу бойынша ескерту. Негізі форма жабылған кезде ашылған барлық формалар жабылады және қосымша өз жұмысын аяқтайды. Ал кез келген басқа форма жабылған кезде, осы форма ғана жабылады, ал басқа формалар ашық күйінде қалады..

6.4 Мәндерді кесте түрінде көрсету және редакциялау

Тіктөртбұрыш төбелері координаттарының мәндерін кесте түрінде көрсетуге және редакциялауға арналған терезе оқиғаларының өңдеуіштерін қарастырайық.

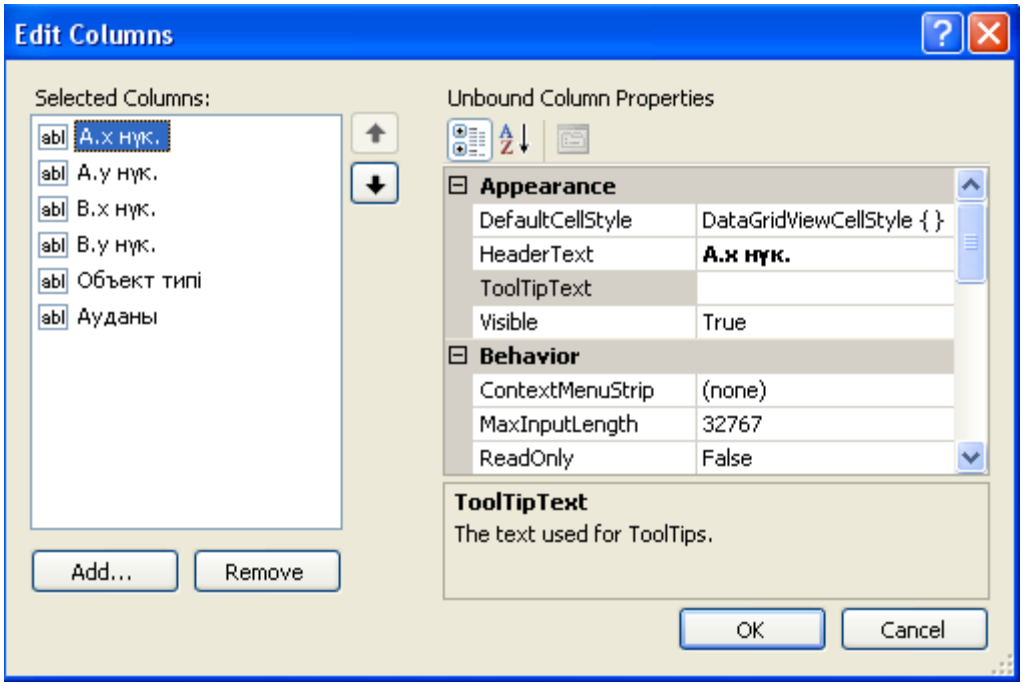
Өңдеуіштер 2-формада жүзеге асырылады (6.6-сурет). Массив мәндерін кесте түрінде көрсету үшін DataGridView элементі қолданылады. Оның қолданылуын қарастырайық, осы элемент пайдаланушыға екіөлшемді массивті енгізуге, көрсетуге мүмкіндік береді.



6.6-сурет – Массив мәндерін кесте түрінде көрсету

DataGridView элементінің қасиеттерінде Columns-ты таңдаймыз, бағаналар параметрінің редакторын іске қосамыз. Редактордың көмегімен біз

керекті бағаналар санын қосамыз, бағдарламада қолданылатын бағана атауын және оның формада көрсетілетін атауын анықтаймыз (6.7-сурет).



6.7-сурет – DataGridView бағаналар редакторымен жұмыс жасау

2-форманың коды :

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
namespace WindowsFormsApplication1
{
 public partial class Form2 : Form
 {
 public static int i, j;
 public static string kop;
 public Form2()
 {
 InitializeComponent();
 }
 private void button1_Click(object sender, EventArgs e)
 {
 for (i = 0; i < 6; i++)
 {
 dataGridView1.Rows.Add();
 for (j = 0; j < 6; j++)
 {
dataGridView1.Rows[i].Cells[j].Value=Form1.a[i,j].ToString()
;
 }
```

```
      if (Form1.a[i,4]==0) dataGridView1.Rows[i].Cells[4].Value =
"Тіктөрбұрыш";
      else dataGridView1.Rows[i].Cells[4].Value = "Квадрат";
      }
      }
      private void button2_Click(object sender, EventArgs e)
      {
      Close();
      }
      private void button3_Click(object sender, EventArgs e)
      {
      string elem = "";
      bool ok;
      int k;
      for (i = 0; i < 6; i++)
      for (j = 0; j < 4; j++)
      {
      do
      {
      ok = true;
      try
      {
      elem = dataGridView1.Rows[i].Cells[j].Value.ToString();
      Form1.a[i, j] = int.Parse(elem);
      }
      catch (Exception any)
      {
      Form5 f5 = new Form5();
      if (f5.ShowDialog() == DialogResult.OK) k = 0;
      dataGridView1.Rows[i].Cells[j].Value = kop;
      ok = false;
      }
      } while (!ok);
      }
      for (i = 0; i < 6; i++)
      {
      if          (Math.Abs(Form1.a[i,0]-Form1.a[i,2])          ==
Math.Abs(Form1.a[i, 1] - Form1.a[i, 3])) Form1.a[i, 4] = 1;
       else Form1.a[i, 4] = 0;
      Form1.a[i, 5] = Math.Abs(Form1.a[i, 0] - Form1.a[i, 2]) *
Math.Abs(Form1.a[i, 1] - Form1.a[i, 3]);
      }
      }
      }
     }
```

2-формада 3 оқиға өңдеуіштері орындалған – DataGridView элементінің кестесіне матрицаны шығару, матрица мәндерін редакциялау және 1-формаға қайту. Кодтың әрбір үзіндісін жеке қарастырайық.

DataGridView элементінің кестесіне матрицаны шығару үшін қолданылатын циклда әрбір жаңа жол бағдарламалық түрде қосылады және онда кесте бағаналарын жекелеп қарап шығу циклі іске қосылады.

Массив 1-формада ауқымды айнымалы түрінде құрылғандықтан оны 2-формада пайдалану үшін Form1.a[i,j] жазуы қолданылады.

Екіөлшемді массивтің 4-ші бағанасында 0-ге тең мәннің болуы тексеріледі – тіктөртбұрыш қабырғалары теңдігінің сипаты (квадрат).

Соңғы бағанаға тіктөртбұрыштың ауданы шығады.

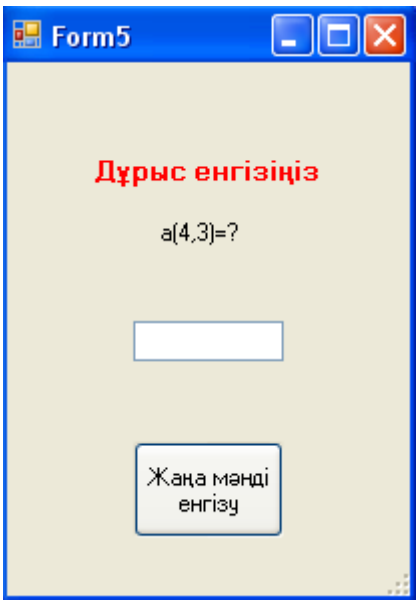
DataGridView элементін редакциялау процессінде 2-формада мәндер тікелей өзгертіледі және «Объекттерді редакциялау» өңдеуіші іске қосылады. Өңдеуіште DataGridView элементінің мәндері 1-форманың массивіне жазылады. Ол қорғалатын блокта орналасады. Егер Сіз сан орнына әріп немесе басқа символды енгізетін болсаңыз, қате орын алған жағдайдың өңдеуіші іске қосылады. Ол мәнді қайта теру үшін 5-ші диалогтық терезе шығарылады (6.8-сурет).

5-форманың коды:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Drawing;
using System.Text;
using System.Windows.Forms;
namespace WindowsFormsApplication1
{
 public partial class Form5 : Form
 {
 public Form5()
 {
 InitializeComponent();
 label1.Text = "a[" + Form2.i.ToString() + "," + 
Form2.j.ToString() + "]= ?";
 }
 private void button1_Click(object sender, EventArgs e)
 {
 Form2.kop = textBox1.Text;
 Close();
 }
 }
}
```

Қатені түзету дұрыс сандық мән енгізілгенше қайталанады.

Егер редакциялау нәтижесінде тіктөртбұрыштың типі өзгерсе, мысалы, квадрат болса, онда ол 4-бағанада ескеріледі, сонымен қатар тіктөртбұрыштың ауданы қайтадан есептеліп, шығарылады.



6.8-сурет – Қатені түзету

2-форманың кодында DataGridView элементіне кезекті жаңа жол бағдарламалық түрде қосылған. DataGridView элементіне бағдарламалық түрде бағаналарды да құруға болады. Мысалы, егер DataGridView элементіне бағдарламалық түрде 6 бағананы қосу керек болса, онда кодтың келесі үзіндісін қолдануға болады:

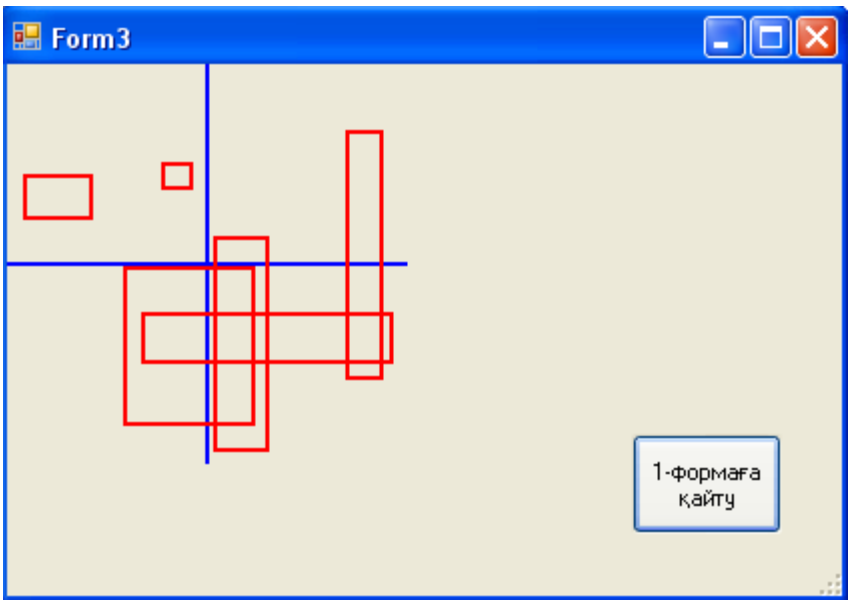
```
dataGridView1.Columns.Clear();
DataGridViewColumn column;
for (int i = 0; i < 6; i++)
{
column = new DataGridViewTextBoxColumn();
column.DataPropertyName = "Column" + i.ToString();
column.Name = "Column" + i.ToString();
dataGridView1.Columns.Add(column);
}
```

Кодтың бірінші жолында DataGridView элементінің бағаналарын жоямыз, column айнымалысын жариялаймыз, одан кейін циклде column объектісін құраймыз. Объект үшін бағдарламада қолданылатын және формаға шығатын атаулар анықталады, осыдан кейін құрылған объект элементке қосылады.

6.5 Тіктөртбұрыштарды графикалық формада көрсету

Тіктөртбұрыштарды шығаратын графикалық форма терезесіндегі оқиғалар өңдеуіштерін қарастырамыз.

Өңдеуіштер 3-формада жүзеге асырылады (6.9-сурет).



6.9-сурет – Тіктөртбұрыштарды формада көрсету

Ескерту, көрсетілген форманың дизайны жоғарғы дәрежеде емес. 3-форманың кодында ешқандай ерекшеліктер жоқ. Төменде форманың коды көрсетілген:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace WindowsFormsApplication1
{
 public partial class Form3 : Form
 {
 public Form3()
 {
 InitializeComponent();
 }
 private void button1_Click(object sender, EventArgs e)
 {
 Close();
 }
 private void Form3_Paint(object sender, PaintEventArgs e)
 {
 int ax, ay, bx, by;
 Pen myPen = new Pen(Color.Blue, 2);
 Graphics g = e.Graphics;
 g.DrawLine(myPen, 0, 100, 60, 100);
 g.DrawLine(myPen, 100, 0, 100, 60);
 myPen = new Pen(Color.Red, 2);
 for (int i = 0; i < 6; i++)
```

```
 {
 if (Form1.a[i, 0] < Form1.a[i, 2]) ax = Form1.a[i, 0];
  else ax = Form1.a[i, 2];
 if (Form1.a[i, 1] < Form1.a[i, 3]) ay = Form1.a[i, 1];
  else ay = Form1.a[i, 3];
 bx = Math.Abs(Form1.a[i, 0] - Form1.a[i, 2]);
 by = Math.Abs(Form1.a[i, 1] - Form1.a[i, 3]);
 g.DrawRectangle(myPen, ax+100, ay+100, bx, by);
 }
 }
 }
}
```

Жобаға «Автор туралы ақпарат» формасы да қосылған.

6.10-сурет – «Автор туралы ақпарат» режімінің терезесі

4-форманың кодында 1-формаға көшетін батырманың өңдеуіші ғана жазылған. Қалған әрекеттер 4-форма элементтерінің қасиеттері арқылы қосылған.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
namespace WindowsFormsApplication1
{
 public partial class Form4 : Form
 {
 public Form4()
```

```
 {
  InitializeComponent();
 }

 private void button1_Click(object sender, EventArgs e)
 {
  Close();
 }
 }
}
```

6.6 Өзін-өзі тексеру сұрақтары

1 MDI қысқартуы нені білдіреді?

2 Қандай жағдайларда батырмасы бар форманы жобалау керек?

3 Формада суреттерді орналастыру үшін әдетте қандай басқару элементі қолданылады?

4 Solution Explorer терезесі арқылы жобаға жаңа форманы қалай қосуға болады?

5 Project режімі арқылы жобаға жаңа форманы қалай қосуға болады?

6 Диалогтық (модальді) терезенің форманың қарапайым терезесінен айырмашылығы қандай?

7 Форманың қарапайым (модальді емес) терезесі қандай әдіс арқылы ашылады?

8 Форманың модальді терезесі қандай әдіс арқылы ашылады?

9 Бағдарламаның келесі үзіндісі нені орындайды:

```
private void button3_Click(object sender, EventArgs e)
 {
        Form5 f5 = new Form5();
        if (f5.ShowDialog() == DialogResult.OK) k = 0;
 } ?
```

10 Ақпаратты кестеде шығару үшін қандай басқару элементі жиі қолданылады?