А.ШАКИРИН,
г.Минск, БАТУ, каф.ВТ.

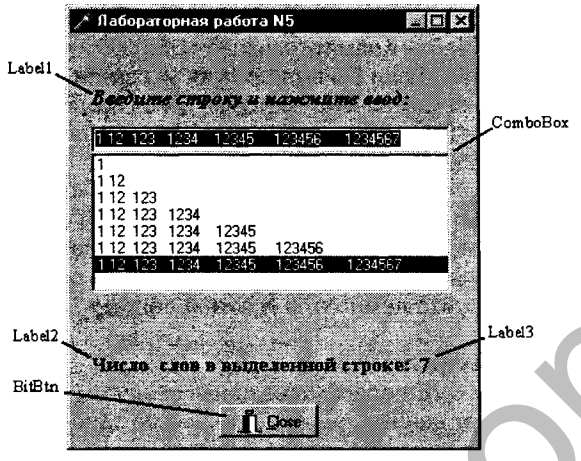
Программирование алгоритмов с использованием строк

Цель этой статьи — освоить применение компонентов ListBox и ComboBox и создать приложение, в котором используются строки.

1. Пример создания приложения

Необходимо создать Windows-приложение для подсчета количества слов в произвольной строке. Слова в строке разделяются любым количеством пробелов. Ввод строки заканчивается нажатием клавиши Enter. Работа приложения должна завершаться нажатием кнопки Close.

Один из возможных вариантов панели интерфейса создаваемого приложения показан на рисунке.



1.1. Размещение компонентов на Форме

При работе со строками ввод и вывод информации на экран удобно организовывать с помощью компонентов *ListBox* и *ComboBox*.

Компонент ***ListBox*** представляет собой список, элементы которого выбираются при помощи клавиатуры или мыши. Список элементов задается свойством *Items*, методы *Add*, *Delete* и *Insert* которого используются соответственно для добавления, удаления и вставки строк. Для определения номера выделенного элемента используется свойство *ItemIndex*.

Компонент ***ComboBox*** представляет собой комбинацию списка *ListBox* и редактора *Edit*, поэтому практически все свойства заимствованы у этих компонентов. Для работы с окном редактирования используется свойство *Text* — как в *Edit*, а для работы со списком выбора используется свойство *Items* — как в *ListBox*. Существует пять модификаций компонента, определяемых его свойством *Style*. В модификации *csSimple* список всегда раскрыт, в остальных он раскрывается после нажатия кнопки справа от редактора.

Компоненты *ListBox* и *ComboBox* находятся на странице *Standard* Палитры Компонентов.

Компонент ***BitBtn*** расположен на странице *Additional* Палитры Компонентов и представляет собой разновидность стандартной кнопки *Button*. Его отличительная особенность — наличие растрового изображения на поверхности кнопки, которое определяется свойством *Glyph*. Кроме того, имеется свойство *Kind*, которое задает одну из одиннадцати стандартных разновидностей кнопок. Нажатие любой из них, кроме *bkCustom* и *bkHelp*, закрывает модальное окно. Кнопка *bkClose* закрывает главное окно и завершает работу программы.

1.2. Создание процедур обработки событий

В момент запуска приложения, когда панель интерфейса появляется на экране, для пользователя удобно, чтобы курсор уже находился в поле редактора компонента *ComboBox*. При активизации Формы возникает событие *OnActivate*, которое можно использовать для передачи фокуса ввода компоненту *ComboBox*. Для создания процедуры-обработчика этого события необходимо в Инспекторе Объектов выбрать компонент *Form1*, на странице *Events* найти событие *OnActivate* и дважды щелкнуть мышью по его правой (белой) части. Курсор установится в тексте процедуры-обработчика события активизации Формы — **procedure TForm1.FormActivate(Sender: TObject)**. В этом месте процедуры наберите оператор передачи фокуса ввода компоненту *ComboBox1* (см. текст модуля UnStr, который приведен в п.1.3).

В соответствии с заданием необходимо, чтобы при нажатии клавиши Enter строка символов, которую пользователь набрал в поле редактирования, переносилась в список выбора компонента *ComboBox*. Для создания процедуры-обработчика этого события необходимо в Инспекторе Объектов выбрать компонент *ComboBox1*, на странице *Events* найти событие *OnKeyPress* и дважды щелкнуть мышью по его правой части. Курсор установится в тексте процедуры-обработчика события нажатия клавиши на клавиатуре — **procedure TForm1.ComboBox1KeyPress(Sender:TObject;var Key:Char)**. В этом месте процедуры, пользуясь текстом модуля UnStr, наберите операторы, которые при нажатии клавиши Enter переносят строку из поля редактирования в список выбора и очищают поле редактирования.

Процесс создания процедуры-обработчика события нажатия клавиши мыши в списке выбора **procedure TForm1.ComboBox1Click(Sender: TObject)** выполняется аналогично для события *OnClick* компонента *ComboBox1*. Пользуясь текстом модуля UnStr, наберите операторы, которые осуществляют основной алгоритм обработки символов выбранной строки.

1.3. Текст модуля UnStr

```
Unit UnStr;
interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls,
  Forms, Dialogs, StdCtrls, Buttons;
type
  TForm1 = class(TForm)
    Label2: TLabel;
    Label3: TLabel;
    BitBtn1: TBitBtn;
    ComboBox1: TComboBox;
    Label1: TLabel;
    procedure ComboBox1KeyPress(Sender: TObject; var Key: Char);
    procedure ComboBox1Click(Sender: TObject);
    procedure FormActivate(Sender: TObject);
  private
    { Private declarations }
```

```
  public
    { Public declarations }
  end;
var
  Form1: TForm1;

Implementation
{$R *.DFM}

// Обработка события активизации Формы
procedure TForm1.FormActivate(Sender: TObject);
begin
 ComboBox1.SetFocus;      // передача фокуса ввода ComboBox1
end;

// Обработка события ввода символа и нажатия клавиши Enter
procedure TForm1.ComboBox1KeyPress(Sender: TObject; var Key: Char);
begin
 if key=#13 then          // если нажата клавиша Enter то
  begin                   // строка из поля редактирования заносится
ComboBox1.Items.Add(ComboBox1.Text); // в список выбора
ComboBox1.Text:='';      // очистка окна редактирования
  end;
end;


// Обработка события нажатия клавиши мыши в списке выбора
procedure TForm1.ComboBox1Click(Sender: TObject);
var
 st         : string;
 n,i,nst,ind: integer;
begin
   n:=0;                    // n содержит количество слов
 ind:=0;
 nst:=ComboBox1.ItemIndex;  // определение номера выбранной строки
  st:=ComboBox1.Items[nst]; // st присваивается выбранная строка
  for i:=1 to Length(st) do // просмотр всех символов строки
    case ind of
    0 : if st[i]<>' ' then  // если встретился символ
                      begin
                      ind:=1;
                      n:=n+1;     // количество слов
                                  //увеличивается на единицу
                                end;
    1 : if st[i]=' ' then     // если встретился пробел
                      ind:=0;
end;
 Label3.Caption:=IntToStr(n); // вывод количества слов в Label3
end;
end.
```

2. Индивидуальные задания

Во всех заданиях исходные данные необходимо вводить с помощью компонента *Edit* в компонент *ListBox*, либо с помощью свойства *Text* в свойство *Items* компонента *ComboBox*. Результат следует выводить с помощью компонента *Label*, ввод строки заканчивать нажатием клавиши *Enter*. Работа приложения должна завершаться нажатием кнопки *Close*.

Для проверки функционирования приложения следует подготовить несколько тестов.

1. Дана строка, состоящая из групп нулей и единиц. Каждая группа отделяется друг от друга одним или несколькими пробелами. Найти количество групп с пятью символами.
2. Дана строка, состоящая из групп нулей и единиц. Каждая группа отделяется друг от друга одним или несколькими пробелами. Найти и вывести на экран самую короткую группу.
3. Дана строка, состоящая из групп нулей и единиц. Каждая группа отделяется друг от друга одним или несколькими пробелами. Подсчитать количество символов в самой длинной группе.
4. Дана строка, состоящая из групп нулей и единиц. Каждая группа отделяется друг от друга одним или несколькими пробелами. Найти и вывести на экран группы с четным количеством символов.

Представление (кодирование) информации в цифровых компьютерах

А.БЕЛОУСОВ,
E-mail: ark@mos.ru
WWW: http://hi-tech.nsys.by

Одним из базовых объектов арифметики является *число*. Что такое число? А Бог его знает... "Предметы я могу лишь называть. Их представляют знаки. Я могу лишь говорить о них, выговорить их я не могу". *(Людвиг Витгенштейн)*. Это вопрос философский, того же разряда, что и "какова цель жизни". Философию мы трогать не будем, отметим только, что число — вещь абстрактная, "пощупать нельзя", но которую можно "выразить", изобразить, а уж образы доступны для манипуляций, с чем мы и будем разбираться.

Почему число? Потому что цифровые компьютеры — устройства вычислительные, манипулирующие именно числами и для своей работы требующие выражения задач в числах. Соответственно, результат тоже представляется в числах, которые уже могут преобразовываться во что-то более приемлемое для человека — звук, графику, тексты и т.п.

Заметьте, что речь идет о цифровых компьютерах. Дело в том, что существуют и другие типы (аналоговые компьютеры, нейросети), функционирующие по иным законам, нежели дискретная математика. Соответственно, понятие числа для них не так уж и важно.

1. Системы счисления

Таким образом, мы разделяем число и его представления, называемые также системами счисления. Система счисления (СС) — это совокупность символов и правил обозначения чисел. Все СС разделяются на позиционные и непозиционные. Наиболее древние известные СС — непозиционные, и ныне они практически полностью вытеснены позиционными системами, хотя кое-какие рудименты древних СС используются до сих пор. Это и шестидесятиричная шумерско-ассирийская система для секунд и минут, и римская (латинская) запись.

В позиционных системах счисления (ПСС) числа представляются в виде последовательности цифр, в которой значение каждой цифры зависит от ее места. *Примечание:* используемая ныне римская система счисления является модификацией оригинальной системы. Например, в числе 4 (IV) мы видим, что размещение единицы перед пятеркой меняет знак единицы, в то время как в оригинальной системе 4 обозначалась бы четырьмя единицами (IIII).

Разновидностей ПСС много, но наиболее употребимы системы с постоянным основанием, где значение цифры домножается на основание в степени, заданной позицией циф-