УВАЖАЕМЫЕ СТУДЕНТЫ! Изучите теоретические сведения к лабораторной работе, выполните пример и задание согласно вашему варианту.

Результаты работы, отчет, предоставить преподавателю на e-mail: r.bigangel@gmail.com до 13.02.2023.

Требования к отчету:

Отчет предоставляется преподавателю в электронном варианте и должен содержать:

- **название работы, постановку цели, вывод;**
- **ответы на контрольные вопросы, указанные преподавателем.**

При возникновении вопросов по приведенному материалу обращаться по следующему номеру телефона: (072)111-37-59, (Viber, WhatsApp), vk.com: https://vk.com/daykini

ВНИМАНИЕ!!! При отправке работы, не забывайте указывать ФИО студента, наименование дисциплины, дата проведения занятия (по расписанию).

Лабораторная работа № 15

Тема: «Разработка программы с использованием ООП»

Цель работы: познакомиться с принципами объектно-ориентированного программирования в среде Delphi. Написать и отладить программу линейного алгоритма, в которой присутствовали бы некоторые критерии и примитивы качественного программного обеспечения.

Общие сведения

Класс – абстрактный тип данных, включающий свойства объекта (поля) и методы. Класс позволяет упростить процесс программирования, так как человеку проще представлять любой объект из реальности, обладающий некоторыми характеристиками (свойствами) и действиями, которые может совершать объект или которые можно совершать над ним.

Класс – это тип данных. Объект класса – переменная типа «класс». Из определения класса следует первое свойство ООП – инкапсуляция. Инкапсуляция данных означает, что они являются не глобальными –

доступными всей программе, а локальными – доступными только малой ее части. Инкапсуляция автоматически подразумевает защиту данных. Для этого в структуре class используется спецификатор раздела private, содержащий данные и методы, доступные только для самого класса. Если данные и методы содержатся в разделе public, они доступны извне класса. Раздел protected содержит данные и методы, доступные из класса и любого его *производного класса.*

Интегрированная среда разработки Delphi

Среда Delphi визуально реализуется в виде нескольких одновременно раскрытых на экране монитора окон. Количество, расположение, размер и вид окон может меняться в зависимости от текущих нужд, что значительно повышает производительность работы. При запуске Delphi можно увидеть на экране картинку (рис. 4.1).

Главное окно всегда присутствует на экране и предназначено для управления процессом создания программы. Основное меню содержит все необходимые средства для управления проектом. Пиктограммы облегчают доступ к наиболее часто применяемым командам основного меню. Через меню компонентов осуществляется доступ к набору стандартных сервисных программ среды Delphi, которые описывают некоторый визуальный элемент (компонент), помещенный программистом в окно формы. Каждый компонент имеет определенный набор свойств (параметров), которые программист может задавать, например цвет, заголовок окна, надпись на кнопке, размер и тип шрифта и др.

Окно инспектора объектов (вызывается с помощью клавиши F11) предназначено для изменения свойств выбранных компонентов и состоит из двух страниц. Страница Properties (Свойства) предназначена для изменения необходимых свойств компонента, страница Events (События) – для определения реакции компонента на то или иное событие (например, нажатие определенной клавиши или щелчок мышью по кнопке).

Окно формы представляет собой проект Windows–окна программы. В

это окно в процессе написания программы помещаются необходимые компоненты. Причем при выполнении программы помещенные компоненты будут иметь тот же вид, что и на этапе проектирования.

Окно текста программы предназначено для просмотра, написания и корректирования текста программы. В системе Delphi используется язык программирования Object Pascal. При первоначальной загрузке в окне текста программы находится текст, содержащий минимальный набор операторов для нормального функционирования пустой формы в качестве Windows-окна. При помещении некоторого компонента в окно формы текст программы автоматически дополняется описанием необходимых для его работы библиотек стандартных программ (раздел uses) и типов переменных (раздел type).

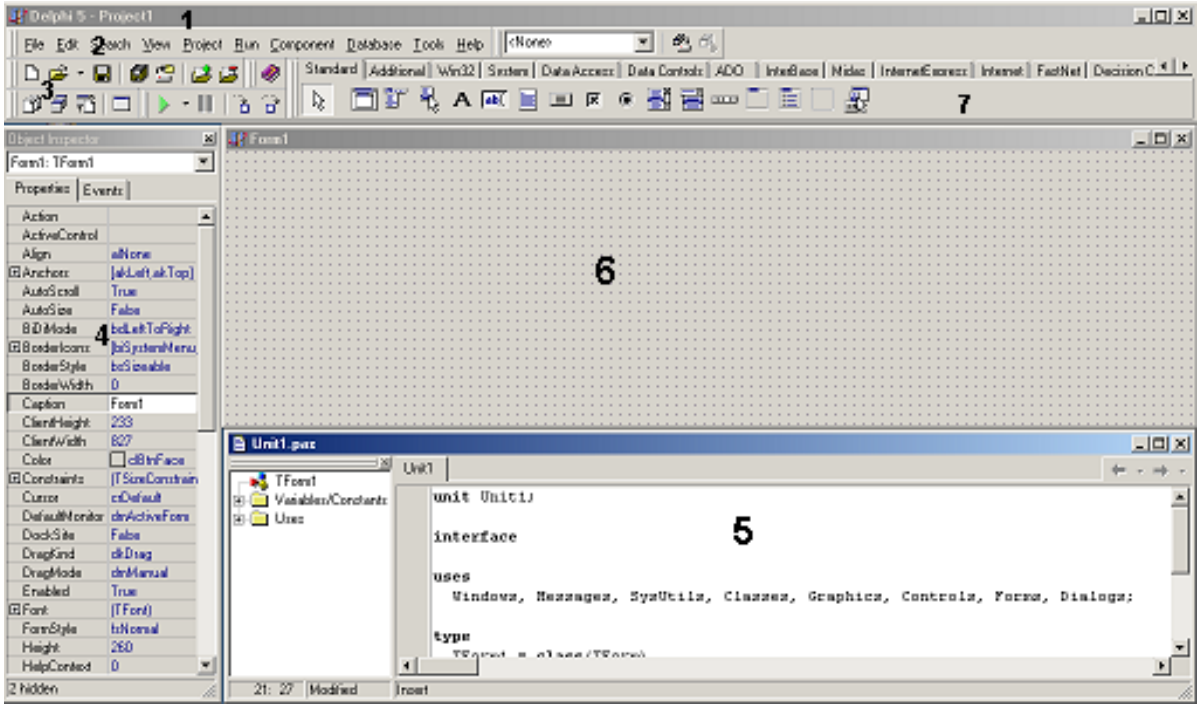

Рис. 4.1. Внешний вид окна Delphi.

1 – главное окно; 2 – основное меню; 3 – пиктограммы основного меню; 4 – окно инспектора объектов; 5 – окно текста программы; 6 – окно пустой формы, 7 – меню компонентов

Программа в среде Delphi составляется как описание алгоритмов, которые необходимо выполнить, если возникает определенное событие, связанное с формой (например, щелчок мыши на кнопке – событие OnClick, создание формы – OnCreate). Для каждого обрабатываемого в форме события с помощью страницы Events инспектора объектов в тексте программы организуется процедура (procedure), в которой записывается на языке Object Pascal требуемый алгоритм.

Переключение между окном формы и окном текста программы осуществляется с помощью клавиши F12.

Меню и команды Delphi

Для того чтобы выдать команду в среде Delphi, можно воспользоваться тремя основными способами:

– с помощью меню;

– с помощью полоски SpeedBar (инструментальной линейки);

– с помощью SpeedMenu (одного из локальных меню, которое активизируется при нажатии правой кнопки мыши).

Меню File

Команды выпадающего меню File можно использовать для работы как с проектами, так и файлами исходного кода.

К командам, работающим с проектами, относятся New, New Application, Open, Reopen, Save Project As, Save All, Close All, Add to Project и Remove from Project. С файлами исходного кода работают команды New, New Form, New Data Module, Open, Reopen, Save As, Save, Close и Print. Основной командой является File/New, которую можно использовать для вызова экспертов, для начала работы с новым приложением, наследования формы из уже существующей и т.д. Для того чтобы открыть проект или файл исходного кода, с которыми вы работали последний раз, используйте команду File/Reopen.

Меню Edit

Стандартные возможности меню Edit применимы как к тексту, так и к

компонентам формы. Можно копировать и вставлять тот или иной текст в редакторе, копировать и вставлять компоненты в одной форме или из одной формы в другую. Также можно копировать и вставлять компоненты в другое групповое окно той же формы, например в панель или блок группы; копировать компоненты из формы в редактор, и наоборот. Delphi помещает компоненты в буфер обмена, преобразуя их в текстовое описание. Можно соответствующим образом отредактировать этот текст, а затем вставить его обратно в форму в виде нового компонента. Можно выбрать несколько компонентов и скопировать их как в другую форму, так и в текстовый редактор. Это может пригодиться, когда вам придется работать с рядом схожих компонентов. Вы сможете скопировать один компонент в редактор, размножить его нужное число раз, а затем вставить назад в форму целую группу.

Меню Search

Если вы выберете команду Incremental Search, то вместо того чтобы показать диалоговое окно, где вводится образец для поиска, Delphi переходит в редактор. Когда вы введете первую букву, редактор перейдет к первому слову, которое начинается с этой буквы. Продолжайте набор букв и, курсор будет последовательно переходить к словам, в начале которых будут стоять введенные символы. Эта команда очень эффективна и чрезвычайно быстра. Команда Browse Symbol вызывает Object Browser – инструмент, который можно использовать для просмотра многих деталей при исследовании откомпилированной программы.

Меню View

Большинство команд меню View применяются для отображения какого–либо окна среды Delphi, например Project Manager, Breakpoints List или Components List. Эти окна не связаны друг с другом. Команда Toggle Form/Unit используется для перехода от формы, над которой вы работаете, к ее исходному коду, и обратно. Команда New edit window создает дубликат окна редактирования и его содержимого. В Delphi это единственный способ

просмотреть два файла рядом друг с другом, поскольку редактор для показа нескольких загруженных файлов использует ярлыки. После дублирования окна редактирования могут содержать разные файлы. Последние две команды меню View можно использовать для удаления с экрана полоски SpeedBar и палитры Components, хотя при этом среда Delphi становится менее удобнее для пользователя. Команда Build All заставляет Delphi откомпилировать каждый исходный файл проекта, даже если после последней трансляции он не был изменен. Для проверки написанного кода без создания программы можно использовать команду Syntax Check. Команда Information дает некоторые подробности о последней выполненной трансляции. Команда Options применяется для установки опций проекта: опций компилятора и редактора связей, опций объекта приложения и т.д.

Меню Run

Меню Run можно назвать Debug (отладка), так как большинство команд в нем относится к отладке, включая саму команду Run. Программа, запускаемая внутри среды Delphi, выполняется в ее интегрированном отладчике (если не отключена соответствующая опция). Для быстрого запуска приложения используется клавиша F9. Остальные команды применяются в процессе отладки для пошагового выполнения программы, установки точек прерывания, просмотра значений переменных и объектов и т.п.

Меню Component

Команды меню Component можно использовать для написания компонентов, добавления их в библиотеку, а также для конфигурирования библиотеки или палитры компонентов.

Меню Tools

Меню Tools содержит список нескольких внешних программ и инструментальных средств. Команда Tools позволяет сконфигурировать это выпадающее меню и добавить в него новые внешние средства. Меню Tools также включает команду для настройки репозитория и команду Options,

которая конфигурирует всю среду разработки Delphi.

Работа с формами

Проектирование форм – ядро визуальной разработки в среде Delphi. Каждый помещаемый в форму компонент или любое задаваемое свойство сохраняется в файле, описывающем форму (DFM–файл), а также оказывает некоторое влияние на исходный текст, связанный с формой (PAS–файл). Можно начать новый пустой проект, создав пустую форму или начать с существующей формы (используя различные доступные шаблоны), или добавить в проект новые формы. Проект (приложение) может иметь любое число форм.

При работе с формой можно обрабатывать ее свойства, свойства одного из ее компонентов или нескольких компонентов одновременно. Для того чтобы выбрать форму или компонент, можно просто щелкнуть по нему мышью или воспользоваться Object Selector (комбинированный список в Object Inspector), где всегда отображены имя и тип выбранного элемента. Для выбора нескольких компонентов можно или нажать клавишу Shift и щелкать по компонентам левой кнопкой мыши, или отбуксировать в форме рамку выбора.

SpeedMenu формы содержит ряд полезных команд. Для изменения относительного расположения компонентов одного вида можно использовать команды Bring To Front и Send To Back. Командой Revert To Inherited можно воспользоваться, чтобы в унаследованной форме установить те значения свойств выбранного компонента, которые были у них в родительской форме. При выборе сразу нескольких компонентов вы можете выровнять их или изменить их размеры.

С помощью SpeedMenu можно также открыть два диалоговых окна, в которых устанавливается порядок обхода визуальных управляющих элементов и порядок создания невизуальных управляющих элементов. Команда Add To Repository добавляет текущую форму в список форм, доступных для использования в других проектах.

Для установки положения компонента кроме применения мыши существуют еще два способа:

– установка значений для свойств Top и Left;

– использование клавиш курсора при нажатой клавише Ctrl.

Метод Ctrl+клавиша курсора особенно удобен при тонкой подстройке положения элемента. Точно также, нажимая клавиши курсора при нажатой клавише Shift, можно подстроить размер компонента.

Палитра компонентов

Для того чтобы добавить в текущую форму новый компонент, можно щелкнуть на одной из страниц палитры Components, а затем щелкнуть в форме, чтобы поместить новый элемент. Причем в форме можно или буксировать мышь с нажатой левой кнопкой, чтобы установить сразу и размер, и положение компонента, или просто щелкнуть один раз, позволяя Delphi установить размер по умолчанию.

Каждая страница палитры содержит компоненты, которые обозначены пиктограммами и именами, появляющимися в виде подсказки. Эти имена являются официальными названиями компонентов. В действительности это названия классов, описывающих компоненты без первой буквы T (например, если класс называется Tbutton, имя будет Button). Если необходимо поместить в форму несколько компонентов одного и того же вида, то при выборе компонента щелчком в палитре удерживайте нажатой клавишу Shift. Затем при каждом щелчке в форме Delphi будет вставлять новый компонент выбранного вида. Чтобы остановить эту операцию, просто щелкните по стандартному селектору (пиктограмма стрелки) слева от палитры Components.

Структура программ Delphi

Программа в Delphi состоит из файла проекта (файл с расширением .dph), одного или нескольких файлов исходного текста (с расширением .pas), файлов с описанием окон формы (с расширением .dfm).

В файле проекта находится информация о модулях, составляющих

данный проект. Файл проекта автоматически создается и редактируется средой Delphi и не предназначен для редактирования.

Файл исходного текста – программный модуль (Unit), предназначенный для размещения текстов программ. В этом файле программист размещает текст программы, написанный на языке PASCAL.

В разделе объявлений описываются типы, переменные, заголовки процедур и функции, которые могут быть использованы другими модулями через операторов подключения библиотек (Uses). В разделе реализации располагаются тела процедур и функций, описанных в разделе объявлений, а также типы переменных, процедуры и функции, которые будут функционировать только в пределах данного модуля. Раздел инициализации используется редко. Модуль имеет следующую структуру:

```
unit Unit1;
interface
// Раздел объявлений
implementation
// Раздел реализации
begin
// Раздел инициализации
end.
```

При компиляции программы Delphi создает файл с расширением .dcu, содержащий результат перевода в машинные коды содержимого файлов с расширением *.pas* и *.dfm*. Компоновщик преобразует файлы с расширением .dcu в единый загружаемый файл с расширением .exe. В файлах, имеющих расширение *.~df, .~dp, .~pa,* хранятся резервные копии файлов с образом формы, проекта и исходного текста соответственно.

Пример написания программы линейного алгоритма

Задание: составить программу вычисления для заданных значений *x, y, z* арифметического выражения:

$$u = tg^2(x+y) - e^{y-z} \cdot \cos x^2 + \sin z^2$$

Настройка формы

Пустая форма в правом верхнем углу имеет кнопки управления, которые предназначены для свертывания формы в пиктограмму , разворачивания формы на весь экран и возвращения к исходному размеру и для закрытия формы . С помощью мыши, «захватывая» одну из кромок формы или выделенную строку заголовка, отрегулируйте нужные размеры формы и ее положение на экране (рис. 4.2).

Новая форма имеет одинаковые имя (Name) и заголовок (Caption) – FORM1. Имя формы менять не рекомендуется, так как оно содержится в тексте программы.

Изменение заголовка формы

Для изменения заголовка вызовите окно инспектора объектов (F11) и щелкните кнопкой мыши на форме. В форме инспектора объектов найдите и щелкните мышью на Properties – Caption . В выделенном окне наберите «Лраб 1».

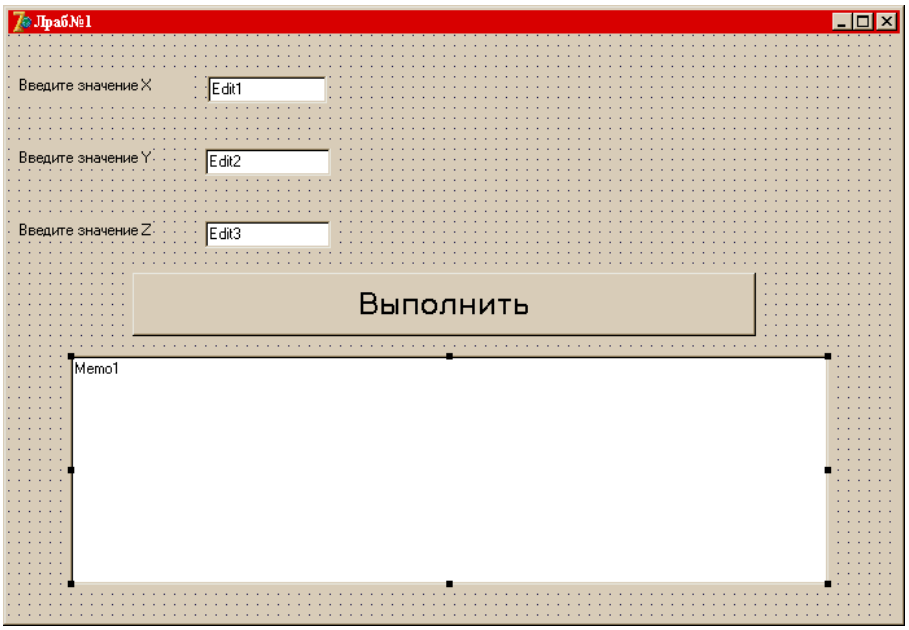

Рис.4.2. Внешний вид формы

Размещение строки ввода (TEdit)

Если необходимо ввести из формы в программу или вывести на форму информацию, которая вмещается в одну строку, используют окно однострочного редактора текста, представляемого компонентом TEdit. В данной программе с помощью однострочного редактора будут вводиться переменные *x, y, z* типа extended или integer.

Выберите в меню компонентов Standard пиктограмму  , щелкните мышью в том месте формы, где вы хотите ее поставить. Вставьте три компонента TEdit в форму. Захватывая их мышью, отрегулируйте размеры окон и их положение. Обратите внимание на то, что в тексте программы появились три новых однотипных переменных Edit1, Edit2, Edit3. В каждой из этих переменных с расширением .Text будет содержаться строка символов (тип String) и отображаться в соответствующем окне Edit.

Так как численные значения переменных *x, y, z* имеют действительный тип для преобразования строковой записи числа, находящегося в переменной Edit.Text, в действительное, используется стандартная функция X:=StrToFloat(Edit1.Text).

Если исходные данные имеют целочисленный тип, например integer, то используется стандартная функция X:=StrToInt(Edit1.Text).

При этом в записи числа не должно быть пробелов, а действительное число пишется с десятичной запятой.

С помощью инспектора объектов установите шрифт и размер символов, отображаемых в строке Edit (свойство Font).

Размещение надписей (TLabel)

На форме имеются четыре пояснительные надписи. Для нанесения таких надписей на форму используется компонент TLabel. Выберите в меню компонентов Standard пиктограмму  , щелкните на ней мышью. После этого в нужном месте формы щелкните мышью, появится надпись Label1. Проделайте это для четырех надписей. Для каждой надписи, щелкнув на ней мышью, отрегулируйте размер и, изменив свойство Caption

инспектора объектов, введите строку, например "Введите значение X:", а также выберите размер символов (свойства Font).

Обратите внимание на то, что в тексте программы автоматически появились четыре новых переменных типа .TLabel. В них хранятся пояснительные строки, которые можно изменять в процессе работы программы.

Размещение многострочного окна вывода (TMemo)

Для вывода результатов работы программы обычно используется текстовое окно, которое представлено компонентом (TMemo). Выберите в меню компонентов пиктограмму  и поместите компонент TMemo на форму. С помощью мыши отрегулируйте его размеры и местоположение. После установки с помощью инспектора свойства ScrollBars – SSBoth в окне появятся вертикальная и горизонтальная полосы прокрутки.

В тексте программы появится переменная Memo1 типа TMemo. Информация, которая отображается построчно в окно типа TMemo, находится в массиве строк Memo1.Lines. Каждая строка имеет тип String.

Для чистки окна используется метод Memo1.Clear. Для того чтобы добавить новую строку в окно, используется метод Memo1.Lines.Add (переменная типа String).

Если нужно вывести число, находящееся в переменной действительного или целого типа, то его надо предварительно преобразовать к типу String и добавить в массив Memo1.Lines. Например, если переменная u:=100 целого типа, то метод Memo1.Line.Add сделает это и в окне появится строка «Значение u=100». Если переменная u:=-256,38666 действительного типа, то при использовании метода Meno1.Lines.Add('Значение u=' + FloatToStrF(u.ffFixed,8,2)) будет выведена строка «Значение u= –256.39». При этом под все число отводится восемь позиций, из которых две позиции занимает его дробная часть.

Если число строк в массиве Memo1 превышает размер окна, то для просмотра всех строк используется вертикальная полоса прокрутки. Если

длина строки Memo1 превосходит количество символов в строке окна, то в окне отображается только начало строки. Для просмотра всей строки используется горизонтальная полоса прокрутки.

Написание программы обработки события нажатия кнопки(ButtonClick)

Поместите на форму кнопку, которая описывается компонентом TButton, для чего выберем в меню компонентов Standard пиктограмму  . С помощью инспектора объектов измените заголовок (Caption) – Button1 на слово «Выполнить» или другое по вашему желанию. Отрегулируйте положение и размер кнопки.

После этого два раза щелкните мышью на кнопке, появится текст программы, дополненной заголовком процедуры обработчика события – нажатия кнопки (Procedure TForm1.ButtonClick(Sender : TObject);).

Наберите текст этой процедуры, приведенный в примере.

Запуск и работа с программой

Запустить программу можно, нажав Run в главном меню Run, или клавишу F9, или пиктограмму  . При этом происходит трансляция и, если нет ошибок, компоновка программы и создание единого загружаемого файла с расширением .exe. На экране появляется активная форма программы.

Работа с программой происходит следующим образом. Нажмите (щелкните мышью) кнопку «Выполнить». В окне Memo1 появится результат. Измените исходные значения *x, y, z* в окнах Edit и снова нажмите кнопку «Выполнить» – появятся новые результаты. Завершить работу программы можно нажав в главном меню Run или кнопку {} на форме.

Текст программы:

```
unit tema1;
interface
uses
Windows, Messages. SysUtils, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls;
```

```
type
TForm1 = class(TForm)
Label1:TLabel;
Edit1:TEdit;
Label2: TLabel;
Edit2:TEdit:
Label3: TLabel;
Edit3: TEdit;
Label4: TLabel;
Memo1:TMemo;
Buttonl : TButfon;
procedure FormCreate(Sender: TObject);
procedure Button1Click(Sender: TObject);
 private
{ Private declarations }
 public
{ Public declarations }
 end;
var
Form1: TForm1;
Implementation
 {$R*.DFM}
```

```
procedure TForm1.Button1Click(Sender: TObject);
var
x, y, z, a, b, c, u : extended;
begin
x:=StrToFloat(Edit1.Text);                    // Считывается значение X
Memo1.Lines.Add('X = '+Edit1.Text);        // Вывод X в окно Memo1
y:=StrToFloat(Edit2.Text);                    //  Считывается значение Y
```

```
Memo1.Lines.Add('Y = '+Edit2.Text);        //  Вывод Y в окно Memo1
z:=StrToFloat(Edit3.Text);                    //  Ситывается значение Z
Memo1.Lines.Add('Z = '+Edit3.Text);        //  Вывод Z в окно Memo1
                                  // Вычисляем арифметическое выражение
a:=Sqr(Sin(x+y)/Cos(x+y));
b:=Exp(y–z);
c:=Sqrt(Cos(Sqr(x))+Sin(Sqr(z)));
u:=a–b*c;
                                  //  Выводим результат в окно Memo1
Memo1.Lines.Add('Результат  U = '+FloatToStrF(u,ffFixed,8,3));
end;
end.
```

Индивидуальные задания к выполнению

Ниже приведены 15 вариантов задач. По указанию преподавателя выберите свое индивидуальное задание. Уточните условие задания, количество, наименование, типы исходных данных. В соответствии с этим установите количество окон Edit, тексты заголовков на форме, размеры шрифтов, а также типы переменных и функции преобразования при вводе и выводе результатов. С помощью инспектора объектов измените цвет формы, шрифт выводимых символов.

Варианты заданий

1. Найти сумму цифр заданного четырехзначного числа.

2. Определить число, полученное в обратном порядке цифр заданного трехзначного числа.

3. Вывести на экран 1 или 0 в зависимости от того, равна ли сумма двух первых цифр заданного четырехзначного числа сумме двух его последних цифр.

4. Вывести на экран 1 или 0 в зависимости от того, равен ли квадрат трехзначного числа кубу суммы цифр этого числа.

5. Вывести на экран 1 или 0 в зависимости от того, есть ли среди первых цифр дробной части заданного положительного вещественного числа определенная цифра.

6. Вывести на экран 1 или 0 в зависимости от того, есть ли среди цифр трехзначного числа одинаковые.

7. Присвоить целой переменной k третью от конца цифру в записи положительного целого числа n.

8. Присвоить целой переменной k первую цифру из дробной части положительного вещественного числа.

9. Целой переменной S присвоить сумму цифр трехзначного целого числа k.

10. Идет k–я секунда суток. Определить, сколько полных часов (h) и полных минут (m) прошло к этому моменту.

11. Определить f – угол (в градусах) между положением часовой стрелки в начале суток и ее положением в h – часов, m – минут и s – секунд ($0 \le H \le 11$, $0 \le m, s \le 59$).

12. Определить h – полное количество часов и m – полное количество минут, прошедших от начала суток до того момента (в первой половине дня), когда часовая стрелка повернулась на f градусов ($0 \le f < 360$, f – *вещественное число*).

13. Пусть k – целое от 1 до 365. Присвоить целой переменной n значение 1,2,3,...,6 или 7 в зависимости от того, на какой день недели (понедельник, вторник, ..., суббота или воскресенье) приходится k-й день невисокосного года, в котором 1 января – понедельник.

14. Поменять местами значения целых переменных x и y, не используя дополнительные переменные.

15. Вывести на экран 1 или 0 в зависимости от того, имеют три заданных числа одинаковую четность или нет.