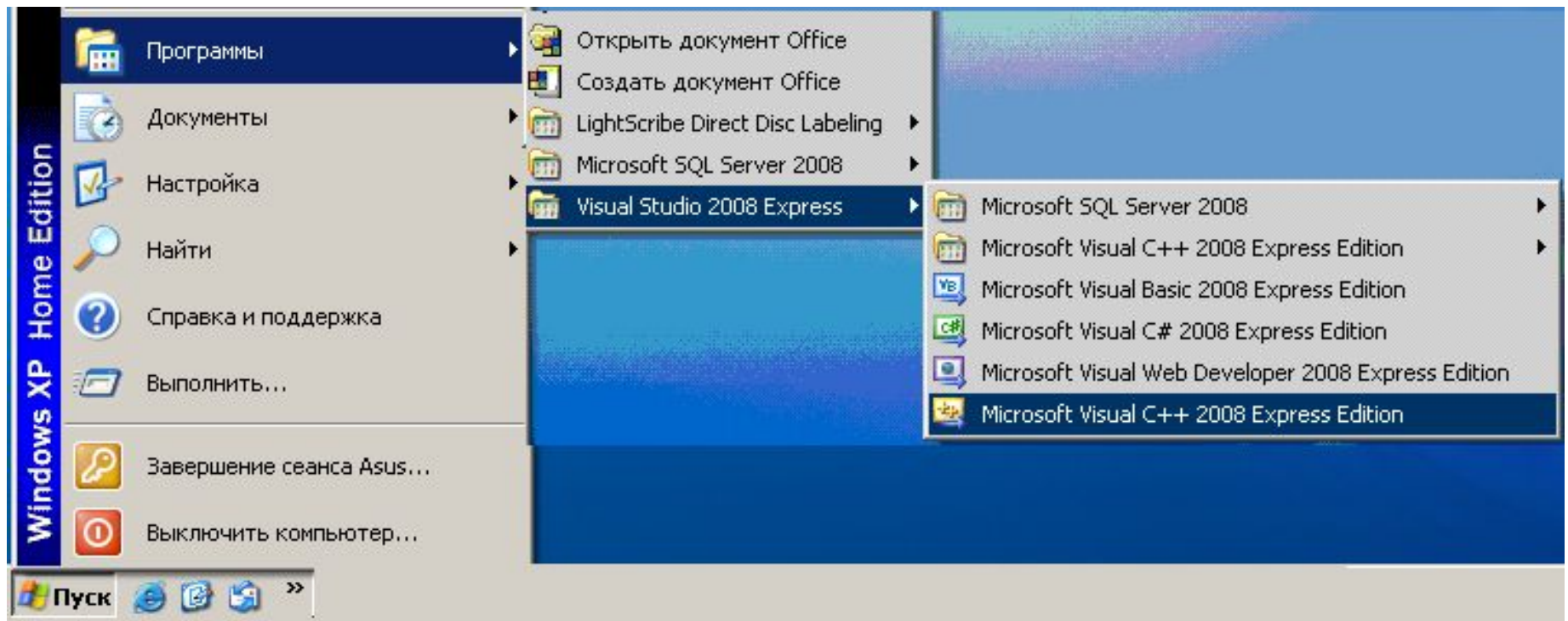


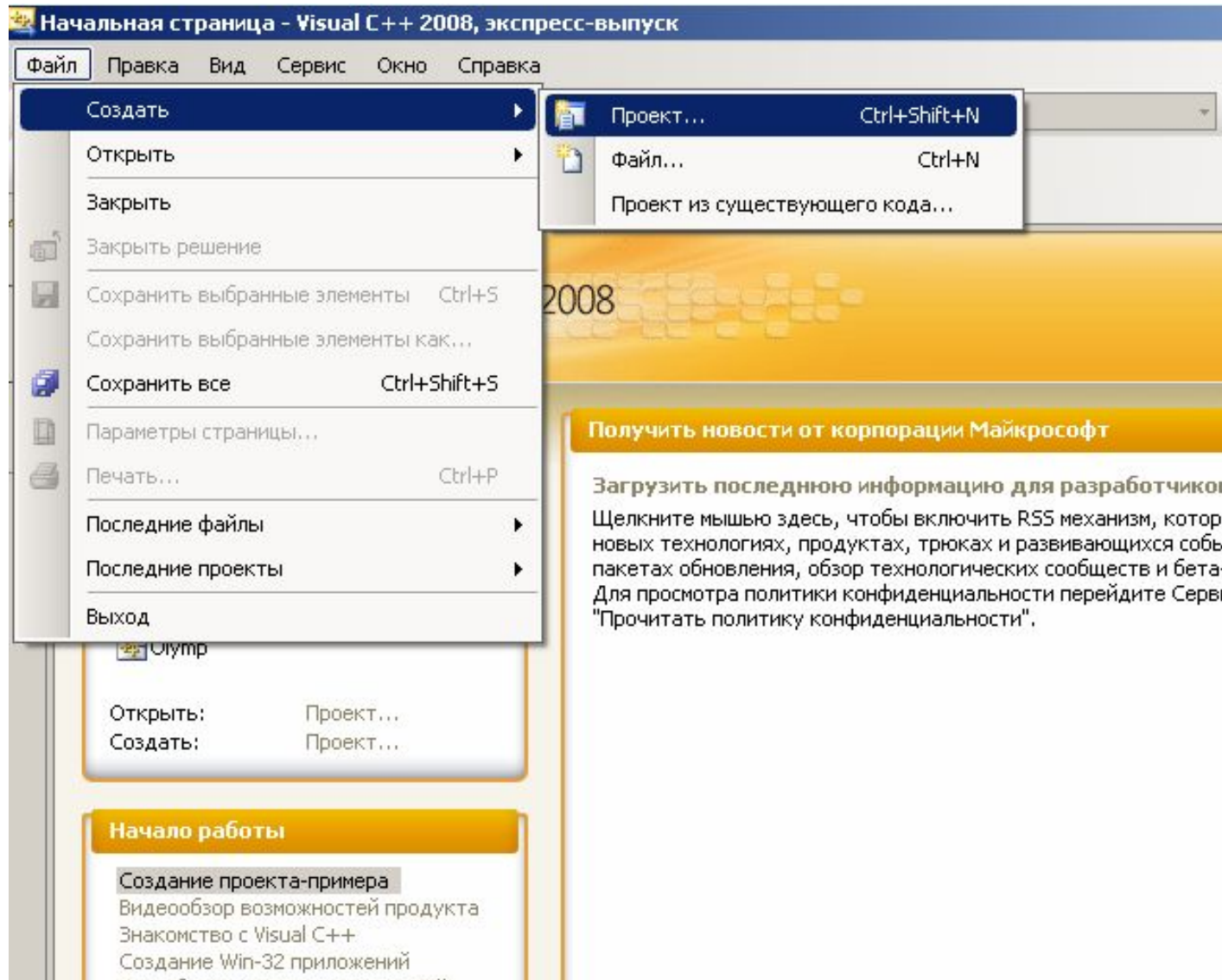
Курсовая работа  
в среде  
Microsoft Visual Studio 2008  
Express

# 1. Запускаем среду

## Microsoft Visual C++ Express Edition



## 2. Создаем новый проект



3. Задаем тип (Приложение Windows Forms), название (Graph1) и каталог для размещения проекта. Нажимаем <Ok>

**Создать проект**

Типы проектов:

- Visual C++
  - CLR
  - Win32
  - Общие

Шаблоны:

**Установленные шаблоны Visual Studio**

- Библиотека классов
- Пустой проект CLR
- Консольное приложение CLR
- Приложение Windows Forms

**Мои шаблоны**

- Найти шаблоны в Интернете...

Проект приложения с пользовательским интерфейсом Windows

Имя: Kursovoj

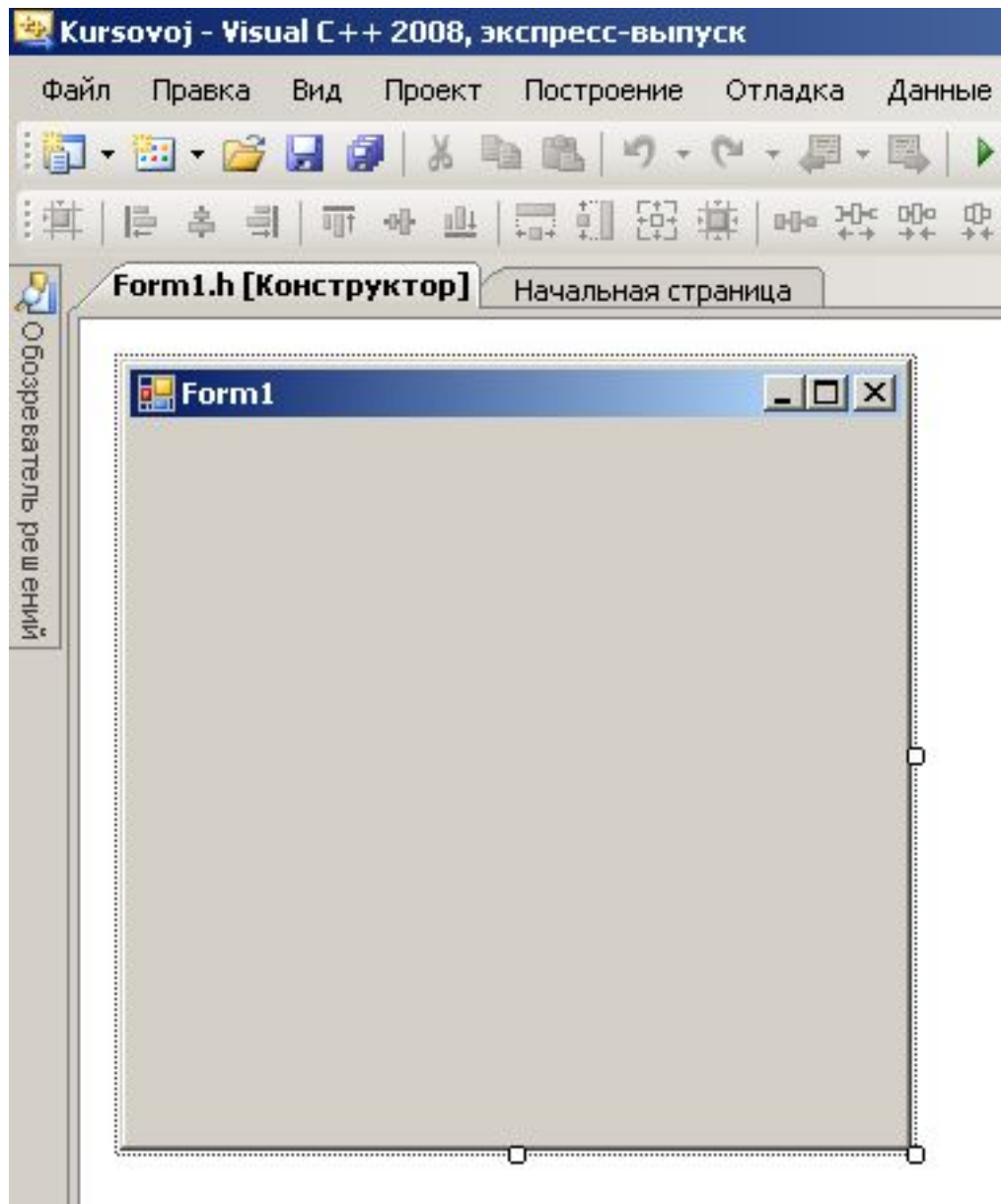
Расположение: D:\Proba\2011 Обзор...

Решение: Создать новое решение  Создать каталог для решения

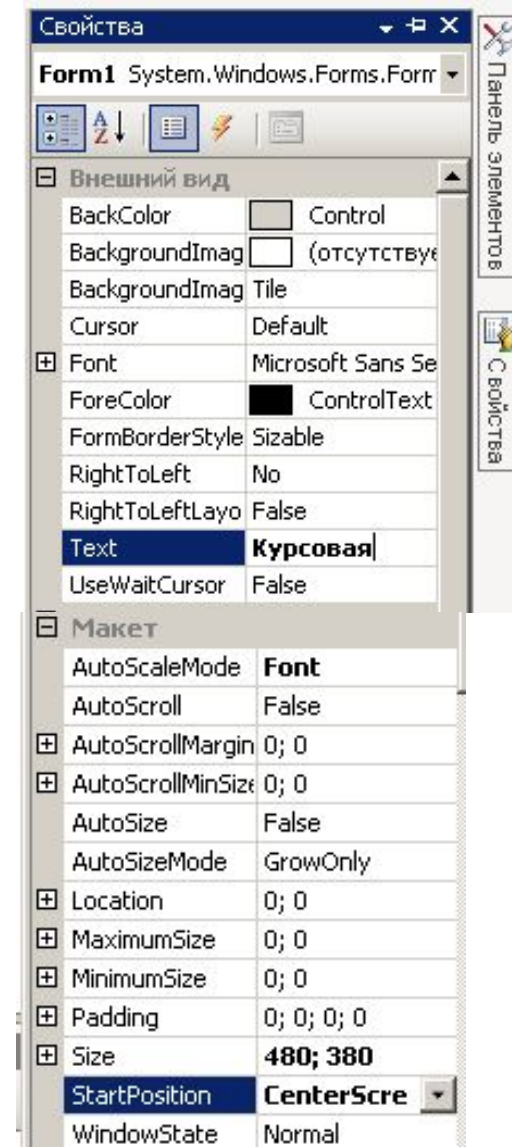
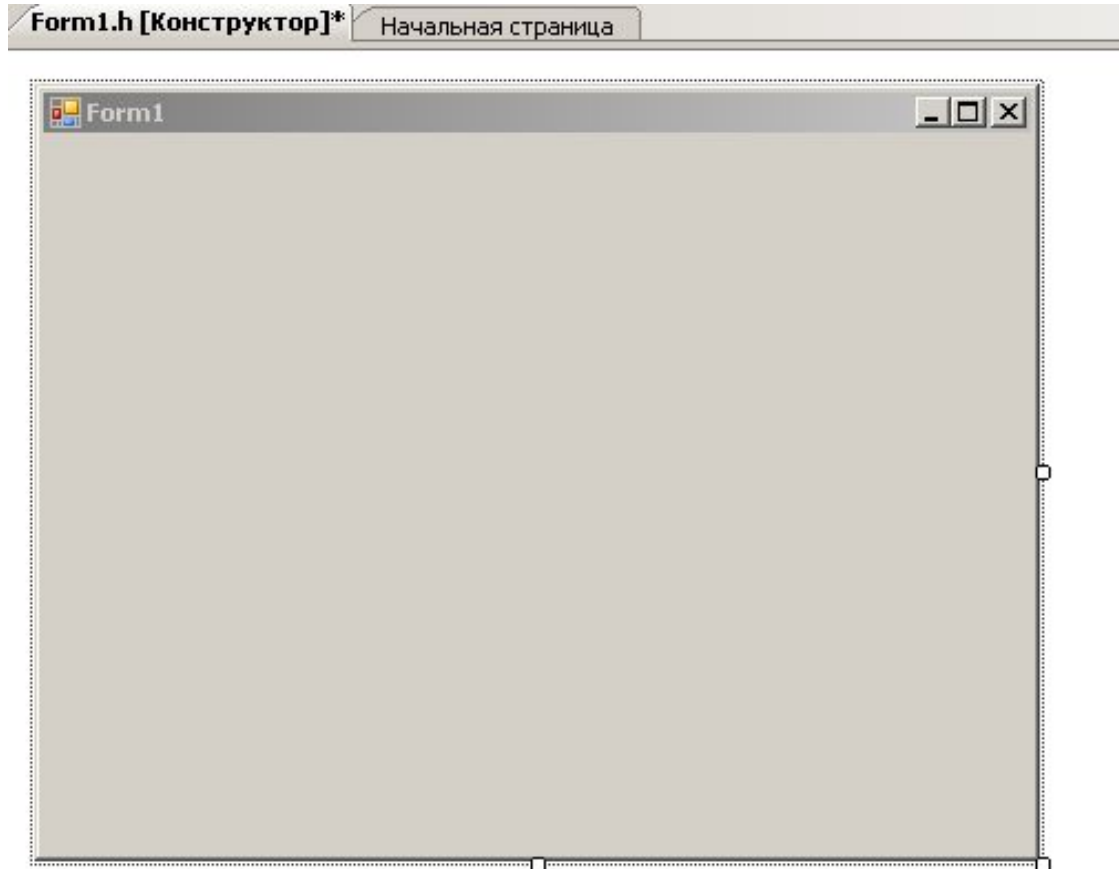
Имя решения: Kursovoj

OK Отмена

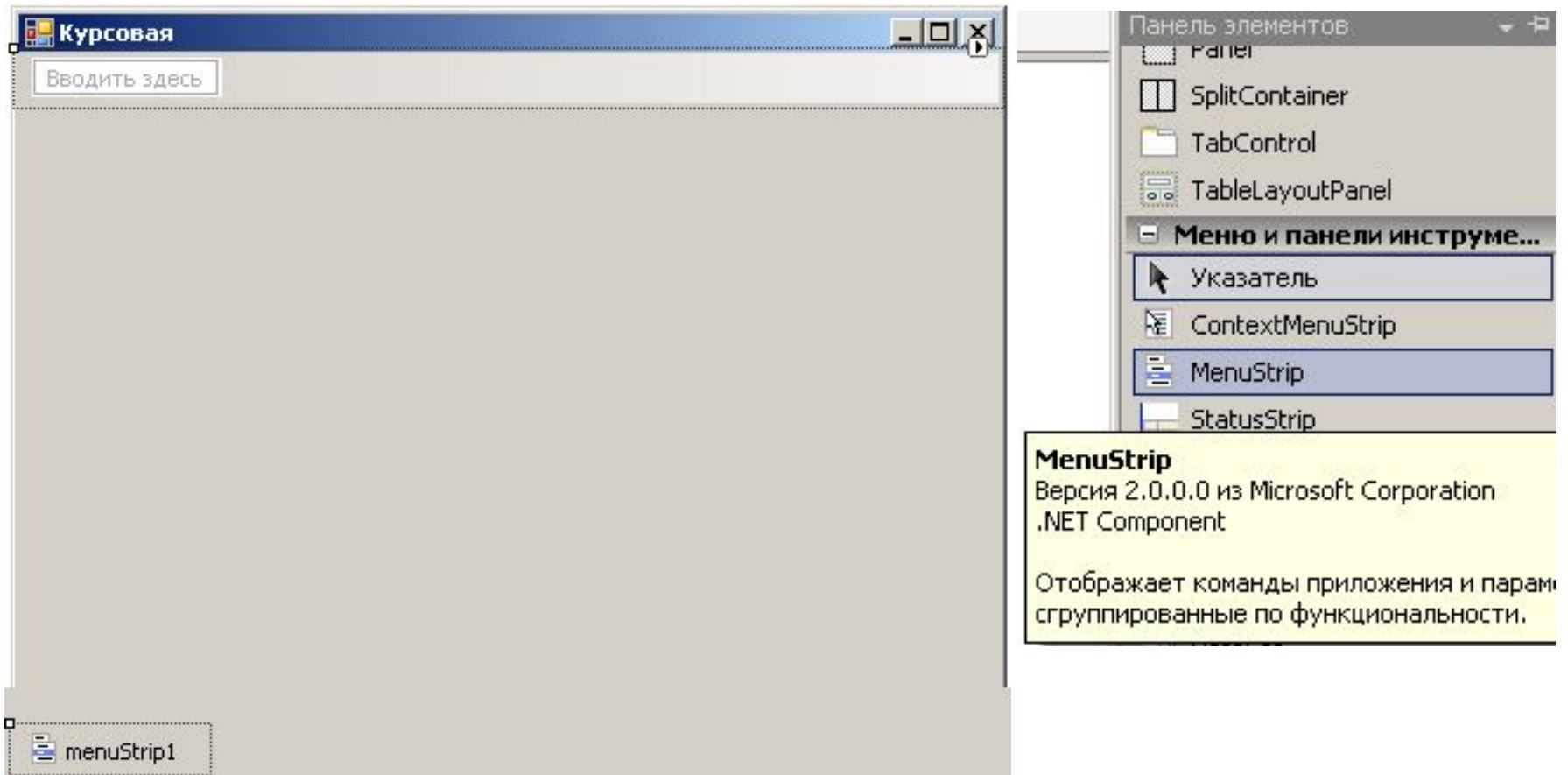
## 4. Мастер создает заготовку формы



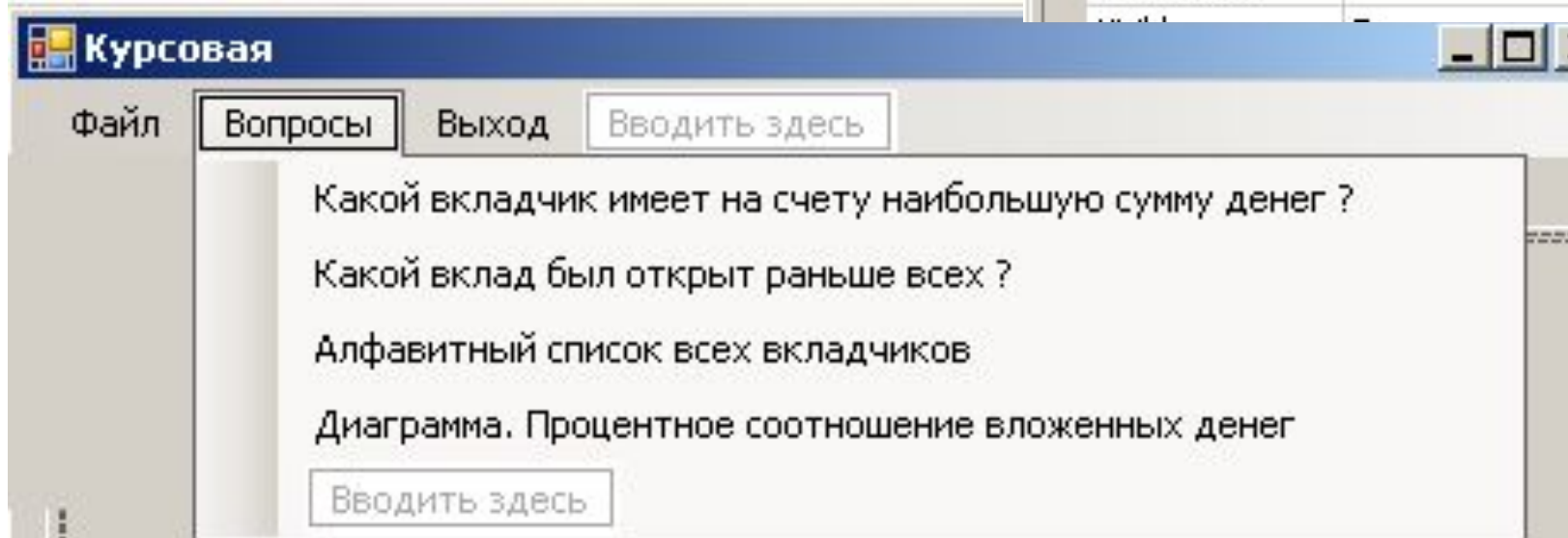
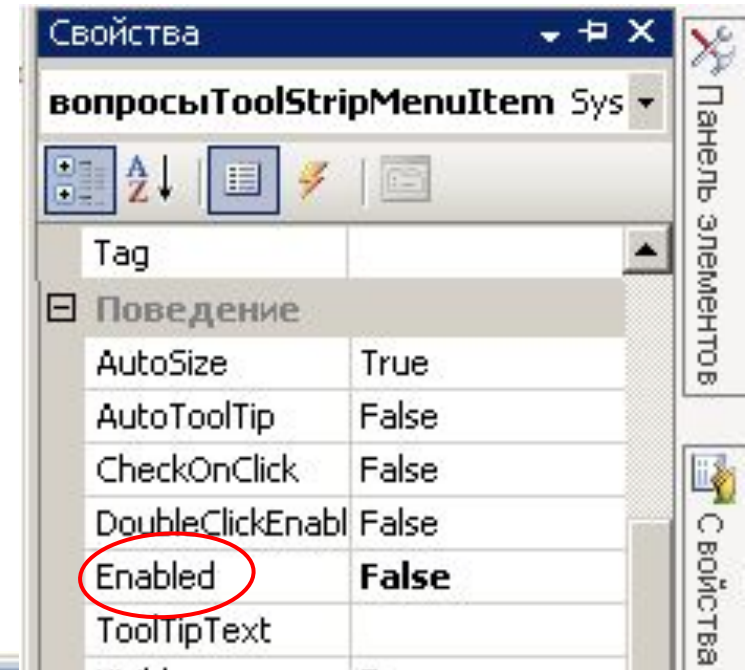
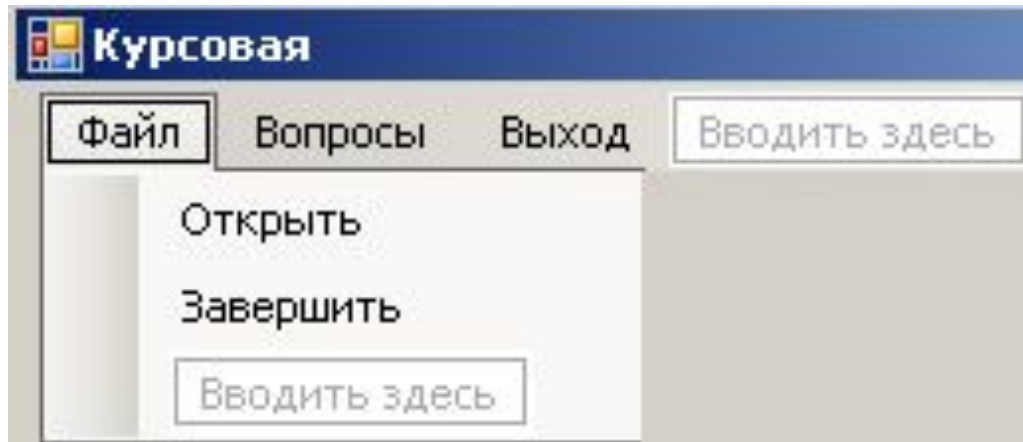
## 5. Устанавливаем размер и свойства формы



## 6. Перетаскиваем на форму элемент MenuStrip

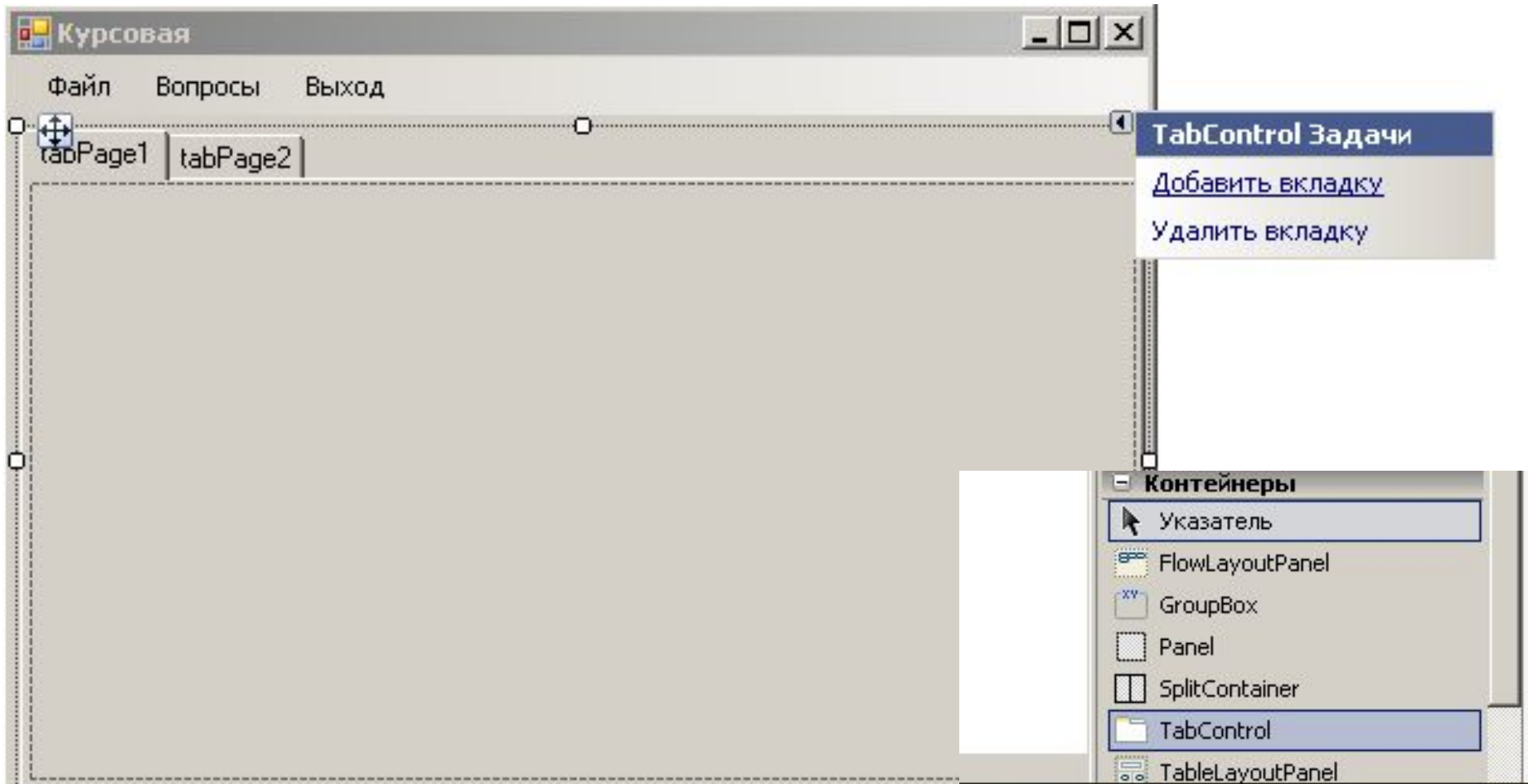


## 7. Наполняем меню нужными пунктами





## 7. Добавляем элемент TabControl



Это один из способов  
добавления / удаления  
вкладок

### TabControl

Версия 2.0.0.0 из Microsoft Corporation  
.NET Component

Обработывает и отображает для пользователя соответствующую коллекцию вкладок, содержащих элементы управления и компоненты.

## 8. Редактируем вкладки TabControl

The image shows two windows from the Visual Studio IDE. The left window is titled "Редактор коллекции tabPage" (TabPage Collection Editor). It contains a list of members: tabPage1, tabPage2, and tabPage3. tabPage3 is selected. Below the list are "Добавить" (Add) and "Удалить" (Remove) buttons. The right window is titled "Свойства" (Properties) and shows the properties for "tabControl1". The "Поведение" (Behavior) section is expanded, showing the "TabPage" property set to "1" and the "TabPage" collection property set to "(Коллекция)".

**Редактор коллекции tabPage**

Члены:

0	tabPage1
1	tabPage2
2	tabPage3

Добавить    Удалить

**Свойства tabPage3:**

**Внешний вид**

BackColor	Control
BackgroundImage	(отсутствует)
BackgroundImageTile	
BorderStyle	None
Cursor	Default
Font	Microsoft Sans Serif;
ForeColor	ControlText
RightToLeft	No
<b>Text</b>	<b>Диаграмма</b>
UseVisualStyleBack	True
UseWaitCursor	False

**Данные**

OK    Отмена

**Свойства**

tabControl1 System.Windows.Form

**Поведение**

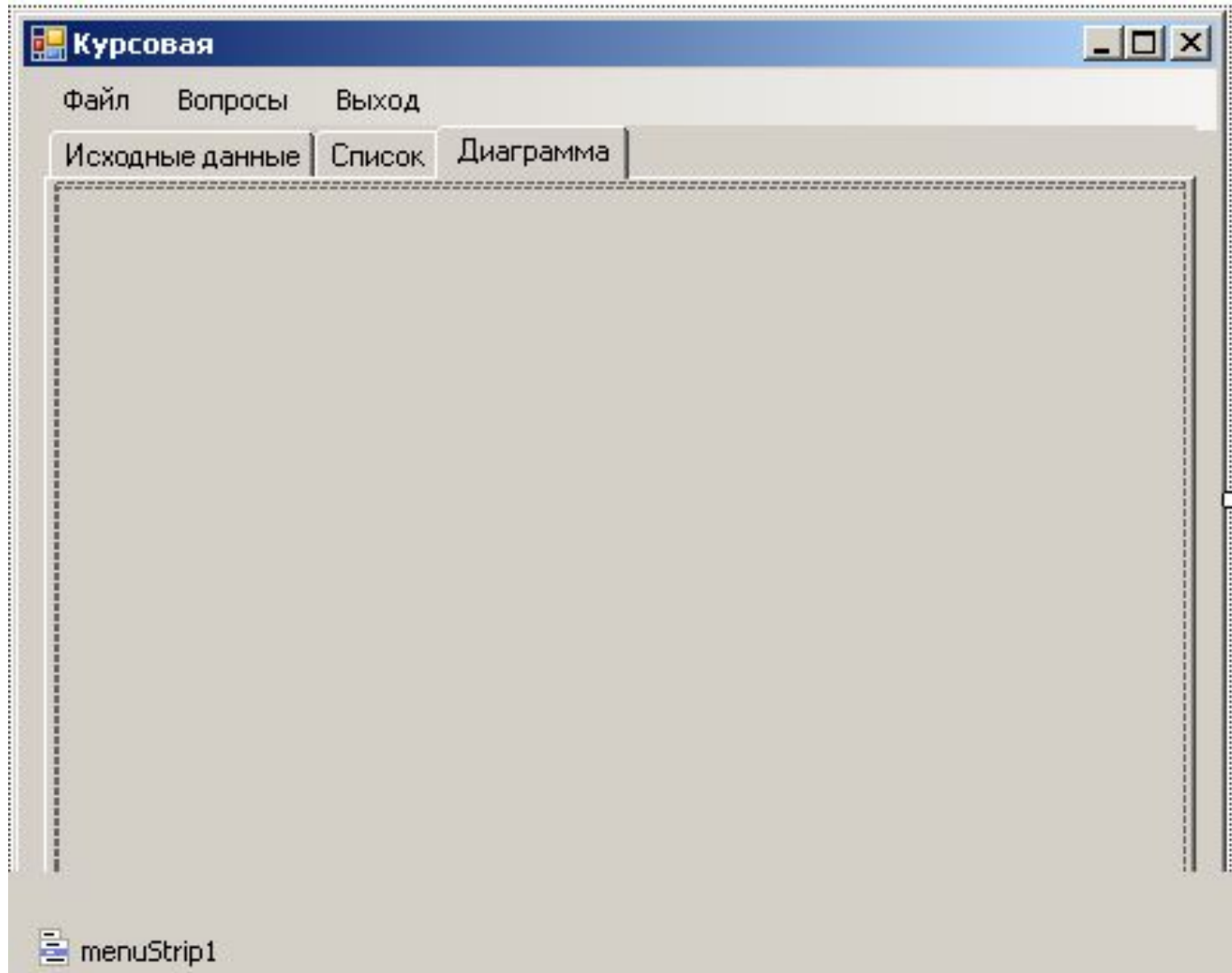
Location	4; 30
Margin	3; 3; 3; 3
MaximumSize	0; 0
MinimumSize	0; 0
Size	467; 274
Alignment	Top
AllowDrop	False
Appearance	Normal
ContextMenuStr	(нет)
DrawMode	Normal
Enabled	True
HotTrack	False
ImeMode	NoControl
ItemSize	58; 18
Multiline	False
Padding	6; 3
ShowToolTips	False
SizeMode	Normal
TabIndex	1
<b>TabPage</b>	<b>1</b>
<b>TabPage</b>	<b>(Коллекция) ...</b>
TabStop	True
Visible	True

**Проектирование**

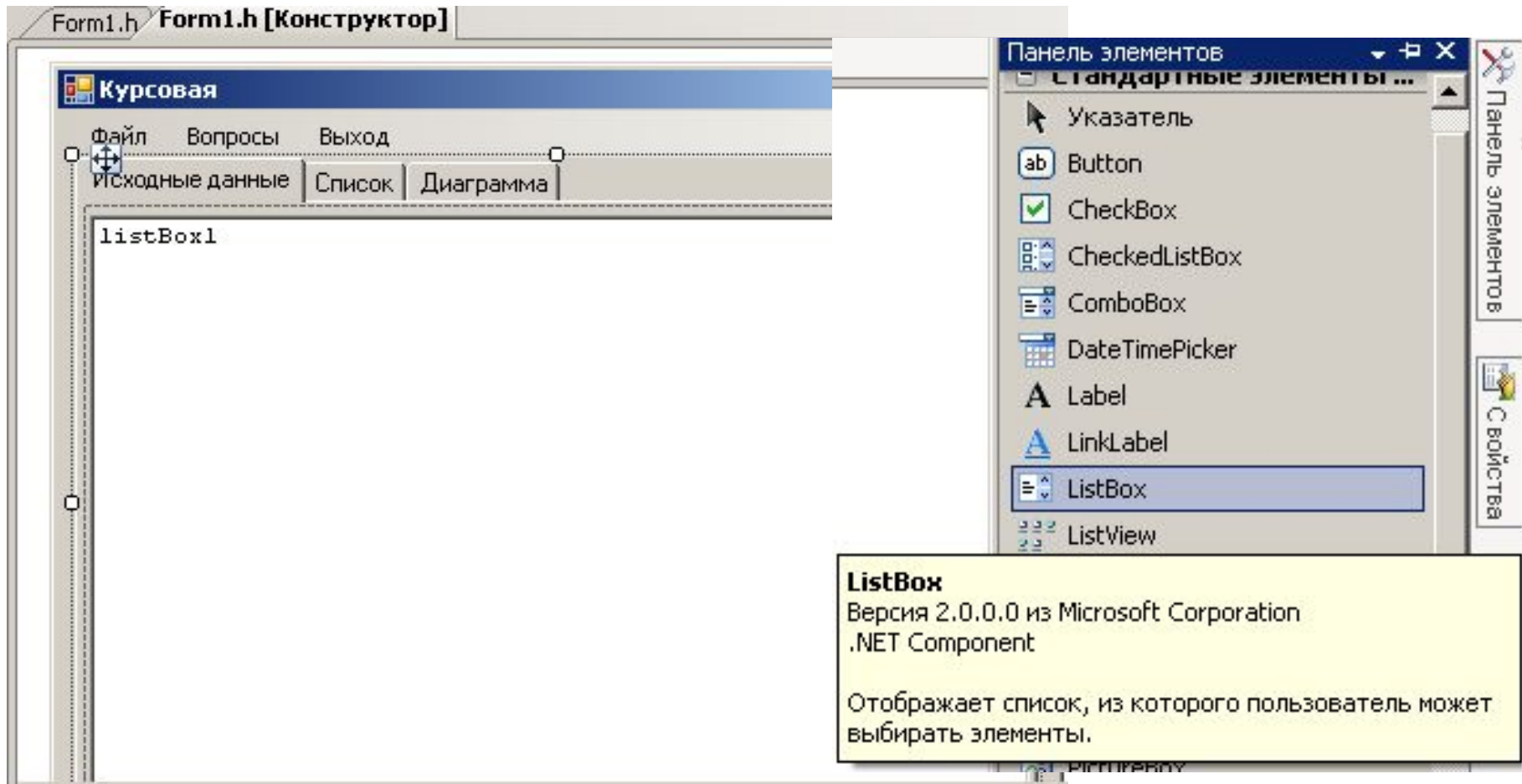
(Name)	tabControl1
--------	-------------

Гораздо больше возможностей обеспечивает свойство TabPages (Коллекция) ...

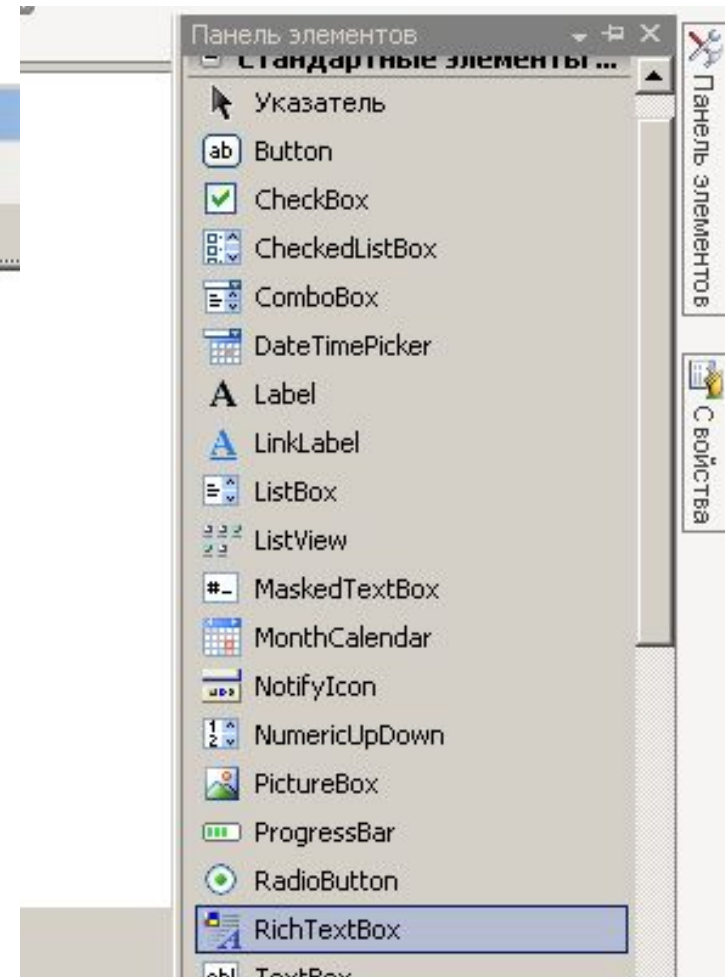
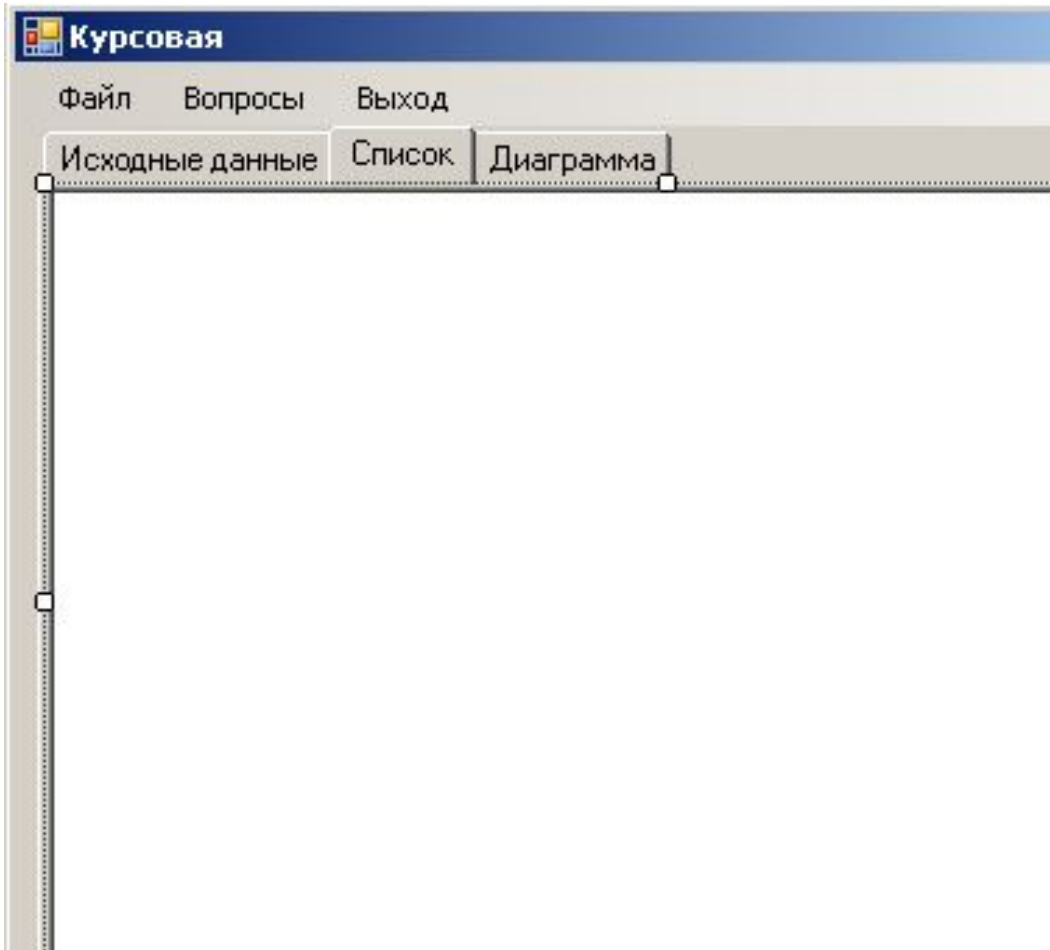
## 9. В результате форма приобретает вид



10. На вкладке «Исходные данные» размещаем элемент ListBox, который получит название **listBox1**



11. На вкладке «Список» размещаем элемент RichTextBox, который получит название **richTextBox1**

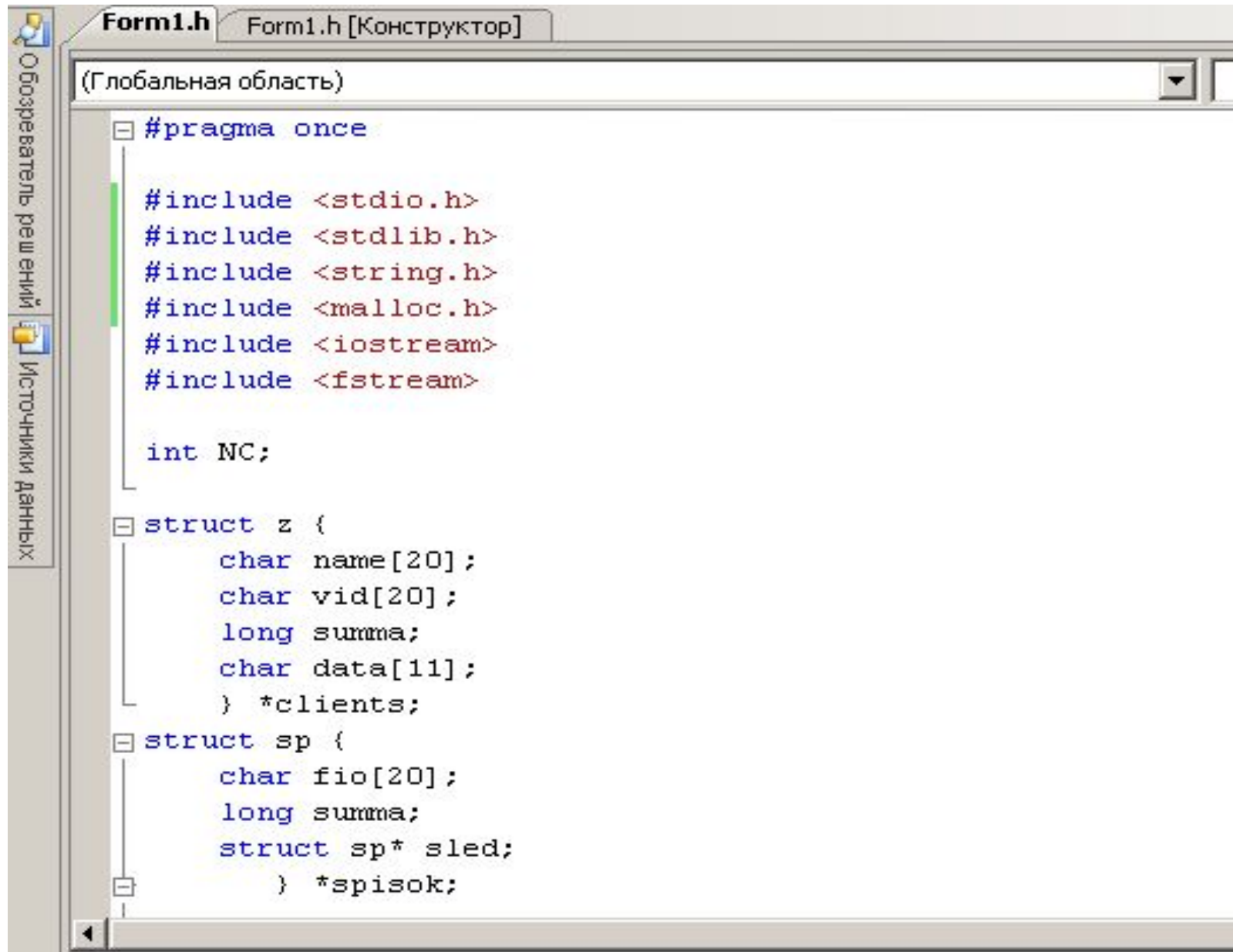


#### **RichTextBox**

Версия 2.0.0.0 из Microsoft Corporation  
.NET Component

Обеспечивает дополнительные возможности ввода и редактирования текста (например, форматирование символов и абзацев).

## 12. Приступаем к созданию программного кода



The screenshot shows a code editor window titled "Form1.h" with a sub-tab "Form1.h [Конструктор]". The editor content is labeled "(Глобальная область)". The code is as follows:

```
#pragma once

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <malloc.h>
#include <iostream>
#include <fstream>

int NC;

struct z {
    char name[20];
    char vid[20];
    long summa;
    char data[11];
} *clients;

struct sp {
    char fio[20];
    long summa;
    struct sp* sled;
} *spisok;
```

```
void vstavka(struct z* client,char* fio)
{
int i;
struct sp *nov,*nt,*z=0;
for(nt=spisok; nt!=0 && strcmp(nt->fio,fio)<0; z=nt, nt=nt->sled);
if(nt && strcmp(nt->fio,fio)==0) return;

nov=(struct sp *) malloc(sizeof(struct sp));
strcpy(nov->fio,fio);
nov->sled=nt;

nov->summa=0;
for(i=0;i<NC;i++)
    if(strcmp(client[i].name,fio)==0)
        nov->summa+=client[i].summa;

if(!z) spisok=nov;
else z->sled=nov;

return;
}
```

```
void text_data(char *s,char *sd)
{
int N_month;
char s0[3],month[12][9]={
"января","февраля","марта","апреля","мая","июня",
"июля","августа","сентября","октября","ноября","декабря"
};
strcpy(s,sd+8);
strcat(s," ");
strncpy(s0,sd+5,2);
s0[2]=0;
sscanf(s0,"%d",&N_month);
strcat(s,month[N_month-1]);
strcat(s," ");
strncat(s,sd,4);
return;
}
```



## 13. Программируем пункты меню.

Выход.

Завершить.

```
private: System::Void выходToolStripMenuItem_Click
```

```
(  
    System::Object^ sender,  
    System::EventArgs^ e  
)
```

```
{  
    this->Close();  
}
```

```
private: System::Void завершитьToolStripMenuItem_Click
```

```
(  
    System::Object^ sender,  
    System::EventArgs^ e  
)
```

```
{  
    this->Close();  
}
```

## 14. Программируем пункты меню. Открыть.

```
private: System::Void открытьToolStripMenuItem_Click
(
    System::Object^ sender,
    System::EventArgs^ e
)
{
    FILE *in;
    int i;
    char ctemp[80];
    String ^s;
    if ((in=fopen("Vklad.dat","r"))==NULL)
    {
        MessageBox::Show("Файл не открыт!",
            "Ошибка!",
            MessageBoxButtons::OK,
            MessageBoxIcon::Error);
        return;
    }
}
```

вопросы ToolStripMenuItem->Enabled=true;

```

        &clients[i].summa, clients[i].data);
    sprintf(ctemp,"%-20s %-20s %7ld %s",
            clients[i].name,clients[i].vid,
            clients[i].summa,clients[i].data);
listBox1->Items->Clear(),
    s=gcnew String(ctemp);
    fscanf(in,"%d",&NC);
listBox1->Items->Add(s);
    clients = new z[NC];
    for(i=0;i<NC;i++)
    fclose(in);
    }
    fscanf(in,"%s%s%ld%s",clients[i].name, clients[i].vid,
            &clients[i].summa, clients[i].data);
    sprintf(ctemp,"%-20s %-20s %7ld %s",
            clients[i].name,clients[i].vid,
            clients[i].summa,clients[i].data);

    s=gcnew String(ctemp);
listBox1->Items->Add(s);
    }
    fclose(in);
    }

```

## 15. Програмуємо пункти меню. Вопрос 1.

```
private: System::Void
какойВкладчикИмеетНаСчетуНаибольшуюСуммуДенегToolStripMenuItem_Click
(System::Object^ sender, System::EventArgs^ e)
{
int i=0;
struct z best;
char ss[80];
String ^s;

strcpy(best.name,clients[0].name);
best.summa=clients[0].summa;
for(i=1;i<NC;i++)
    if (clients[i].summa>best.summa)
        {
            strcpy(best.name,clients[i].name);
            best.summa=clients[i].summa;
        }
sprintf(ss,"%s\n %ld рублей",best.name,best.summa);
s=gcnew String(ss);
MessageBox::Show(s,"Наибольший вклад");
}
```

## 16. Програмуємо пункти меню. Вопрос 2.

```
private: System::Void какойВкладБылОткрытРаньшеВсехToolStripMenuItem_Click
        (System::Object^ sender, System::EventArgs^ e)
{
    int i;
    char sd[17];
    char ss[80];
    String ^s;
    struct z* best=clients;
    for(i=1;i<NC;i++)
        if (strcmp(clients[i].data,best->data)<0) best=&clients[i];
    text_data(sd,best->data);

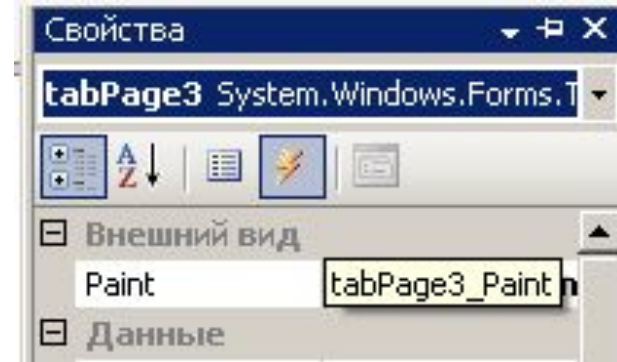
    sprintf(ss,"%s\n%s\n%ld рублей\nоткрыт %s",
            best->vid,best->name, best->summa,sd);
    s=gcnew String(ss);
    MessageBox::Show(s,"Самый продолжительный вклад");
}
```

## 17. Программируем пункты меню. Список.

```
private: System::Void алфавитныйСписокВсехВкладчиковToolStripMenuItem_Click  
(System::Object^ sender, System::EventArgs^ e)  
  
{  
int i;  
struct sp* nt;  
char ss[80];  
String ^s;  
tabControl1->SelectTab(1); // переходим на вкладку "СПИСОК"  
if(!spisok)  
    for(i=0;i<NC;i++)  
        vstavka(clients,clients[i].name);  
richTextBox1->ReadOnly=1;  
richTextBox1->Clear();  
richTextBox1->Text = "\nАЛФАВИТНЫЙ СПИСОК\n===== \n";  
for(nt=spisok; nt!=0; nt=nt->sled)  
    {  
    sprintf(ss,"%-20s %10ld",nt->fio,nt->summa);  
    s=gcnnew String(ss,0,31);  
richTextBox1->Text = richTextBox1->Text+"\n"+s;  
    }  
}
```

## 18. Программируем построение диаграммы

Метод **Paint** вызывается  
автоматически всегда, когда надо  
перерисовать элемент  
управления



```
private: System::Void tabPage3_Paint
(System::Object^ sender, System::Windows::Forms::PaintEventArgs^ e)
{
    Pen ^myPen= gcnew Pen(System::Drawing::Color::Black,3);
    SolidBrush ^myBrush= gcnew SolidBrush(Color::FromArgb(196,0,0,0));
    System::Drawing::Font^ myFont=gcnew System::Drawing::Font("Arial",8);

    int i,K;
    int iRed,iGreen,iBlue;
    int aStart,aEnd;
    long Sum;
    float xPos,yPos;
    struct sp *nt;

    Graphics ^g=tabPage3->CreateGraphics();
    g->Clear(System::Drawing::Color::White);
```

```
if(!spisok)
    for(i=0;i<NC;i++)
        vstavka(clients,clients[i].name);
K=0; Sum=0;
for(nt=spisok; nt!=0; nt=nt->sled)
    {
    K++;
    Sum+=nt->summa;
    }
g->DrawEllipse(myPen,25,25,185,185);
aEnd=0;

for(nt=spisok,i=0; nt!=0; nt=nt->sled,i++)
    {
    iRed=(((i+1)&4)>0)*255/(i/8+1);
    iGreen=(((i+1)&2)>0)*255/(i/8+1);
    iBlue=(((i+1)&1)>0)*255/(i/8+1);
    aStart=aEnd;
    aEnd+=nt->summa*360/Sum;
    if(i==K-1)aEnd=360;
```



```
myBrush->Color::set(Color::FromArgb(196,iRed,iGreen,iBlue));
g->FillPie(myBrush,25,25,185,185,aStart,aEnd-aStart);
g->FillRectangle(myBrush,300,50+(i-1)*20,20,20);
```

```
myBrush->Color::set(Color::FromArgb(196,0,0,0));
xPos=30+(185-25)/2+(185-25)/1.5*Math::Cos(Math::PI*(aStart+aEnd)/360);
yPos=30+(185-25)/2+(185-25)/1.5*Math::Sin(Math::PI*(aStart+aEnd)/360);
```

```
g->DrawString(gcnew String(nt->fio),myFont,myBrush,320,55+(i-1)*20);
g->DrawString(Convert::ToString(nt->summa*100/Sum)+"%",
               myFont,myBrush,xPos,yPos);
```

```
} //конец цикла for...
```

```
} // конец функции
```

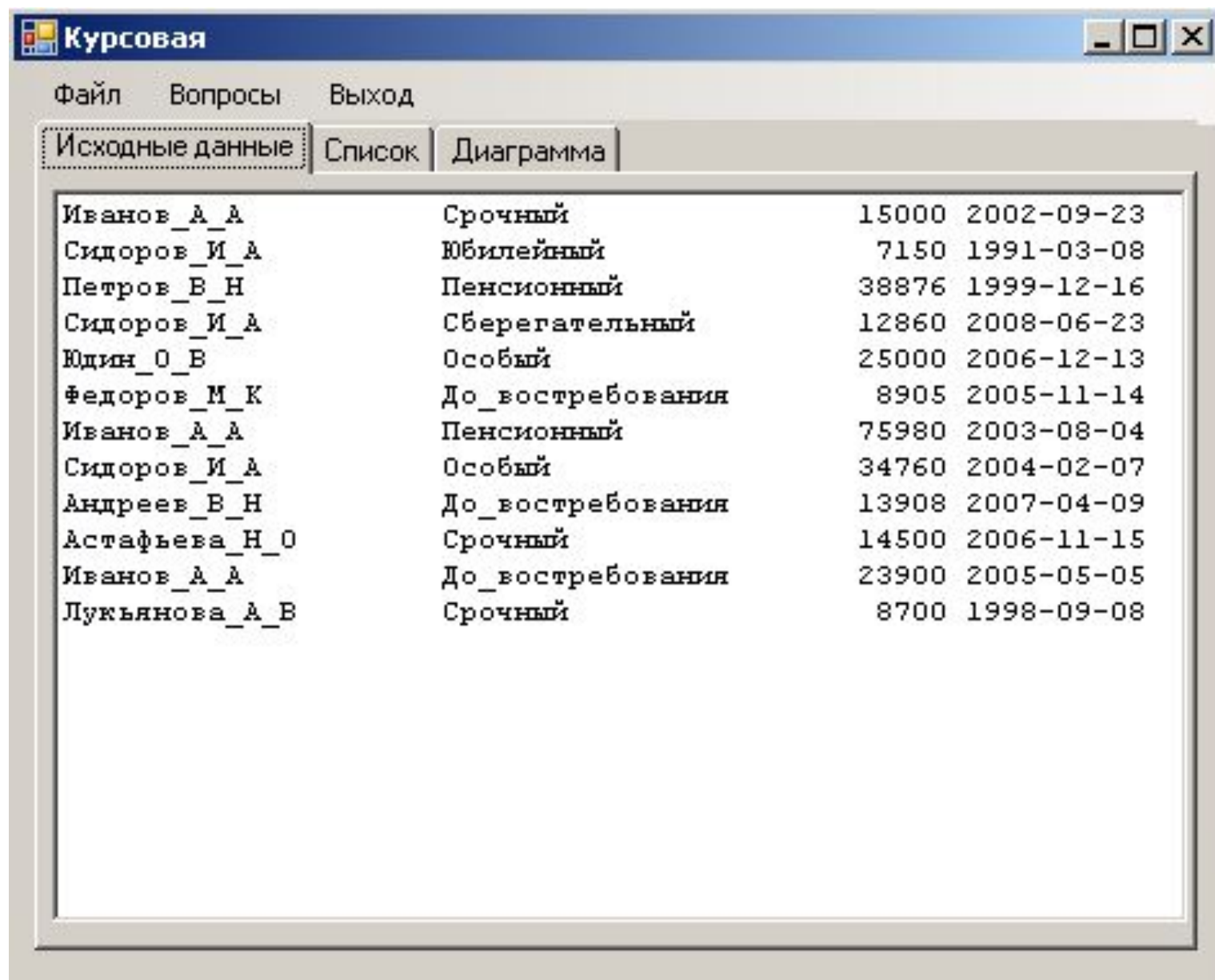
## 19. Программируем пункт меню Диаграмма...

Здесь нужно просто передать фокус вкладке «**Диаграмма**», чтобы автоматически вызвать метод **Paint** для **tabPage3**

И описанный в 18-м пункте метод нарисует диаграмму.

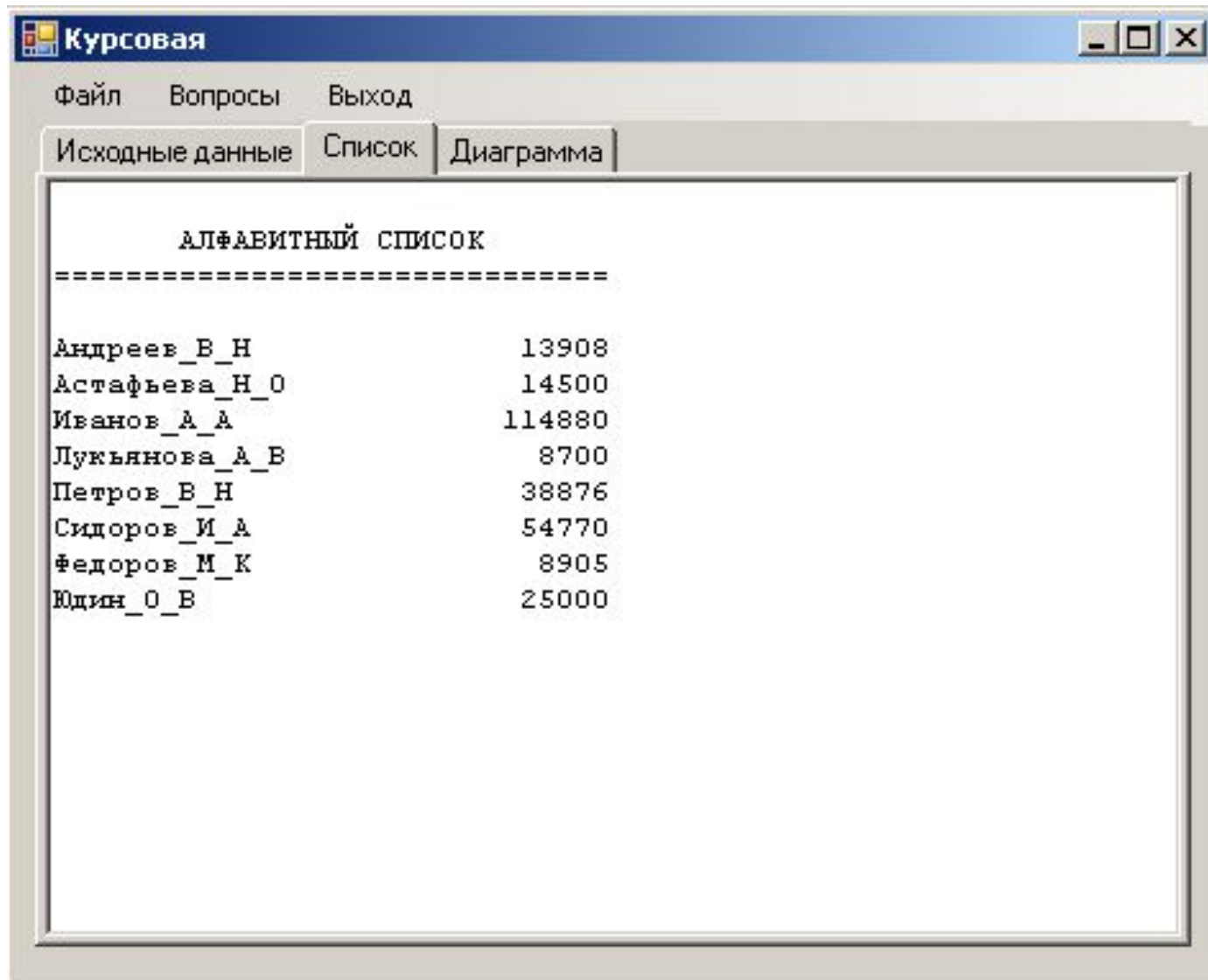
```
private: System::Void
диаграммаПроцентноеСоотношениеВложенныхДенегToolStripMenuItem_Click
(System::Object^ sender, System::EventArgs^ e)
{
tabControl1->SelectTab(2); // переходим на вкладку «ДИАГРАММА»
}
```

Запускаем на выполнение. Файл – Открыть ...



Имя	Статус	Сумма	Дата
Иванов_А_А	Срочный	15000	2002-09-23
Сидоров_И_А	Юбилейный	7150	1991-03-08
Петров_В_Н	Пенсионный	38876	1999-12-16
Сидоров_И_А	Сберегательный	12860	2008-06-23
Юдин_О_В	Особый	25000	2006-12-13
Федоров_М_К	До_востребования	8905	2005-11-14
Иванов_А_А	Пенсионный	75980	2003-08-04
Сидоров_И_А	Особый	34760	2004-02-07
Андреев_В_Н	До_востребования	13908	2007-04-09
Астафьева_Н_О	Срочный	14500	2006-11-15
Иванов_А_А	До_востребования	23900	2005-05-05
Лукьянова_А_В	Срочный	8700	1998-09-08

## Вопросы – Алфавитный список всех вкладчиков



# Вопросы – 1

The screenshot shows a Windows application window titled "Курсовая". The menu bar includes "Файл", "Вопросы", and "Выход". The "Вопросы" menu is open, displaying four options: "Какой вкладчик имеет на счету наибольшую сумму денег?", "Какой вклад был открыт раньше всех?", "Алфавитный список всех вкладчиков", and "Диаграмма. Процентное соотношение вложенных денег".

In the background, a table of deposit data is visible. The table has columns for Name, Term, Amount, and Date. The data is as follows:

Имя	Срок	Сумма	Дата
Иванов_А_А	До_востребования	75980	2003-08-04
Сидоров_И_А		34760	2004-02-07
Андреев_В_Н		13908	2007-04-09
Астафьева_Н_О		14500	2006-11-15
Иванов_А_А		23900	2005-05-05
Лукьянова_А_В		8700	1998-09-08

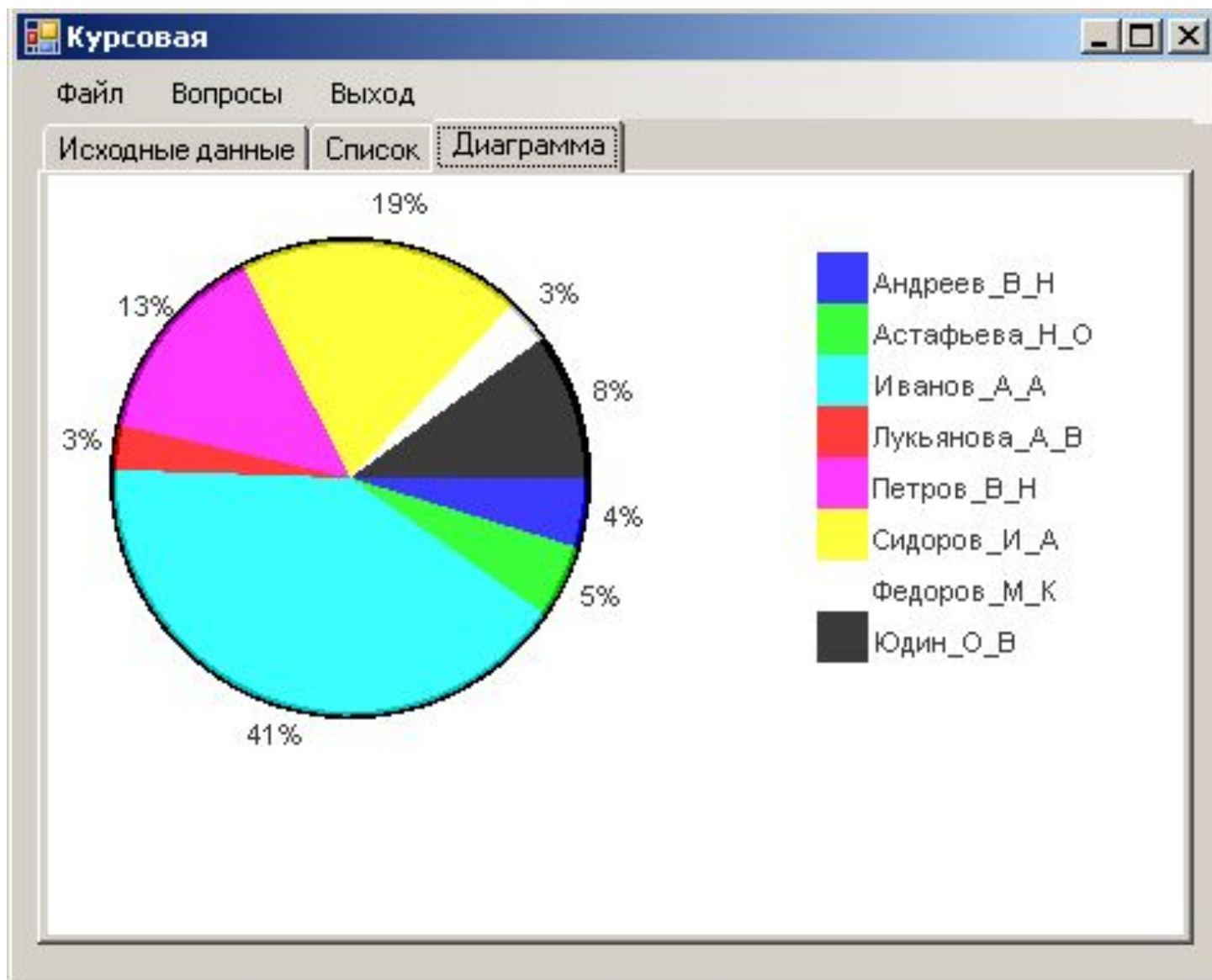
In the foreground, a dialog box titled "Наибольший вклад" is open, displaying the result: "Иванов\_А\_А" and "75980 рублей". An "OK" button is at the bottom of the dialog.

## Вопросы – 2

The screenshot shows a Windows application window titled "Курсовая". The menu bar includes "Файл", "Вопросы", and "Выход". The "Вопросы" menu is open, displaying four options: "Какой вкладчик имеет на счету наибольшую сумму денег?", "Какой вклад был открыт раньше всех?", "Алфавитный список всех вкладчиков", and "Диаграмма. Процентное соотношение вложенных денег". The second option is highlighted. In the foreground, a dialog box titled "Самый продолжительный вклад" is open, displaying the following information: "Юбилейный", "Сидоров\_И\_А", "7150 рублей", and "открыт 08 марта 1991". An "ОК" button is at the bottom of the dialog box. In the background, a table of data is visible, showing names and dates.

Имя	Дата
Иванов_А_А	2006-12-13
Федоров_М_К	2005-11-14
Иванов_А_А	2003-08-04
Сидоров_И_А	2004-02-07
Андреев_В_Н	2007-04-09
Астафьева_Н_О	2006-11-15
Иванов_А_А	2005-05-05
Лукьянова_А_В	1998-09-08

# Диаграмма



Доработка проекта



## 20. Добавим на форму компонент OpenFileDialog

The screenshot shows the Visual Studio IDE with a form named 'Form1.h' in the 'Конструктор' (Designer) view. The form contains a menu strip with 'Исходные данные', 'Список', and 'Диаграмма' items, and a list box labeled 'listBox1'. The 'Панель элементов' (Toolbox) is open, showing the 'Открыть файл' (OpenFileDialog) component selected. A tooltip for 'OpenFileDialog' is visible, stating: 'OpenFileDialog, Версия 2.0.0.0 из Microsoft Corporation, .NET Component. Отображает диалоговое окно, позволяющее пользователю открыть файл.'

The 'Свойства' (Properties) window for 'openFileDialog1' is also shown, displaying the following settings:

Свойства	
<b>openFileDialog1</b> System.Windows.F...	
[Иконки]	
Внешний вид	
Title	
Данные	
(ApplicationSetti	
FileName	<b>Vklad</b>
InitialDirectory	
Tag	
Поведение	
AddExtension	True
CheckFileExists	True
CheckPathExists	True
DefaultExt	
DereferenceLink	True
Filter	<b>Data files (*.dat)  </b>
FilterIndex	1
Multiselect	False
ReadOnlyChecke	False
RestoreDirectory	False
ShowHelp	False
ShowReadOnly	False
SupportMultiDott	False
ValidateNames	True
Проектирован	
(Name)	<b>openFileDialog1</b>
GenerateMembe	True
Modifiers	Private

Установим свойства OpenFileDialog:

FileName = Vklad

Filter = Data Files (\*.dat) | \*.dat

## 21. Перепрограммируем пункт меню «Открыть».

### Было

```
if ((in=fopen("Vklad.dat","r"))==NULL)
{
    MessageBox::Show("Файл не открыт!","Ошибка!",
        MessageBoxButtons::OK,MessageBoxIcon::Error);
    return;
}
```

### Делаем

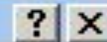
```
if (openFileDialog1->ShowDialog()==System::Windows::Forms::DialogResult::OK)
{
    s=openFileDialog1->FileName;
    char *str_tmp=(char*)(void*)Marshal::StringToHGlobalAnsi(s);
    if ((in=fopen(str_tmp,"r"))==NULL)
        {   MessageBox::Show("Файл не открыт!","Ошибка!",
            MessageBoxButtons::OK,MessageBoxIcon::Error);
        return;
        }
}
else return;
```

**В начало добавим** (где идет серия команд **using namespace**)

```
using namespace System::Runtime::InteropServices;
```






```
namespace Kursovoj {  
  
    using namespace System;  
    using namespace System::ComponentModel;  
    using namespace System::Collections;  
    using namespace System::Windows::Forms;  
    using namespace System::Data;  
    using namespace System::Drawing;  
    using namespace System::Runtime::InteropServices;  
  
    /// <summary>  
    /// Сводка для Form1
```




# Открыть



Папка:



-   
Недавние документы
-   
Рабочий стол
-   
Мои документы
-   
Мой компьютер
-   
Сетевое окружение

-  Debug
-  Kursovoj
-  Vklad

Имя файла:

Тип файлов:

Открыть

Отмена

# Как один проект разбить на несколько файлов

// Файл Struct.h – описание шаблонов структур

// + прототипы функций

#ifndef Struct\_H /\*При подключении к другим файлам фрагмент от #IFDEF\*/

#define Struct\_H /\*до #ENDIFбудет обрабатывается только один раз\*/

/\* Трактуется так: если Struct\_H не задана, то . . . , ИНАЧЕ ничего не делать\*/

struct z {

    char name[20];

    char vid[20];

    long summa;

    char data[11];

};

struct sp {

    char fio[20];

    long summa;

    struct sp\* sled;

};

#endif // Struct\_H

/\*При многократном использовании исключено повторное описание шаблона\*/

/\*

Здесь размещаются прототипы всех функций. Отовсюду их теперь можно убирать, т.к. за счет #include “Struct.h” они везде будут подключаться.

Вообще, прототипы могут обрабатываться компилятором несколько раз!!!

А вот повторное описание шаблона структуры НЕДОПУСТИМО!!!

Именно по этой причине мы применяем механизм #ifndef-#define-#endif

\*/

# Как один проект разбить на несколько файлов

## // Файл Main.c

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <alloc.h>
/* <malloc.h> для с++ */
```

```
#include "Struct.h "
```

```
int NC;
struct z *clients;
struct sp *spisok;
```

```
main()
{
. . .
}
```

## // Файл Func.c

```
#include <stdio.h>
#include <conio.h>
#include <string.h>
#include <alloc.h>
/* <malloc.h> для с++ */
```

```
#include "Struct.h "
```

```
extern int NC;
extern struct z *clients;
extern struct sp *spisok;
```

```
/*
Здесь можно разместить реализацию функций,
прототипы которых в Struct.h.
А вызываются они, например, из main()
*/
```

Файлы **Struct.h** и **Func.c** просто добавляются в проект, к уже имеющемуся там (как правило) **Main.c**

Файлов типа **Func.c** может несколько. Их строения подобны!

## Преобразования для типа **String**.

// Преобразование **String** в **char**

// Требуется подключить

```
using namespace System::Runtime::InteropServices;
```

```
String ^s;
```

```
char *str_tmp=(char*)(void*)Marshal::StringToHGlobalAnsi(s);
```

// Обратно

```
char *str;
```

```
String ^s=gcnew String(str);
```

// Число в **String**

```
String ^s=Convert::ToString(число);
```

// Обратно

```
int N=Int32::Parse(s);
```