ПРОГРАМИРАНЕ С VISUAL BASIC 6.0

1. Събитийно програмиране

При събитийното програмиране програмата е в аморфно (изчакващо) състояние и настъпването на определени събития определя кога и коя част от програмния код ще бъде изпълнена. В този вид програмиране изпълнението на програмата зависи от събитията, които ще възникнат. Реакцията на определено събитие чрез изпълнение на определен програмен код не е задължителна. Реакцията на събитие, за което липсва програмен код е стандартно, съгласно общоприетите разбирания на ОС. При събитийното програмиране класовете от обекти имат допълнителна характеристика: набор от събития, на които техните екземпляри имат право да реагират.

Инициатори на събития. Инициатор (генератор) на едно събитие може да бъде:

- извършване на определено действие на потребителя, който използва програмата – натискане на клавиш, придвижване или кликване на мишката и др.
- оповестяване, че текущия сеанс на работа с компютъра завършва и др.
- езиков процесор – изтичане на зададен период от време;
- програмният код – изпълнение на специфични оператори или прилагане на определени методи към даден екземпляр на обекта.

Какво представлява Visual Basic?

Той е интегрирана система за проектиране на програми, управлявани от събития, включваща:

- текстов редактор;
- транслатор от компилативен тип;
- програма за проверка и поправяне;
- рисуване на елементи, вместо тяхното записване (visual=визуален).

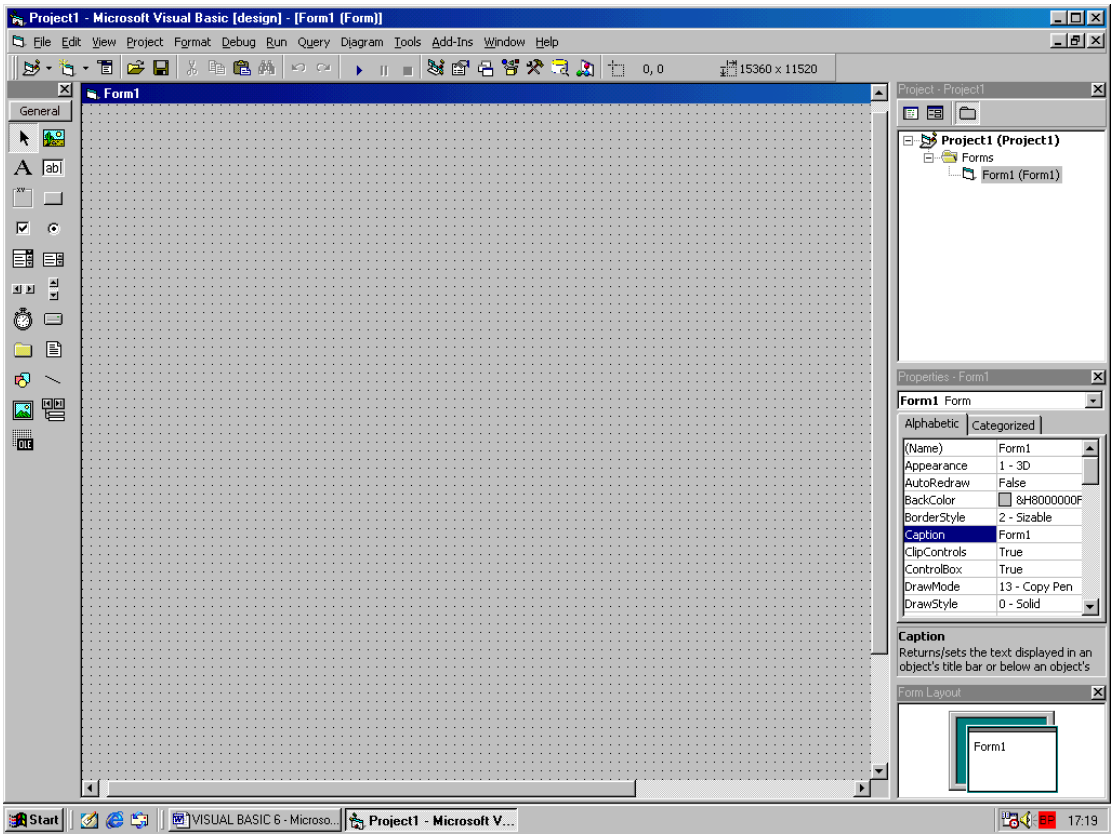
Разработил: инж. Даниела Цветкова – старши учител по информатика и ИТ, ПМГ “Екзарх Антим I“ гр. Видин

Той е език за програмиране, чрез който се описва реакцията на дадени обекти на възникнали събития.

2. Запознаване със средата Visual Basic 6.0

За стартиране на Visual Basic 6 е необходимо от старт менюто на Windows да се избере Programs/Microsoft Visual Basic 6.0/Visual Basic 6.0.

При стартиране на средата се визуализира следното работно пространство, което условно може да се раздели на 4 прозореца – главен прозорец, инспектор на обектите, прозорец на формата, редактор на програмен код



Главният прозорец служи за управление на разработките. Тук са разположени главното меню, панела за бърз достъп и палитрата с компонентите. Палитрата с компонентите е разположена вляво на главния прозорец. Именно от нея се избират контролите, които са необходими на приложението и ще се разполагат във формата. По подразбиране е отворена страницата Standart, където са поместени всички стандартни контроли.

Разработил: инж. Даниела Цветкова – старши учител по информатика и ИТ, ПМГ "Екзарх Антим I" гр. Видин

Инспектор на обекти – представлява прозорец с две страници, отбелязани с етикетите Project и Properties. Страницата със свойствата показва достъпните свойства на обекта, който е текущо избран в прозореца на формата. Стойностите, които свойствата могат да получават са следните:

- текст – пример за такива свойства са Caption и Name;
- избор от падащ списък - например свойството Color;

Прозорец на формата. При стартиране на Visual Basic 6.0 се започва нов, празен проект. Във всеки проект има поне една форма, която представлява главната форма на проекта. По подразбиране името на този прозорец е “Form1”.

Редактор на програмен код – той се намира зад прозореца на формата и първоначално не е видим. Този прозорец дава възможност да се редактира кода на програмата, написана на Basic. За да се достигне до него е необходимо двойно кликване върху избраната форма или върху контрола от формата. В горната си част той съдържа два падащи списъка. От левият се избира съответната контрола, за която ще се пише кода, а десният съдържа всички допустими събития съответстващи на избраната контрола.

Режими на работа. Използват се два режима: среда за разработка и среда за изпълнение на приложенията.

В средата за разработка се създават и модифицират обектите, въвежда се програмния код.

В средата за изпълнение се симулира изпълнението на приложението.

Задача: Да се създаде приложение за смяна на цвета на потребителския прозорец при натискане на бутон.

1. При стартиране на средата автоматично се отваря празна форма с име Form1;
2. За да се промени името на формата е необходимо в прозореца на свойствата в свойството Caption да се запише новото име, например “Моето първо приложение”;

Разработил: инж. Даниела Цветкова – старши учител по информатика и ИТ, ПМГ “Екзарх Антим I“ гр. Видин

3. За да се постави даден компонент на формата е необходимо да се избере от палитрата с компонентите и да се пренесе върху формата. Има възможност за промяна на размерите. Избира се контролата Command. Подобно на формата, за да се смени заглавието на бутона е необходимо да се смени аргумента на свойството Caption, например с “Промени цвета!”.

4. Чрез двойно кликване върху бутона се отваря редактора на програмен код. Записва се следния код:

```
Private Sub Command1_Click()
Form1.BackColor = vbGreen
End Sub
```

Задачи за изпълнение:

1. Напишете приложение, което посредством натискане на 3 бутона със заглавия “Червено”, “Зелено”, “Жълто” да променя цвета на формата в съответния цвят.
2. Напишете приложение, което посредством натискане на 5 бутона с различни заглавия да променя заглавието на формата със заглавието на бутона.
3. Напишете приложение, което посредством натискане на бутон да променя размерите на формата.
4. Напишете приложение, което посредством натискане на бутон да променя размерите на бутона.

3. Преглед на езика Visual Basic 6.0

3.1. Правила за запис

Начинът на записване на буквите не оказва влияние на написаното, т.е. A=a;

Прост оператор трябва да се запише изцяло в рамките на един логически ред;

Последователни физически редове могат да бъдат обединени в един логически ред чрез знака за подчертаване (_) в края си;

Някои части от структурните оператори трябва да са сами в рамките на един ред;

Операторите, записани в един ред се разделят с двоеточие (:).

3.2. Идентификатори

Те започват с буква, продължават с букви, цифри и знак за подчертаване и не трябва да бъдат служебни думи на езика.

Кирилицата е равноправна на латиницата.

Когато се използват за имена на програма броят на знаците е ограничен на 255, а на обекти е 40 знака.

3.3. Типове данни

- цяло число:

Byte – 0 - 255, 1 байт ОП, няма знак;

Integer - -32 768 – 32 767, 2 байта, знак %;

Long - -2 147 483 648 – 2 147 483 647, 4 байта, знак &.

- приближено число:

Single – 0 и абс.стойност $1{,}401298.10^{-45}$ – $3{,}402823.10^{+38}$, 4 байта, знак !;

Double – 0 и абс.стойност $4{,}94065645841247.10^{-324}$ – $1{,}79769313486232.10^{+308}$, 8 байта, знак #.

- логически – Boolean: True, False.
- знаков низ:

String – променлива дължина до около 2 милиарда, 10+брой знаци байта, знак $;

String*n – фиксирана дължина n до 65 400 знака, брой знаци байта, няма знак.

Нови типове данни:

- парична сума – Currency;
- дата и час – Date;
- универсален тип – Variant (той е по подразбиране).

3.4. Аритметични оператори

^ Степенуване;

* / Умножение и деление;

\ Целочислено деление (отрязва цифрите след десетичната запетая);

Mod Остатък от целочислено деление;

+ - Събиране и изваждане;

3.5. Оператори за сравнение

> по - голямо;

< по – малко;

>= по – голямо или равно;

<= по – малко или равно;

= равно;

<> различно;

3.6. Логически оператори

Not отрицание;

And логическо умножение (и);

Or логическо събиране (или);

3.7. Стандартни функции

Abs Абсолютна стойност;

Asc връща ASCII код на символ;

Chr връща символ, кореспондиращ със съответния ASCII код;

Cos косинус на ъгъл;

Format конвертира дата или число в текст;

Instr търсене на позицията на част от низ в друг низ;

Left отделя лявата част на низ;

Len дължина на низ;

Mid отделя част от низ;

Now	текущо време и час;
Right	отделя дясната част на низ;
Rnd	случайно число;
Sin	синус на ъгъл;
Sqr	квадратен корен на число;
Str	конвертира число в низ;
Trim	отрязва водещите интервали в низ;
Val	конвертира текст в число;

Примери:

```
Val(txtExample.Text)

MyNumber = 3.14159
txtExample.Text = Str(MyNumber)


MyNumber = 3.14159
txtExample.Text = Format(MyNumber, "#.##") //3.14

MyString = "Visual Basic is fun!"
LeftString = Left(MyString, 3) //"Vis"

MyString = "Visual Basic is fun!"
RightString = Right(MyString, 6) // "s fun!"

MyString = "Visual Basic is fun!"
MidString = Mid(MyString, 3, 6)// "sual B"

MyString = "Visual Basic is fun!"
LenString = Len(MyString)//20


MyString = "Visual Basic is fun!"
```

Location = Instr(3, MyString, “sic”)//10

Asc(“A”)//65

Chr(48)//1

<u>Randomize</u>

Number = Int(101 * Rnd) + 100 // между 100 и 200

3.8. Дефиниране на променливи и константи

Променливите могат да бъдат дефинирани по два начина:явна и неявна декларация.

Неявно дефиниране на променлива се осъществява само на процедурно равнище чрез записване на неизвестното до този момент име. Използването им е изключително опасно. Оператор Option Explicit в общата част на модул забранява неявно дефиниране в него. При неявно дефиниране и при липса на тип при явно дефиниране той се определя:

- по специфичен знак, записан след името;
- по първата буква на името;
- като универсален (Variant) в останалите случаи.

Явното дефиниране се осъществява чрез служебните думи Dim, Public, Private, Static. Public и Private могат да се използват само в общата част на стандартен модул и определят видимостта на името:

Public – в цялата програма;

Private – само в съответния модул;

Static – може да се използва само на процедурно ниво и определя време на живот равно на времето за изпълнение на програмата, т.е. подобни променливи запазват своята текуща стойност между две поредни изпълнения на съответната процедура.

Явно дефиниране на променлива може да се осъществи на всяко място от програмата, но до използването на името, чрез запис от вида:

Dim <име> As [New] <тип>;

Всички променливи се “раждат” със стойност 0 при числовите, празен низ “” при знаков низ и Empty при Variant. Стойността при обектовите променливи е Nothing без New или указател към създадения в момента нов екземпляр на класа при наличие на New.

Константи се дефинират чрез: [Public/Private] Const <име> As <тип>;<израз>. Пример: Const PI = 3.14159

Задача: Създайте приложение за пресмятане на квадратен корен по зададено произволно число. (Използват се текстови полета за вход и изход и 3 бутона за “Нови данни”, ”Пресмятане”, ”Изход”.

За да се реализира следната задача е необходимо да се приложи условен оператор за проверка на входните данни и стандартната функция квадратен корен SQR().

```
Private Sub cmdExit_Click()
    End
End Sub
Private Sub cmdNew_Click()
    TxtInput.Text=""
    TxtResult.Text=""
    TxtInput.SetFocus
 End Sub
Private Sub cmdRes_Click()
If TxtInput.Text<0 Then
        MsgBox “Въвели сте число, което попада извън дефиниционната област!”
        Exit Sub
Else
        TxtResult.Text=sqr(TxtInput.Text)
End if
End Sub
```

Синтаксис на оператора MsgBox:

MsgBox prompt, button+icon, title, helpfile, context;

Ако се използва като функция:

Dim Answer As Integer

Answer=MsgBox (prompt, buttons, title, helpfile, context);

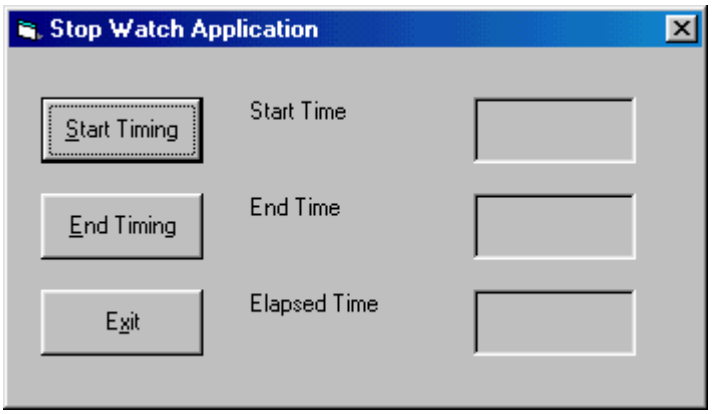
За валидизиране на данните се използват следните функции:

- ***IsNumeric, която връща стойност True, ако аргументът съдържа само цифри;***

- ***IsDate, която връща стойност True, ако аргументът е валидна дата.***

Задачи за изпълнение:

1. Напишете приложение за решаване на уравнение от типа: ax+b=0.
2. Напишете приложение за решаване на уравнение от типа: $ax^2+bx+c=0$.
3. Създайте приложение, което съдържа 3 командни бутона и 6 етикета. При натискане на бутоните StartTiming и EndTiming в етикетите се отбелязва съответното време, в което е бил натиснат всеки един от бутоните, ElapsedTime се получава като разлика от стойностите на предходните два етикета.




```
Option Explicit
Dim StartTime As Variant
Dim EndTime As Variant
Dim ElapsedTime As Variant

Private Sub cmdStart_click()
    StartTime = Now
    lblStart.Caption = Format(StartTime, "hh:mm:ss")
    lblEnd.Caption = ""
    lblElapsed.Caption = ""
End Sub

Private Sub cmdEnd_click()
```

```
    EndTime = Now
    ElapsedTime = EndTime - StartTime
    lblEnd.Caption = Format(EndTime, "hh:mm:ss")
    lblElapsed.Caption = Format(ElapsedTime, "hh:mm:ss")
End Sub

Private Sub cmdExit_click()
    End
End Sub
```

4. Някои оператори

4.1. Прости оператори

Операторите за присвояване:

Let <променлива>=<израз>;

Set <обектна променлива>=<обектен израз>;

Lset<променлива>=<знаков израз/променлива>;

RSet<променлива>=<знаков низ>.

Операторите за безусловен преход:

Явен: GoTo <етикет/ номер на ред>;

Неявен: Exit…(Do, For, Sub, Function).

Операторите за работа с форми:

Load <форма> - въвеждане в ОП;

Unload <форма> - извеждане от ОП.

4.2. Условни оператори

```
If <условие1> Then
   <оператори1>
Elseif <условие2>
   <оператори2>
Else
    <допълнителни оператори>
```

End if.

4.3. Оператор за многовариантен избор

Select Case <тестов израз>
    Case <списък1>
        <оператори1>
    Case <списък2>
        <оператори2>
    Case <списък n>
        <оператори n>
End Select.

4.4. Цикли с условие

<u>С предусловие:</u>

Do {While/Until} <условие>
    <оператори> Exit Do
Loop;

While <условие>
    <оператори>
Went;

<u>С постусловие:</u>

Do
    <оператори> Exit Do
Loop {While/Until} <условие>;

<u>Други цикли:</u>

For <променлива>=<нач.ст-ст> To <крайна ст-ст> Step <стъпка>
    <оператори> Exit For
Next

For Each <обектна променлива> In <колекция>
    <оператори> Exit For
Next.

4.5. Дефиниране на процедури и функции

Процедура:

Sub <име>

<оператори> End Sub

End Sub

Функция:

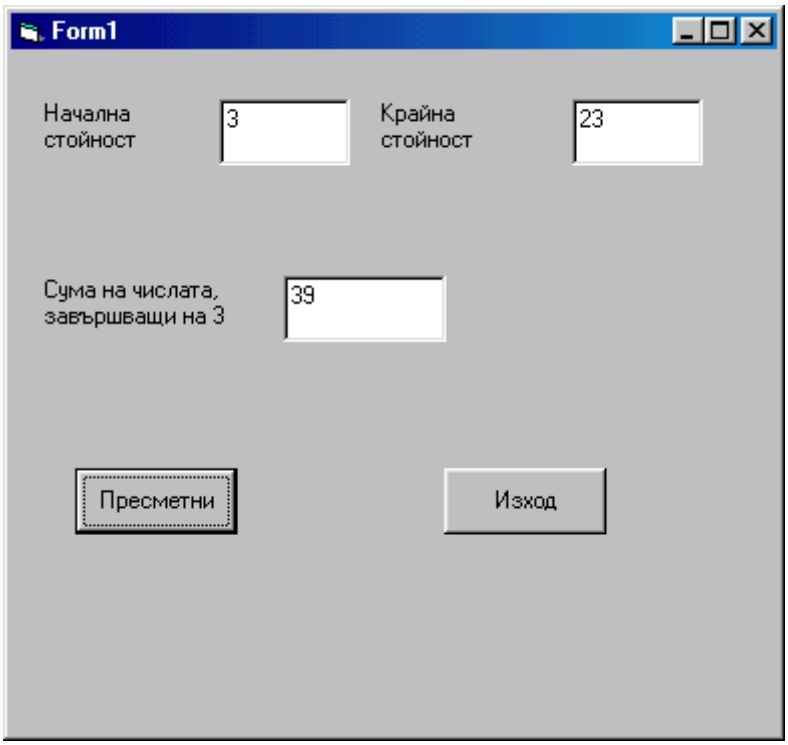
Function <име> As <тип>

<оператори> Exit Function

End Function

Процедурите се активират по два начина: чрез своето име и чрез Call <име>. Функциите се използват само в изрази, където се заместват с изчислената от тях стойност.

Задача: Да се създаде приложение за намиране на сумата на всички цели числа завършващи на 3 в даден диапазон. Долната и горната граница се въвеждат от потребителя.

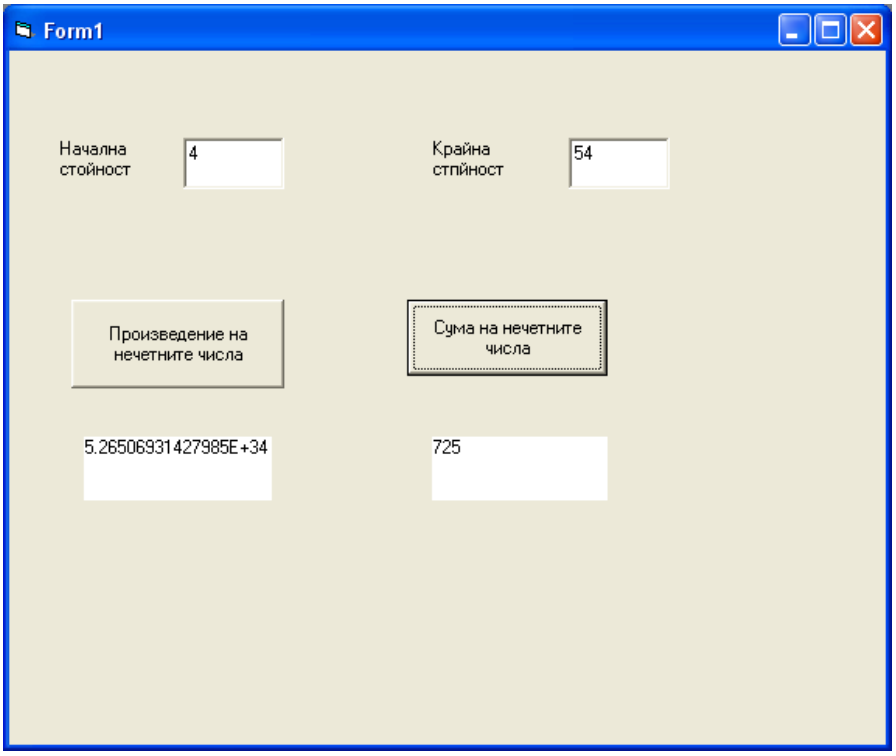



```
Private Sub Command1_Click()
Dim i, S As Integer
For i = Val(Text1.Text) To Val(Text2.Text)
```

```
    If i Mod 10 = 3 Then
      S = S + i
    End If
  Next
  Text3.Text = S
  End Sub
  Private Sub Command2_Click()
    End
  End Sub
```

Задачи за изпълнение:

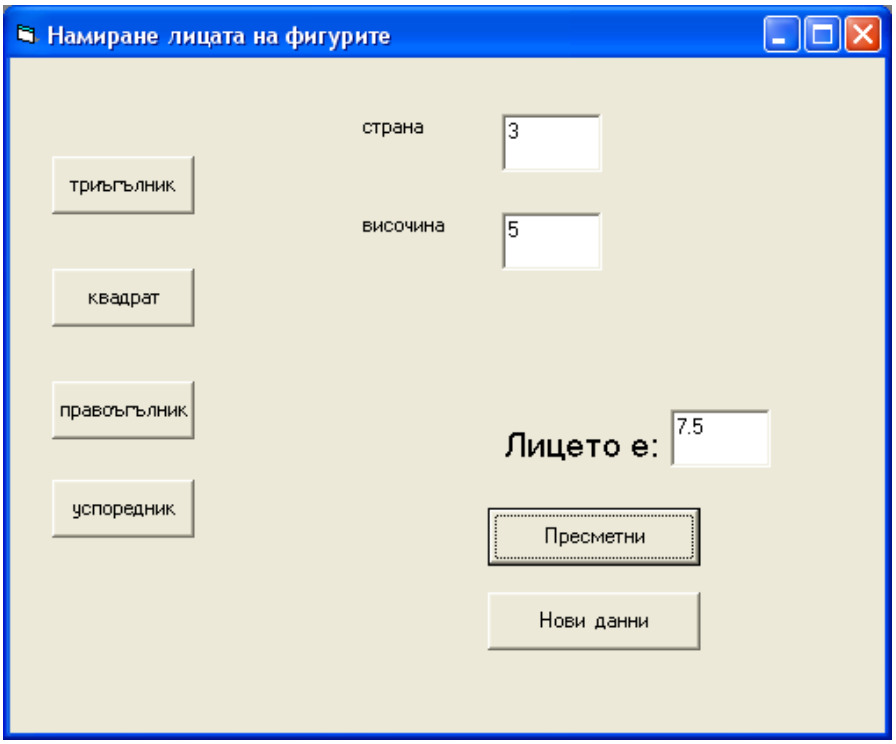
1. Да се състави приложение, което дава възможност на потребителя чрез бутони да избира да изчислява произведение или сума на нечетните числа в даден диапазон.




```
Dim s, p As Variant
Dim i As Integer

Private Sub Command1_Click()
p = 1
```

```
For i = Val(Text1.Text) To Val(Text2.Text)
   If i Mod 2 <> 0 Then
      p = p * i
   End If
Next
Label3.Caption = Format(p,"##.##")
End Sub

Private Sub Command2_Click()

For i = Val(Text1.Text) To Val(Text2.Text)
   If i Mod 2 <> 0 Then
      s = s + i
   End If
Next
Label4.Caption = Format (s,"##.##")
End Sub
```

2. Да се състави приложение за пресмятане на лица на фигурите (триъгълник, квадрат, правоъгълник, успоредник) по избор на потребителя.




```
Dim s, a, h, c As Variant

Private Sub Command1_Click()
Label1.Visible = True
Label1.Caption = "ñòðàíà"
Label2.Visible = True
Label2.Caption = "âèñî÷èíà"
Text1.Visible = True
Text2.Visible = True
Text1.SetFocus
c = 1

End Sub

Private Sub Command2_Click()
Label1.Visible = True
Label1.Caption = "ñòðàíà"
Text1.Visible = True
Text1.SetFocus
```

```
Label2.Visible = False
Text2.Visible = False
c = 2
End Sub

Private Sub Command3_Click()
Label1.Visible = True
Label1.Caption = "ñòðàíà a"
Label2.Visible = True
Label2.Caption = "ñòðàíà b"
Text1.Visible = True
Text2.Visible = True
Text1.SetFocus
c = 3
End Sub

Private Sub Command4_Click()
Label1.Visible = True
Label1.Caption = "ñòðàíà"
Label2.Visible = True
Label2.Caption = "âèñî÷èíà"
Text1.Visible = True
Text2.Visible = True
Text1.SetFocus
c = 4
End Sub

Private Sub Command5_Click()
If c = 1 Then
   Text3.Text = Val(Text1.Text) * Val(Text2.Text) / 2
ElseIf c = 2 Then
   Text3.Text = Val(Text1.Text) * Val(Text1.Text)
```

```
ElseIf c = 3 Then
   Text3.Text = Val(Text1.Text) * Val(Text2.Text)
ElseIf c = 4 Then
   Text3.Text = Val(Text1.Text) * Val(Text2.Text)
End If

End Sub

Private Sub Command6_Click()
Label1.Visible = False
Label2.Visible = False
Text1.Visible = False
Text2.Visible = False
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
c = 0
End Sub

Private Sub Form_Load()
Label1.Visible = False
Label2.Visible = False
Text1.Visible = False
Text2.Visible = False
End Sub
```

3. Да се състави приложение, което намира сумата на всички числа от 1 до 100 чрез трите различни видове оператори за цикъл.
4. Да се създаде приложение, което с помощта на 2 бутона дава възможност за събиране и отпускане на картинка (наподобява щора). Приложението да позволява картинката да се променя чрез избор на потребителя от 4 възможни други картинки.

```
   Dim h, p As Integer
```

```
Private Sub cmdChange_Click()
If p = 0 Then
        Picture1.Picture = Picture2.Picture
        p = 1
ElseIf p = 1 Then
        Picture1.Picture = Picture3.Picture
        p = 2
ElseIf p = 2 Then
        Picture1.Picture = Picture4.Picture
        p = 3
Else: Picture1.Picture = Picture5.Picture
        p = 0
End If
End Sub
Private Sub cmdClose_Click()
        End
End Sub

Private Sub cmdDown_Click()
Do While Picture1.Height < h
        Picture1.Height = Picture1.Height + 1
Loop
End Sub

Private Sub cmdUp_Click()
h = Picture1.Height
Do While Picture1.Height > 50
        Picture1.Height = Picture1.Height - 1
Loop
End Sub
```

5. Обекти в средата на събитийно програмиране и графичния потребителски интерфейс

5.1. Общи постановки

Формите са основен елемент на ГПИ, който съответства на понятието прозорец. Те носят върху себе си останалите елементи на ГПИ. Формите се създават от потребителя и имат статут на клас от обекти. Елементите, които носи формата са екземпляри на предварително създадени класове обекти. Подобно на формите някои елементи на ГПИ имат възможност да носят върху себе си други елементи на ГПИ.

5.2. Свойства обектите

Всеки клас от обекти има специфични свойства и методи, чрез които става използването на неговите екземпляри. Екземпляр от даден клас се цитира чрез указател към него (обектова променлива). Свойствата дават възможност за управление на екземпляра и сведения за неговото състояние. Някои свойства са достъпни само при проектиране, други – само при изпълнение и трети – в двете фази. Всяко свойство има стойност, която се нарича аргумент. Тази стойност може да бъде само от допустимите за съответния обект стойности. Например:

Command.Visible=True, Допустимите стойности за това свойство са True, False.

5.3. Методи на обектите

Всеки клас от обекти има специфични методи, чрез които става възможно да се използват неговите екземпляри. Част от методите се нуждаят от определен брой фактически параметри, чрез което те приличат на процедурите и функциите в обичайното процедурно програмиране. Всички екземпляри от определен клас реагират на набор от специфични за клас събития.

5.4. Работа с форми. Свойства и събития

Формите носят останалите елементи. Две форми в един проект не могат да носят еднакви имена (свойството Name), също така и два елемента, носени от една форма.

Формата се зарежда в паметта чрез оператора Load и прилагане към нея на метода Show;

Видимата част на формата се извежда от ОП чрез Unload.

Основни свойства:

- Име (Name) и Индекс (Index);
- Видимост (Visible);
- Достъпност (Enabled);
- Изглед (Appearance);
- Шрифт (Font);
- Ред на клавиш (Tab – Tabindex, Tabstop);
- Местоположение (Top, Left, Height, Width);

Основна мерна единица е 1/44 инча или 1/567 сантиметра.

Основни събития:

- Кликване с мишката (Click, DblClick;);
- Деактивиране (LostFocus);
- Активиране (GotFocus);
- Действия с мишката (MouseDown, MouseUp, MouseMove, DagOver, DragDrop);
- Действия с клавиатурата (KeyDown, KeyUp);
- Събитие промяна(Change);
- Проверка на предишния фокусиран елемент (CausesValidation);
- Потвърждаване (Validate);
- Нова позиция на плъзгача за елементи, които имат такъв (Scroll).

Като главен елемент, формите имат някои специфични свойства и събития:

Свойства	Събития

1. Дъщерна форма MDIChild – да/не 2. Обработка на клавишите KeyPreview – да/не 3. Стил на рамката BorderStyle 4. Елементи и място на прозореца ControlBox, StartUpPosition… 5. Икона Icon 6. Управление на графични методи DrawMode, DrawStyle…	1. Инициализация Initialize– създаване на нов екземпляр от конкретна форма 2. Завършване Terminate 3. Въвеждане в ОП Load 4. Заявка за извеждане QueryUnload 5. Извеждане от ОП Unload 6. Активиране на прозореца Activate 7. Деактивиране Deactivate 8. Промяна на размера Rasize 9. Опресняване Refresh, Paint

Константата Me е указател към екземпляра, за който се изпълнява кодът на формата. Нов екземпляр се създава с New. Формите се показват чрез метода Show [<стил>], където <стил> е vbModal – последователно или vbModeLess – паралелно (по подразбиране). Записването на True на свойството Visible е еквивалентно на прилагането на метода Show. Формата се скрива чрез метода Hide, еквивалентен на записване на False на свойството Visible.

Задача: Да се съхранят позицията и размера на формата.

Декларират се променливи, в които ще се запомнят размерите и позицията на формата.

```
Dim formTop, formLeft, formHeight, formWidth  As Integer
Private Sub Form_Load()
    formTop = Me.Top
    formLeft = Me.Left
    formHeight = Me.Height
    formWidth = Me.Width
End Sub
Private Sub Form_Click()
```

```
        Me.Top = formTop
        Me.Left = formLeft
        Me.Height = formHeight
        Me.Width = formWidth
    End Sub
```

Задачи за изпълнение:

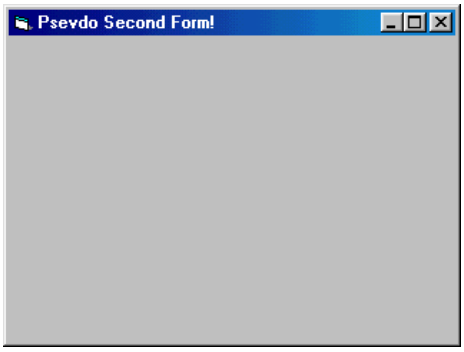
1. Като се използва формата от предходната задача да се проверява при изход дали потребителя иска да затвори формата.

```
Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
    Dim Answer As Integer
    Answer = MsgBox("Do you really want to close this window?", vbYesNo, "Events?")
    If Answer = vbYes Then
       Cancel = False
    Else
       Cancel = True
    End If
```

2. Да се създаде приложение, което клонира вече създадената форма.

Отваряме нова форма без да я създаваме от Visual Basic:

```
Private Sub Form_Click()
Dim x As New Form1
x.Caption = "Psevdo Second Form!"
x.Show
End Sub
```



6. Видове обекти на ГПИ

Обектите се избират от палитрата с класове обекти и се рисуват върху съответната форма. Обикновено тя съдържа стандартните. Могат да се добавят и други компоненти.

6.1. Прости компоненти

Етикет (Label) – за показване на надписи;

Текстова кутия (TextBox) – за вход и изход на всякакви текстови данни;

Команден бутон (CommandButton) – за реализиране на действия;

Рисунка (PictureBox) – която играе роля на групиращ носител за други елементи;

Изображение (Image) – използва по-малко системни ресурси и се рисува по-бързо.

Стандартни диалози – за избор на шрифт, цвят, печатащо устройство и име на файл се реализират с прилагане на методи към елемент за общ диалог (CommonDialog). Елементът няма видимо изображение във формата, но прилагането на метод показва свой прозорец. Методите, използващи различни свойства са:

ShowOpen – диалог за име на входен файл;

ShowSave – диалог за име на изходен файл;

ShowFont – диалог за избор на шрифт;

ShowColor – диалог за избор на цвят;

ShowPrinter – диалог за избор на печатащо устройство.

Този елемент не е част от стандартната палитра с елементи на ГПИ и трябва да се добави към нея.

6.2. Елемент за избор

Контролна кутия (CheckBox) осигурява възможност за избор на независими помежду си възможности от типа Да/Не. За задаване и определяне на положението се използва свойството Value със стийности vbUnchecked(0), vbChecked(1), vbGrayed(2).

6.3. Елемент за взаимно изключващ се избор

Изборният (радио) бутон (Option Button) дава възможност за избор на една от няколко възможности, които са взаимно изключващи се помежду си. Няколко бутона, които са разположени върху един и същ носещ елемент работят съвместно в група. Свойството Value е True или False.

6.4. Елементи за избор от списък

Списъчните кутии (List Box) и комбинираните (Combo Box) осигуряват по-сложни възможности за избор. Свойството Стил (Style) определя вида на елементите:

При комбинираните кутии стойностите са:

VbComboDropDown (0) – комбинация от текстова кутия за директен вход и падащ списък за избор;

VbComboSample (1) – комбинация от текстова кутия за директен вход и видим списък за избор;

VbComboDropList (2) – избор от разгъващ се списък.

При списъчните кутии стойностите са:

VbListBoxStandart (0) – показва се само текста;

VbListBoxCheckbox (1) – към текста има отметка.

Допълнителни свойства на списъците са:

Брой (ListCount) – недостъпно е при проектиране;

Списък (List) – масив от низове за избор;

Данни (ItemData) – масив от Long;

Множествен избор (MultiSelect) – 0-забрана, 1-прост, 2-разширен;

Сортиран (Sort) – Да/Не;

Избран ред (ListIndex) – (-1)-няма такъв, 0...ListCount-1;

Методи на списъци

Има възможност текстовете, които ще бъдат предложени да бъдат записани в свойството List още в режим на проектиране.

При режим на изпълнение те могат да бъдат променени с прилагане на следните методи:

Добавяне – AddItem <низ> ;

Изтриване – RemoveItem <индекс>;

Нулиране – Clear.

6.5. Използване на контроли

Има две категории ActiveX контроли – стандартни и потребителски. Потребителските са опционални, т.е. трябва допълнително да се добавят Project/ Components/ Controls.

Свойствата на контролите по време на проектиране се задават чрез Properties или <F4>: обект.свойство=израз.

Може да се използват и масиви с контроли, което означава група от контроли с едно и също име, тип и процедури за събития. Свойството Index отразява поредния номер на елемента в масива и номерацията започва от 0 до 32 767. Масивите имат три предимства:

- позволяват да се добавят контроли по време на изпълнение на програмата;
- всеки елемент онаследява общите свойства и процедури;
- могат да поделят код.

Масивите се създават по време на проектиране по следните начина: като се зададе едно и също име на две контроли от един и същ тип; чрез копиране; чрез свойството Index. Обръщението към контрол от масив става чрез неговото име и индекс - име_на_контрол(индекс).

По време на изпълнение се създават по следните начини:

- чрез конструкцията Load. Тя копира всички настройки без Visible, Index, TabIndex.

Задача: Да се създаде форма, която да съдържа 3 бутона Add, Remove и

Close. Бутона Add да дава възможност за добавяне на краен брой текстови курии върху формата, бутона Remove – да премахва избрана текстова кутия.

```
Private Sub Command1_Click()
Dim n As Integer
n=Text1().Count
Load Text1(n)
Text!(n).Top=Text1(n-1).Top+400
Text1(n).Visible=True
End Sub

Private Sub Command2_Click()
Dim k As Integer
k=Text1().Count-1
If k>0 Then
  Unload Text1(k)
End if
End Sub
```

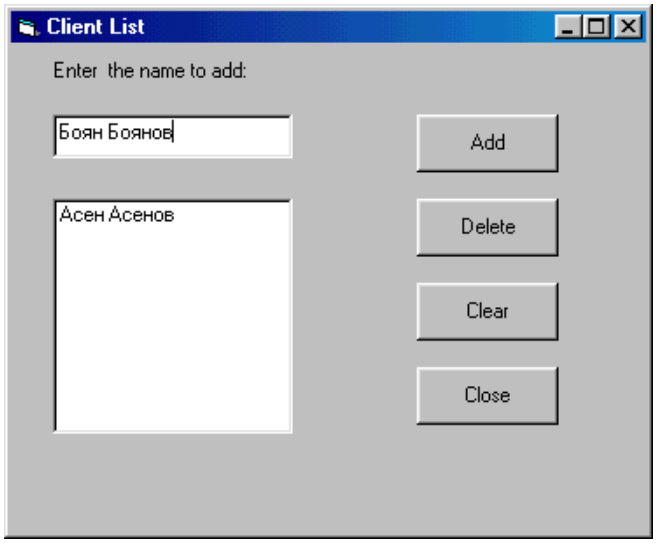
- чрез метода Add. Той дава възможност за динамично дабавяне без използване на масиви.

Задача: Динамично добавяне на команден бутон върху форма.

```
Private Sub Form_Load()
Form1.Controls.Add  “vb.CmmandButton”, “cmdObj”, Form1
With Form1.Controls(“cmdObj”)
  .Visible=True
  .Width=2000
  .Caption=”Dinamic Button”
End with
End Sub
```

Задачи за самостоятелна работа:

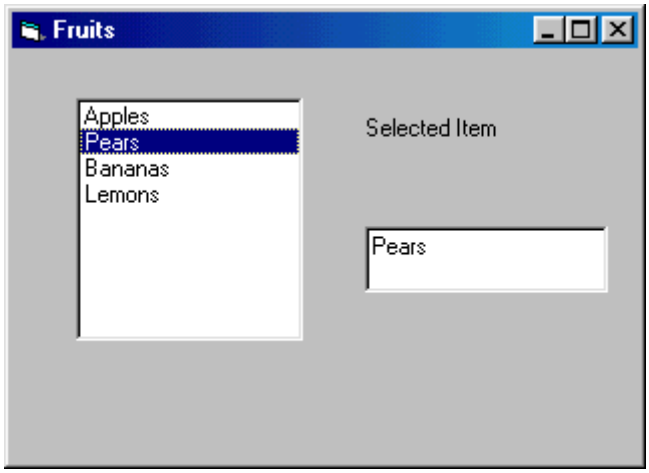
1. Създайте приложение, което съдържа следните елементи: списъчна кутия, текстово поле, етикет и 4 ком. бутона. Действието на всеки един от бутоните е следното: при въвеждане на произволна стойност в текстовото поле и натискане на бутона Add да се извършва добавяне на тази стойност в списъчната кутия; чрез бутона Delete – да се изтрива избран запис; чрез бутона Clear – да се изтриват всички записи; чрез бутона Close – да се излиза от приложението.




```
Private Sub cmdAdd_Click()
   lstClientList.AddItem txtInput.Text
   txtInput.Text = ""
   txtInput.SetFocus
End Sub

Private Sub cmdClose_Click()
End
End Sub

Private Sub cmdDelete_Click()
   If lstClientList.ListIndex >= 0 Then
      lstClientList.RemoveItem lstClientList.ListIndex
   Else
```

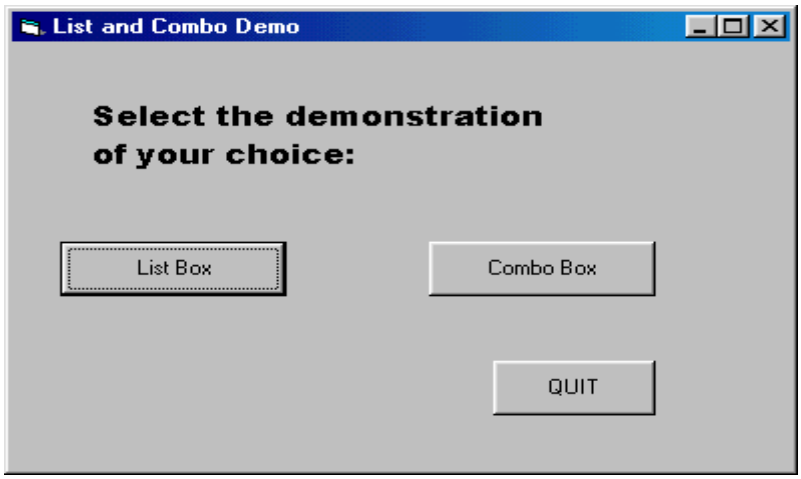
```
        Beep
    End If
End Sub
Private Sub cmdClear_Click()
    lstClientList.Clear
End Sub
```

2. Създайте приложение, което при избор на елемент от списъчна кутия стойността му се показва в текстово поле.

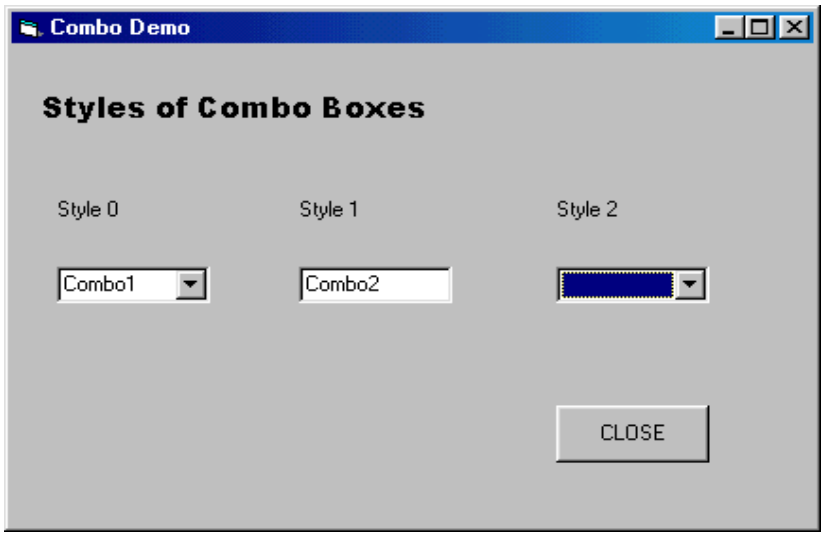



```
Private Sub lstFruits_Click()
 txtSelection.Text = lstFruits.List(lstFruits.ListIndex)
End Sub
```

3. Създайте приложение за демонстрация на комбинирана кутия и списъчна кутия, като се използва задача 1.




```
Private Sub cmdListBox_Click()
 frmClientList.Show
End Sub
Private Sub cmdComboBox_Click()
 ComboDemo.Show
End Sub
Private Sub cmdQuit_Click()
 End
End Sub
```




```
Private Sub cmdClose_Click()
 Unload Me
End Sub
```

4. Създайте приложение, което съдържа етикет, текстово поле, 3 опционални бутона и 2 полета за отметка. С помощта на опционалните бутони да се извършва избор на шрифт за текста в текстовото поле, а с с полетата за отметка да се избира стойност за свойствата FontBold и FontItalic на текстовото поле.
5. Създайте приложение, което съдържа еди команден бутон и 5 елемента Image. На три от картините са показани различните състояния на светофара и при стартиране те да са невидими, на четвъртата картина да е изобразена кола. Петиият елемент Image служи за визуализиране смяната на различните състояния на светофара при натискане на бутона Change. При зелено светещ светофар да се стартира движението на колата.

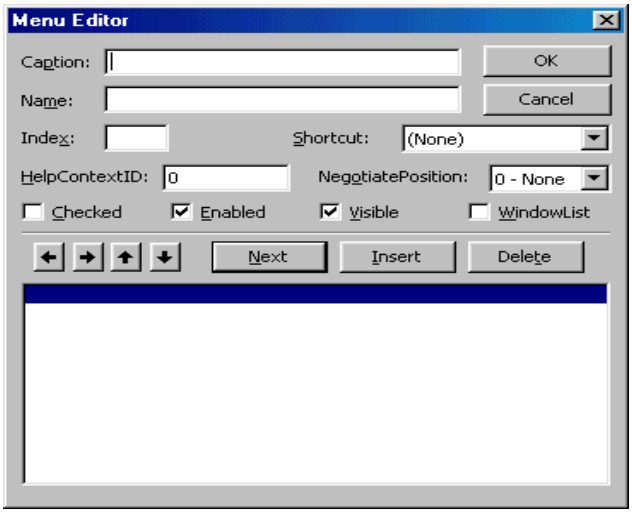
```
Option Explicit
Dim original As Single

Private Sub cmdChange_Click()
Select Case imglight.Picture
       Case imgGreen.Picture
          imglight.Picture = imgYellow.Picture
       Case imgYellow.Picture
          imglight.Picture = imgRed.Picture
       Case imgRed.Picture
          imglight.Picture = imgGreen.Picture
      original = imgCar.Left
      Do While imgCar.Left > 0
    imgCar.Left = imgCar.Left - 1
    Loop
     imgCar.Left = original
   End Select
End Sub

Private Sub Form_Load()
```

```
    imglight.Picture = imgYellow.Picture
End Sub
```

7. Работа с менюта

Менютата са важен елемент на формите. Те се проектират със специален редактор. Допустими са подменюта до четвърто ниво. Знакът (&) в името на менюто определя следващата буква като клавиш за бърз достъп. Знакът (-) като надпис единствено на подменю реализира разделителна ивица. Менютата разпознават само събитие Click(). Формите имат метод PopupMenu, чрез който се показват контекстни менюта. С оператори Load и Unload в масив от менюта може да се добавят и изтриват елементи.

Редакторът на менюта се избира Tools/ Menu Editor.



Пример за работа с меню: Създайте приложение, при което се използва вече създадената форма от стр. 6 за намиране на квадратен корен на произволно число и към нея се реализира хоризонтално меню, което да заменя бутоните.

8. Графика и анимация

8.1. Графични елементи и методи

При проектирането на форма чрез елементи отсечка (Line) и фигура (Shape) може да се реализират графични компоненти. Файлове с изображения се разполагат върху форма чрез елементите рисунка (PictureBox) и изображение (Image). За реализиране на графика във

форма се използва лява координатна система, т.е. т.O(0,0) е разположена в горен ляв ъгъл на работната форма.

Свойството Shape определя фигурата:

- vbShapeRectangle(0) – правоъгълник;
- vbShapeSquare(1) – квадрат;
- vbShapeOval(2) – елипса;
- vbShapeCircle(3) – окръжност;
- vbShapeRoundedRectangle(4) – правоъгълник със заоблени ъгли;
- vbShapeRoundedSquare(5) – заоблен квадрат.

Графични методи:

Формите и рисунките имат методи за чертане при изпълнение:

Line – отсечка и правоъгълник;

Circle – дъга, елипса, окръжност, кръг;

Print – изписване на текст във форма;

Cls – анулиране на изчертаната графика;

Pset – задава цвета на посочената точка;

Point – връща цвета на посочената точка;

PaintPicture –показване на файл с изображение.

С графичните методи са свързани свойствата:

CurrentX, CurrentY – текущи координати;

FillStyle, FillColor – стил и цвят на чертане;

BackColor – цвят на фона.

8.2. Анимация

Проста анимация се постига по 3 начина:

1. Създаване на невидими елементи Image, в които при проектиране в свойството Picture се подготвят изобрзженията, които при изпълнение ще се копират в същото свойство на видим елемент;
2. Промяна на мястото (Top, Left) на видим елемент;
3. Показване и скриване на елементи.

Необходимо е да се измерва времето. За целта се използва елемента Хронометър (Timer). Този елемент е без видимо изображение и има свойство Interval, определящо през колко милисекунди се генерира събитие Timer на разрешен хронометър.

Задача: Да се изчертае точка върху форма при кликване с мишката върху формата.

```
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
DrawWidth = 10 /дебелина на четката
ForeColor = QBColor(Rnd * 15) /произволен цвят
PSet (X, Y) /изчертаване на точката
End Sub
```

Задачи за изпълнение:

1. Да се изчертае линия от точка до точка при кликване с мишката. (използват се метода Line и свойството на обекта форма Me.AutoRedraw=true)

   ```
   Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
   Line (X, Y)
   DrawWidth = 4
   ForeColor = QBColor(Rnd * 15)
   Me.AutoRedraw = True
   End Sub
   ```

2. Да се изчертае окръжност с радиус 4 см с център O(3,4).

   ```
   Private Sub Form_Click()
   ScaleMode = 7/ свойство, определящо мерната единица за обекта (5-инчове; 6-милиметри; 7-сантиметри)
   Circle (3, 4), 4
   End Sub
   ```

3. Да се изчертае окръжност в центъра на формата. (Да се използва ScaleWidth/2 и ScaleHeight/2)

4. Да се изчертае окръжност с център, посочен от мишката и произволен цвят на линията.

9. Връзка с бази от данни

9.1. Как Visual Basic 6.0 извършва достъп до данни?

Интерфейсът за достъп до данни е обектен модел. Във Visual Basic има три различни интерфейса за достъп до данни:

- ActivX Data Object (ADO);
- Remote Data Object (RDO);
- Data Access Objects (DAO).

Може да се използват всяка от тези технологии за достъп до данни, но ADO е най-новата и най-мощната от тях.

Data Access Objects

Пътвата технология е DAO. Тя позволява да се извършва достъп до данните в локални и отдалечени бази от данни, както и да се управлява структурата на определени типове бази от данни. DAO осигурява йерархичен обектен модел, който улеснява използването му.

Той предоставя два основни начина за достъп до данни:

- Microsoft Joint Engine Technology (Jet) позволява да се извършва достъп до данни в настолни източници на данни като Microsoft Access, FoxPro, Paradox, Lotus 1-2-3.
- ODBCDirect позволява достъп до отдалечени сървъри за бази от данни без помощта на Microsoft Jet.

ODBCDirect е част от обектната библиотека DAO 3.5 и представлява негово разширение, а не отделна технология.

Ограничения: не се проектирана за работа в среда клиент/сървър; не може да извършва достъп до други източници освен бази данни.

Remote Data Objects

Тя е проектирана за работа с клиент/сървър, а не с настолни бази данни. RDO се възползва от предимствата на интелигентните сървъри за бази данни, които използват сложни машини за заявки като SQL Server и Oracle.

Ограничения: не е достатъчно ефикасно да се използва да извършва достъп до настолни бази от данни; не отговаря на нуждите на разработчиците на Интернет приложения.

ActivX Data Objects

ADO е най-новата технология на Microsoft за достъп до данни и се явява като интерфейс на OLE DB. OLE DB е стратегически интерфейс от ниско ниво за всички типове данни.

OLE DB е проектиран така, че да надскочи успеха на ODBC, като осигури отворен стандарт за достъп до всички видове данни.

9.2. Концепции на релационните бази данни.

Моделът на релационните бази данни е стандартът за проектиране на бази от данни. Базата данни съхранява и представя данните като набор от таблици. Структурата се дефинира, като се установят релациите между таблиците; по този начин данните се свързват в самата база данни. Таблицата представлява логическо групиране на свързана информация. Всеки запис съдържа информация за едно вписване в таблицата. Записът се състои от множество полета. Всяко поле съдържа определена информация за записа. За да се определи еднозначно даден ред, всяка таблица трябва да има първичен ключ. Първичният ключ представлява поле или група от полета, чиято стойност е уникална за всеки запис в таблицата.

Предимства на релационният модел бази от данни:

- Опростяване на дизайна чрез работа с таблици;
- Осигурява пълен релационен език за дефиниране, извличане и обновяване на данните;

- Предоставя правила за цялост на данните и подобряване на надеждността.

Релационните бази данни поддържат един стандартен език, наречен Structured Query Language (SQL).

9.3. Елементи за връзка с база данни

Стандартният елемент за връзка се нарича Data Control. Той поддържа връзка с БД Access до версия 3.5. Зад този елемент се скриват обекти БД (Database) и извадка (Recordset).

Задача: Създайте приложение, съдържащо 4 текстови полета, 1 етикет и един контрол Data. При преглеждане на различните записи в базата данни в етикета да се изписва съответно за всеки студент “Добър”, ако има успех под 4,50 и “Отличен”, ако има успех над 4,50. Базата данни ще съдържа 4 полета: факултетен номер, име, фамилия и успех.
Нека първо да създадем тази база данни: От главното меню се избира Add-Ins/Visual Data Manager/VisData От главното меню на VisData : File/New/Microsoft Access/Version 7.0 MDB Създаваме таблицата и след това се записва със съответното име – Students.mdb. Отваряме нова форма и нареждаме необходимите елементи, следва да свържем контрола Data с новосъздадената база данни: - На свойството DatabaseName избираме базата данни, с която ще работим; - На свойството RecordSource задаваме името на таблицата от базата данни. Следващите стъпки са да свържем текстовите кутии със съответните полета от таблицата: - свойството DataSource приема стойност името на контрола Data; - свойството DataField приема стойност името на

съответното поле, което ще се визуализира в текстовата кутия.

След тези настройки вече може базата данни да бъде разгледана посредством Data Control.

Кодът, който трябва да запишем е следният:

```
Private Sub Data1_Reposition()
If Val(Text4.Text) < 4.5 Then
   Label1.Caption = "Добър студент"
Else
   Label1.Caption = "Отличен студент"
End If
End Sub
```

*Събитието **Reposition** се използва при установяване на нов текущ запис (позициониране).*

Задачи за изпълнение:

Задача 1: Като се използва вече създадените база данни и форма да се добавят 4 бутона за добавяне на записи, изтриване, търсене по име, търсене с промяна.

За да се реши тази задача е необходимо да се знаят следните методи за работа с Data контрола:

MoveLast – изпрати на последния запис;

MoveFirst – изпрати на първия запис;

AddNew – добави нов запис;

FindFirst – намери първият отговарящ на критерия запис;

NoMatch – не е намерен идентичен запис;

Edit – редактиране на запис;

Update – обновяване на базата данни;

```
Private Sub Command1_Click()
Data1.Recordset.MoveLast
Data1.Recordset.AddNew
Data1.Recordset![firstname] = InputBox("Въведи име:")
```

```
Data1.Recordset![lastname] = InputBox("Въведи фамилия:")
Data1.Recordset![fac_nomer] = InputBox("Въведи фак.номер:")
Data1.Recordset![uspeh] = InputBox("Успех:")
Data1.Recordset.Update
End Sub

Private Sub Command2_Click()
Data1.Recordset.Delete
Data1.Recordset.MoveNext
End Sub

Private Sub Command3_Click()
Dim name, search As String
Data1.Recordset.MoveFirst
name = InputBox("Въведи име за търсене:")
search = "firstname='" & name & "'"
Data1.Recordset.FindFirst search
If Data1.Recordset.NoMatch Then
   MsgBox "Няма такова име!"
End If
End Sub

Private Sub Command4_Click()
Dim name, search As String
Data1.Recordset.MoveFirst
name = InputBox("Въведи име за търсене:")
search = "firstname='" & name & "'"
Data1.Recordset.FindFirst search
If Not Data1.Recordset.NoMatch Then
  Data1.Recordset.Edit
  Data1.Recordset![uspeh] = InputBox("Въведи успех")
  Data1.Recordset.Update
```

```
Else
   MsgBox "Няма такова име!"
End If
End Sub
```

Работа с елемента ADO Data Control:

Контролът ADO Data се използва за бързо създаване на връзки с обвързани с данни контроли и с доставчици на данни. Обвързани с данни контроли са тези, които притежават свойството DataSource, включително CheckBox, ComboBox, Image, Label, ListBox, PictureBox и TextBox. В допълнение на това са включени и други като DataGrid, DataCombo, DataList, Chart. ADO Data контролът се добавя ръчно от компонентите.

Свързване към източник на данни – извършва се чрез свойството ConnectionString. След това се дефинира свойството RecordSource.

Свойства на ADO Data:

Част от свойствата са:

- Връзка (ConnectionString) – определя БД;
- Източник (RecordSource) – име на таблица;
- Потребител (UserName) – потребител;
- Извадка (Recordset) – указател към Recordset;
- Лимит (MaxRecords) – максимален брой на записите;

Събития на ADO Data:

- Грешка (Error) – поява на грешка вън от ВБ;
- Следва придвижване (WillMove);
- Придвижването завърши (MoveComplete);
- Ще се променя поле (WillChangeField);
- Полето е променено (FieldChangeComplete);
- Ще се променя запис (WillChangeRecord);
- Записът е променен (RecordChangeComplete);
- Ще се променя извадка (WillChangeRecordset);

- Извадката е променена (RecordsetChangeComplete);
- Край на извадката (EndOfRecordset).

Задача: Да се създаде приложение, което съдържа 4 елемента текстови кутии и 1 ADO Data. Да се даде възможност на потребителя да разглежда базата данни.

Ще използваме вече създадена база по избор.

Поставяме елементите върху формата. Необходимо е да зададем низ за свързване с източника на данни. От прозореца Properties избираме свойството ConnectionString.

- В позицията Use ODBC Data Source избираме Name MS Access 97 Database
- При Use Connection String избираме Microsoft Jet 3.51 OLE DB Provider
- За Record Source избираме 2-adCmdTable
- Избраме базата данни, с която ще работим.

Нека обвържем текстовите кутии с избраната база данни. За това е необходимо да използваме свойствата DataSource (от къде ще взема данните) и DataField (кое поле ще визуализира) на текстовите кутии

Използване на Data Form Wizard

Вместо ръчно да се обвързват контролите може да се използва Data Form Wizard за създаването на форми, съдържащи обвързани контроли. Този модул трябва допълнително да се инсталира от главното меню: Add-Ins / Add-In Manager: в диалоговия прозорец се избира VB 6 Data Form Wizard и се поставя отметка в полето Loaded/Unloaded.

За да се използва от менюто Add-Ins се избира Data Form Wizard и се следват инструкциите за създаване на формата за разглеждане на базата данни.

Задача: Да се създаде приложение като се използва Data Form Wizard за обвързване на контроли за произволна база данни.

Задачи за изпълнение:

1. Създайте приложение, което съдържа следните елементи – ADO Data, 1 DataCombo и 1 ListBox. Базата данни да се създаде и да съдържа информация за служители в една фирма (име, фамилия, заплата). При селектиране на даден запис в DataCombo неговата стойност да се вписва в списъчната кутия.

   Свойствата на DataCombo: RowSource=adodc1; ListField=Firstname

```
Private Sub DataCombo1_Click(Area As Integer)
If Area = List Then
   List1.AddItem DataCombo1.Text
End If
End Sub
```

2. Върху същата форма да се добавят DataList и още една списъчна кутия. При селектиране на даден запис неговата стойност да се вписва в списъчната кутия.

   Свойствата на DataList: RowSource=adodc1; ListField=Firstname

```
Private Sub DataList1_Click()
List2.AddItem DataList1.Text
End Sub
```

3. Върху същата форма да се добавят DataGrid и още една списъчна кутия. При селектиране на даден запис неговата стойност да се вписва в списъчната кутия.

   Свойството на DataGrid: DataSource=adodc1

```
Private Sub DataGrid1_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
Dim col As Integer, bookmark As Variant
```

```
col = DataGrid1.ColContaining(X)
bookmark = DataGrid1.RowBookmark(DataGrid1.RowContaining(Y))
List3.AddItem DataGrid1.Columns(col).CellValue(bookmark)
End Sub
```

Писане на код за ADO контрол

ADO Data контролът има свойство наречено Recordset, което представлява група от записи. Това свойство само по себе си е обект, който има свои свпйства и методи. След като се дефинира Recordset може да се проверят свойствата му BOF и EOF. Те указват началото или края на Recordset обекта. Свойството Filter се използва за пресяване на записи по избран начин.

Методи:

Update - за обновяване на данните се използва метода;

AddNew - за добавяне на нови записи;

Delete - за изтриване;

Find - за търсене по критерии;

CancelUpdate - за отказ от обновяване.

Задача: Приложение за търсене на записи в база данни.

Върху формата се поставят 1 команден бутон, 1 текстова кутия и 1 Adodc1. Извършва се обвързване на Adodc1 с база данни students.mdb. В събитието Click на командния бутон се записва:

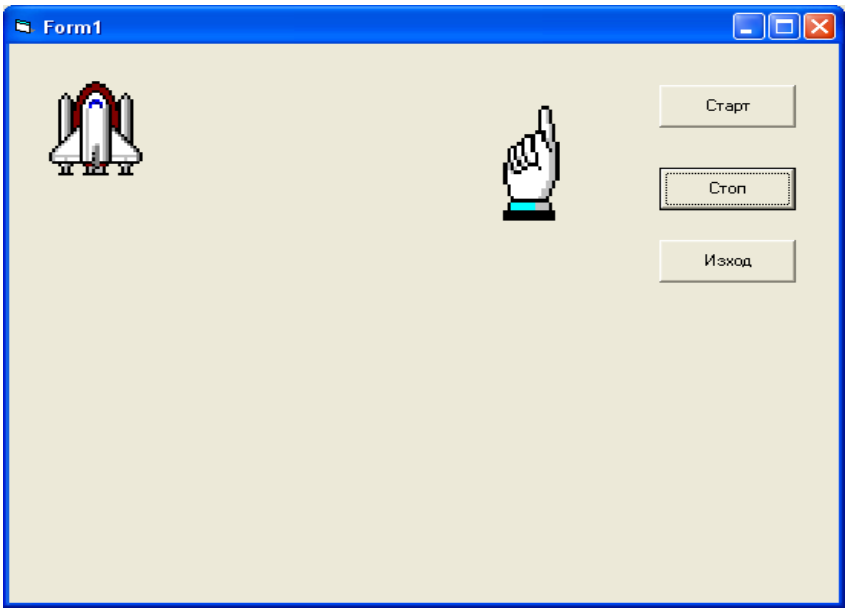
```
Adodc1.Recordset.MoveFirst
Adodc1.Recordset.Find “firstname=’Ivan’”
If Adodc1.Recordset.EOF then
   MsgBox “Record not found”
End if
```


Разработил: инж. Даниела Цветкова – старши учител по информатика и ИТ, ПМГ “Екзарх Антим I“ гр. Видин

ПРИЛОЖЕНИЕ 1

Проекти за самостоятелна работа

Задача1: Създайте приложение, което съдържа 3 командни бутона, 4 елемента Image (ракета, стрелка нагоре, стрелка надолу и празен Image) и 1 таймер.



При избиране на бутона “Старт”, таймера да започне да отброява интервалите за смяна на посоката, изобразена в “празния” Image и ракетата да се движи в указаната посока (нагоре и надолу).
При избиране на бутона “Стоп” – действието да се преустанови и при бутона “Изход” – да се визуализира съобщението:”Do you really want to quit” с 2 възможности за избор Yes, No. При избор на Yes да се излезе от приложението и при No да се връща управлението на формата.

```
Dim h As Integer

Private Sub Command1_Click()
Timer1.Enabled = True
End Sub

Private Sub Command2_Click()
Timer1.Enabled = False
End Sub

Private Sub Command3_Click()
```

```
End
End Sub

Private Sub Form_Load()
Image3.Picture = Image1.Picture
End Sub

Private Sub Timer1_Timer()

h = Form1.Top
Select Case Image3.Picture
   Case Image1.Picture
      Do While Image4.Top > 0
         Image4.Top = Image4.Top - 1
      Loop
      Image3.Picture = Image2.Picture
   Case Image2.Picture
      Do While Image4.Top < h
         Image4.Top = Image4.Top + 1
      Loop
      Image3.Picture = Image1.Picture
 End Select
End Sub
```

Задача2: Създайте приложение, което съдържа 3 командни бутона, 4 елемента Image (велосипед, стрелки – ляво и дясно и празен Image) и 1 таймер.

При избиране на бутона “Старт”, таймера да започне да отброява интервалите за смяна на посоката, изобразена в „празния” Image и велосипеда да се движи в указаната посока (наляво и надясно).
При избиране на бутона “Стоп” – действието да се преустанови и при бутона “Изход” – да се визуализира съобщението:”Do you really want to quit” с 2 възможности за избор Yes, No. При избор на Yes да се излезе от приложението и при No да се връща управлението на формата.

Задача 3: Създайте приложение, което да представлява изпитна листовка (кръстовище с най-малко 3 автомобила), с един въпрос (пример “Кой ще премине пръв?”), опционни бутони за избор на отговор и команден бутон “Изход” за отказване, при което да се визуализира “Сигурен ли сте, че се отказвате?”с възможности “Yes” и ”No”.

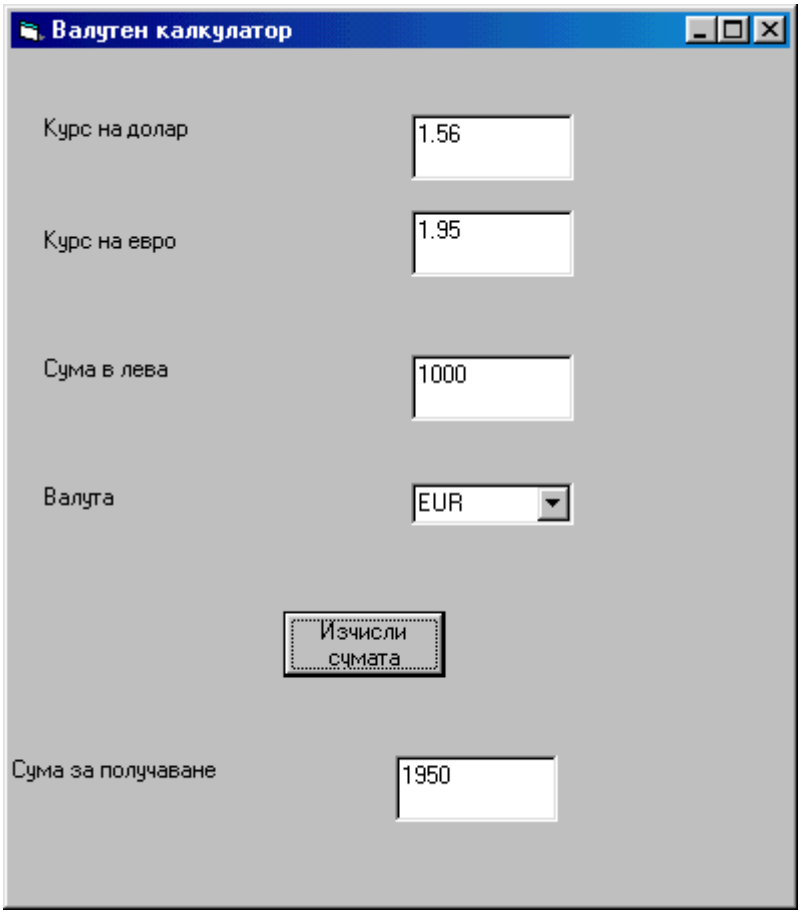
Да се даде възможност на потребителя да провери правилността на своя отговор чрез динамичен команден бутон “Провери”, който се визуализира след направения избор. При кликване върху този бутон да се изпише правилния отговор и да се анимира движението на автомобилите. (Листовката е по избор)

```
Private Sub Command1_Click(Index As Integer)

Select Case Index
```

```
  Case 0
    Unload Me
  Case 1
     If Option2.Value = True Then
        MsgBox "Вярно"
        Else
        MsgBox "Не е вярно"
     End If
        Option2.Enabled = True
        Option3.Enabled = True
        Option4.Enabled = True
End Select
End Sub

Private Sub Option2_Click()
Option3.Enabled = False
Option4.Enabled = False
dinamic
End Sub

Private Sub Option3_Click()
Option2.Enabled = False
Option4.Enabled = False
dinamic
End Sub

Private Sub Option4_Click()
Option2.Enabled = False
Option3.Enabled = False
dinamic
End Sub
Private Sub dinamic()
```

```
Dim n As Integer
n = Command1().Count
If n = 1 Then
Load Command1(n)
Command1(n).Top = Command1(n - 1).Top - 700
Command1(n).Caption = "Провери"
Command1(n).Visible = True
End If
End Sub
```

Задача 4: Създайте приложение “Валутен калкулатор”



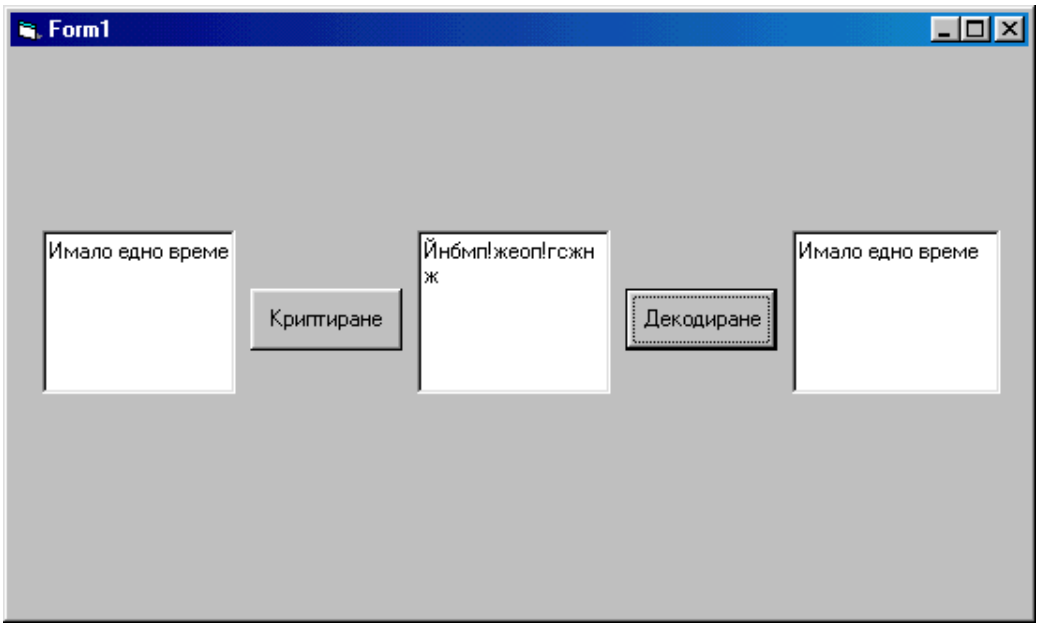

```
Private Sub Command1_Click()
If Text1.Text = "" Or Text2.Text = "" Or Text3.Text = "" Then
  MsgBox "Въведи стойност!"
Else
  If Combo1.ListIndex = 0 Then
    Text4.Text = Val(Text1.Text) * Val(Text3.Text)
```

```
    Else
      Text4.Text = Val(Text2.Text) * Val(Text3.Text)
    End If
  End If
  End Sub
```

Желателно е да се направи валидизация на данните, т.е да бъдат въвеждани само числа от клавиатурата.

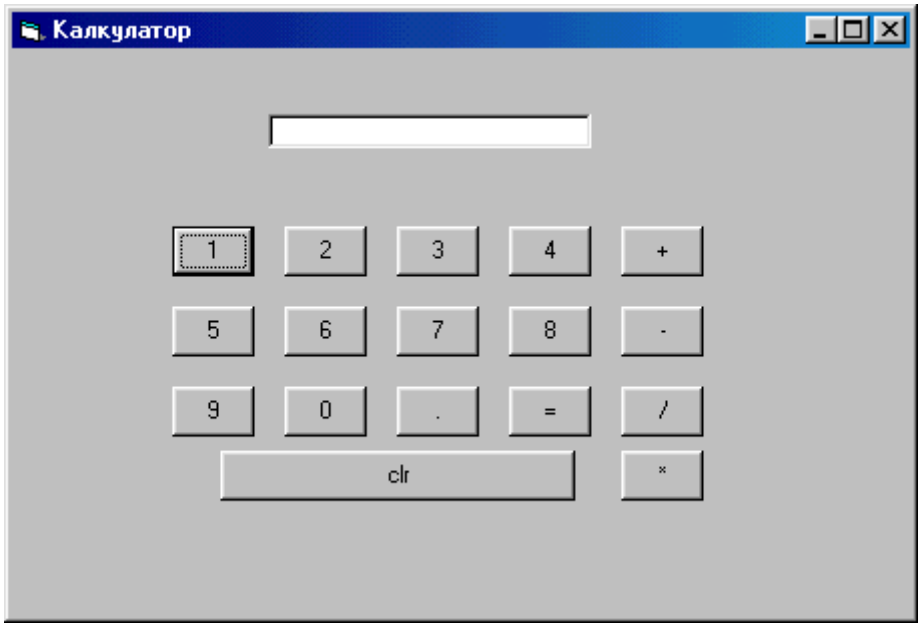
Задача 5: Създайте приложение, съдържащо 4 текстови полета, 1 етикет и един контрол ADO. При преглеждане на различните записи в базата данни в етикета да се изписва съответно за всеки студент “Добър”, ако има успех под 4,50 и “Отличен”, ако има успех над 4,50.
Базата данни ще съдържа 4 полета: факултетен номер, име, фамилия и успех. Да се добавят 5 командни бутона: Добавяне, Търсене, Търсене с промяна, Изтриване и Изход.

Задача 6: Създайте приложение за кодиране и декодиране на текст.



```
Private Sub Command1_Click()
    Dim s As String, i As Integer
    s = RichTextBox1.Text
    RichTextBox2.Text = ""
    For i = 1 To Len(s)
       RichTextBox2.Text = RichTextBox2.Text + Chr(Asc(Mid(s, i, 1)) + 1)
    Next
    End Sub

    Private Sub Command2_Click()
    Dim sd As String, i As Integer
    sd = RichTextBox2.Text
    RichTextBox3.Text = ""
    For i = 1 To Len(sd)
       RichTextBox3.Text = RichTextBox3.Text + Chr(Asc(Mid(sd, i, 1)) - 1)
    Next
    End Sub
```

Задача 7: Създайте приложение – калкулатор.




```
    Dim result, a(1) As Single
    Dim ind As Integer
```

```
Dim t As String
Private Sub Command1_Click(Index As Integer)
  Label1.Caption = Label1.Caption + Command1(Index).Caption
End Sub

Private Sub Command2_Click()
Label1.Caption = Label1.Caption + Command2.Caption
Command2.Enabled = False
End Sub

Private Sub Command3_Click()
a(ind) = Val(Label1.Caption)
Select Case t
Case "+"
   result = a(0) + a(1)
Case "-"
   result = a(0) - a(1)
Case "*"
   result = a(0) * a(1)
Case "/"
   If a(1) = 0 Then
     MsgBox "delenie na 0"
   Else
     result = a(0) / a(1)
   End If
   End Select
   Label1.Caption = result
End Sub

Private Sub Command4_Click(Index As Integer)
t = Command4(Index).Caption
Command2.Enabled = True
```

```
  a(ind) = Val(Label1.Caption)
  Label1.Caption = ""
  ind = 1
  End Sub

  Private Sub Command5_Click()
  Label1.Caption = ""
  ind = 0
  Command2.Enabled = True
  End Sub

  Private Sub Form_Load()
  Label1.Caption = ""
  ind = 0
End Sub
```