Т. А. Ронжина,
дипломант конкурса ИНФО-2015 в номинации «Методическая копилка учителя информатики», лицей № 11 им. Т. И. Александровой, г. Йошкар-Ола, Республика Марий Эл

ОТ ИДЕИ К РЕЗУЛЬТАТУ: ПРОЕКТЫ В СРЕДЕ LAZARUS

Аннотация

В предлагаемом сборнике практических работ представлены справочная информация по среде программирования Lazarus, а также практические работы с рекомендациями к выполнению и проектные задания для внеурочной деятельности.

Ключевые слова: Lazarus, объектно-ориентированное программирование (ООП), программирование, линейный алгоритм, алгоритм ветвления, сборник практических работ.

Контактная информация

Ронжина Татьяна Александровна, учитель информатики лицея № 11 им. Т. И. Александровой, г. Йошкар-Ола, Республика Марий Эл; *адрес:* 424000, Республика Марий Эл, г. Йошкар-Ола, ул. Комсомольская, д. 157; *телефон:* (836-2) 42-15-92; *e-mail:* chachyka89@mail.ru

T. A. Ronzhina,
Lyceum 11 named after T. I. Alexandrova, Yoshkar-Ola, Mari El Republic

FROM IDEA TO RESULT: PROJECTS IN THE ENVIRONMENT LAZARUS

Abstract

The proposed collection of practical works presents background information on the programming environment Lazarus as well as practical works with the recommendations and project tasks for extracurricular activities.

Keywords: Lazarus, object-oriented programming (OOP), programming, linear algorithm, branching algorithm, collection of practical works.

Программирование является центральной темой курса информатики. Именно по программированию проходит всероссийская олимпиада школьников по информатике. Тема алгоритмизации и программирования занимает около 40–50 % от общего материала в ЕГЭ. И если заглянуть в профессиональные перспективы, то профессия программиста — одна из самых востребованных. Однако при освоении программирования у многих школьников возникают сложности.

По программе изучение алгоритмизации и программирования начинается в VIII или IX классе. Уже после нескольких уроков большинство детей не воспринимают материал на должном уровне. Зададимся вопросом: «Почему непонятно?» Возможно, потому, что ранее ничего подобного не было и, сталкиваясь с данной темой, школьник не готов к ее восприятию? Тогда в каком классе необходимо знакомить детей с понятиями алгоритмизации? Чтобы ответить на данный вопрос, обратимся к этапам развития мышления ребенка. На ранней стадии развития ребенку нужно играть, выполнять действия с реальными объектами, с теми, которые можно увидеть, потрогать, возможно, изменить. Затем идет стадия наглядно-образного мышления, когда ребенок может мысленно выполнять действия с имеющимися изображениями и мысленными образами. И только после этого идет стадия понятийного мышления.

Мы предлагаем в изучении алгоритмизации и программирования провести подобную аналогию:

- действия с реальным объектом — робототехника (V—VII классы);
- наглядные образы — объектно-ориентированное программирование (VI—VIII классы);
- понятийное мышление — процедурные языки программирования (IX—XI классы).

В предлагаемом методическом пособии собраны практические работы, которые погружают ребенка в мир объектно-ориентированного программирования (ООП). Теоретический материал, который используется при работе, предоставляет возможность проводить интеграцию предмета информатики с любыми другими предметами, например, практическая работа «Изделия из ниток» может быть выполнена в рамках интеграции с предметом «Технология», практическая работа «Калькулятор» — с предметом «Математика», практическая работа «Фигура и цвет» — как элемент урока геометрии. Возможности ООП позволяют решать многие задачи моделирования, например, в предметах естественнонаучного цикла рассматриваемые проекты часто являются результатом исследовательской деятельности учащихся.

К объектно-ориентированным системам визуального проектирования относятся Visual Basic, Delphi, C++ Builder, Visual C++. Все указанные среды являются платным программным обеспечением, поэтому изучать ООП в школе мы предлагаем на базе среды Lazarus. Lazarus — бесплатный и свободно распространяемый продукт. Его можно загрузить на официальном сайте производителя: http://lazarus.freepascal.org, где необходимо выбрать файл установки, соответствующий вашей операционной системе.

В среде Lazarus программист получает возможность не просто создавать программный код, но и наглядно (визуально) показывать системе, что бы он хотел увидеть. Технология визуального программирования по-

зволяет строить интерфейс будущей программы из специальных компонентов, реализующих нужные свойства. Каждый компонент содержит готовый программный код и все необходимые для работы данные, что избавляет программиста от создания того, что уже создано ранее. Среда визуального программирования Lazarus сочетает в себе компилятор, объектно-ориентированные средства визуального программирования и различные технологии, облегчающие и ускоряющие создание программы.

Методическое пособие разбито на четыре раздела.

В первом разделе приведена справочная информация. В таблицах представлены основные компоненты, их свойства и события, этапы сохранения проекта.

Во втором разделе представлены различные алгоритмические структуры, позволяющие изучить (повторить, вспомнить) теоретические основы программирования. Последовательность изложения теоретического материала максимально приближена к последовательности выполнения практических работ. В разделе представлены линейные алгоритмы и алгоритмы ветвления.

Практические работы, представленные в третьем разделе, имеют следующую структуру: текст задания, схема расположения объектов на форме, изображения окон проекта до и после выполнения программного кода. Практические работы содержат рекомендации к выполнению, особенно подробно разобрана первая работа.

В четвертом разделе собраны проектные работы, которые отличаются от практических отсутствием рекомендаций к выполнению. Большая часть представленных работ повторяет темы практических работ, поэтому их можно использовать при оценивании учащихся, как задание на дополнительную отметку, для детей, пропустивших занятие, или детей, занимающихся дистанционно. Такие работы, как «Погода» и «Поймай меня», могут быть предложены в рамках повышенного уровня сложности при изучении соответствующих тем.

На наш взгляд, представленное методическое пособие будет полезно учителям информатики при подготовке к урокам или может стать своеобразным самоучителем для ученика.

1. Справочная информация

1.1. Интерфейс среды

Среда Lazarus состоит из нескольких, вообще говоря, не связанных окон. Открыть все окна возможно при помощи вкладки *Вид*.

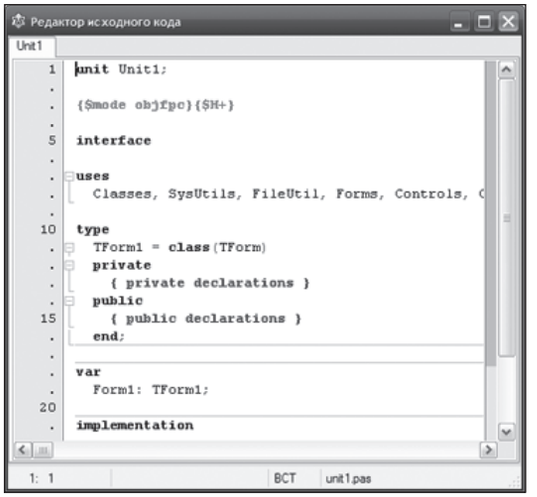
Главное окно.

С помощью этого окна можно управлять процессом разработки приложения. В нем предусмотрены команды управления файлами, компиляцией, редактированием, окнами и т. д. Окно разбито на три функциональных блока: главное меню, панель инструментов, палитра компонентов (рис. 1, *а*).

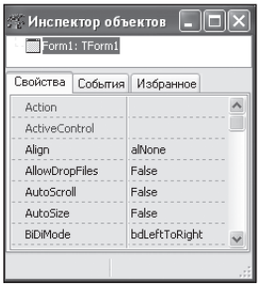
Редактор исходного кода Lazarus.

Именно в этом окне мы будем набирать тексты программ. Многие функции и возможности этого редактора совпадают с возможностями обычных текстовых редакторов, например Блокнота. Текст в редакторе можно выделять, копировать, вырезать, вставлять. Кроме того, в редакторе можно осуществлять поиск заданного фрагмента текста, выполнять вставку и замену. Основное преимущество редактора заключается в том, что он обладает возможностями подсветки синтаксиса, причем не только Pascal, но и других языков, а также рядом других удобств. В частности, выделенный фрагмент текста можно сдвигать вправо или влево на количество позиций, указанных в настройках *Окружение* → *Параметры* → *Редактор* → *Общие* → *Отступ блока*, что очень удобно для форматирования с целью структурирования кода (рис. 1, *б*).

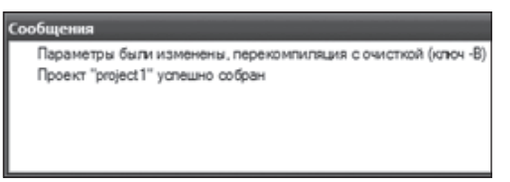
а) главное меню, панель инструментов и палитра компонентов



б) редактор исходного кода Lazarus



в) инспектор объектов



г) фрагмент окна сообщений

Рис. 1. Основные окна среды Lazarus

Инспектор объектов.

Назначение инспектора объектов — это просмотр всех свойств и методов объектов. В верхней части окна показывается иерархия объектов, а снизу расположены три вкладки: *Свойства*, *События*, *Избранное* (рис. 1, *в*). На вкладке *Свойства* перечисляются все свойства выбранного объекта. На вкладке *События* перечисляются все события для объекта. На вкладке *Избранное* — избранные свойства и методы.

Окно сообщений.

В этом окне выводятся сообщения компилятора, компоновщика и отладчика (рис. 1, *г*).

1.2. Основные компоненты (объекты)

Объект	Описание
	Текстовое окно, без права ввода информации Label — Метка
	Текстовое окно для ввода текстовой информации Edit — Окно ввода
	Управляющая кнопка Button — Кнопка
	Управляющая кнопка с изображением BitBtn — Кнопка
	CheckBox — Флажок
	RadioButton — Переключатель
	Список текстовых или числовых значений ListBox — Список
	Image — Изображение
	Поле вывода геометрических фигур Shape — Форма
	Рамка, объединяющая группу объектов GroupBox — Группа объектов
	Рамка, объединяющая группу переключателей RadioGroup — Группа переключателей
	Рамка, объединяющая группу флажков CheckGroup — Группа флажков
	Splitter — Разделитель
	Timer — Таймер

1.3. Основные свойства объектов

Свойство	Перевод
Align	Выровнять
Alignment	Выравнивание
AutoSize	Автоматический размер
BorderStyle	Стиль границы
Caption	Подпись
Checked	Принимает значение true, если выбрана радиокнопка или флажок, и значение false — в противном случае
Color	Цвет
Cursor	Курсор
Enabled	Включен
Font	Шрифт
Height	Высота
Hint	Текст подсказки
Left	Лево
Name	Имя
Picture	Загрузка изображения
Shape	Меняет вид фигуры Shape
ShowHint	Показать подсказку
Stretch	Растягивать (изображение)
Top	Верх
Visible	Видимый
Width	Ширина

1.4. Основные события

Событие	Перевод
BorderSpacing	Расстояние между границами
OnChangeBounds	Выбрать другую границу
OnClick	При нажатии
OnContextPopu	Выпадающее контекстное меню
OnDblClick	При двойном нажатии
OnDradDrop	Перетаскивание объекта
OnEndDrag	Когда закончили перетаскивать объект
OnMouseDown	При нажатии мышкой
OnMouseEnter	При наведении мышки
OnMouseMove	При движении мышки
OnMouseUp	Когда отпускаем мышку, но находимся над объектом
OnResize	При изменении размера
OnStartDrag	Когда начинаем перетаскивать объект

1.5. Сохранение проектов

Каждый новый проект сохраняется в отдельную папку. Нежелательно изменять имена файлов проекта и исходного кода. Сохранение проекта происходит в несколько этапов:

- *Файл*, *Сохранить как*;

- сохраните файл проекта под именем Project1;

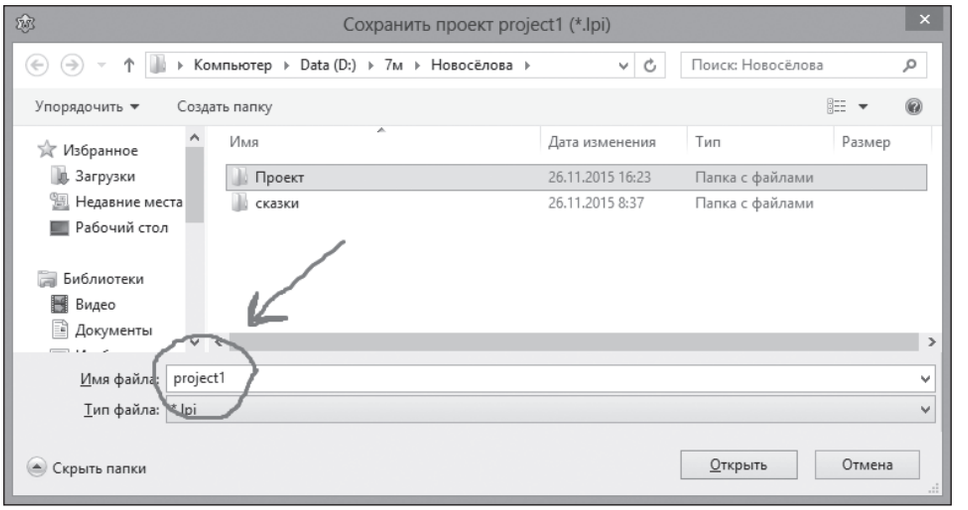


Рис. 2

- сохраните файл исходного кода проекта под именем unit1.

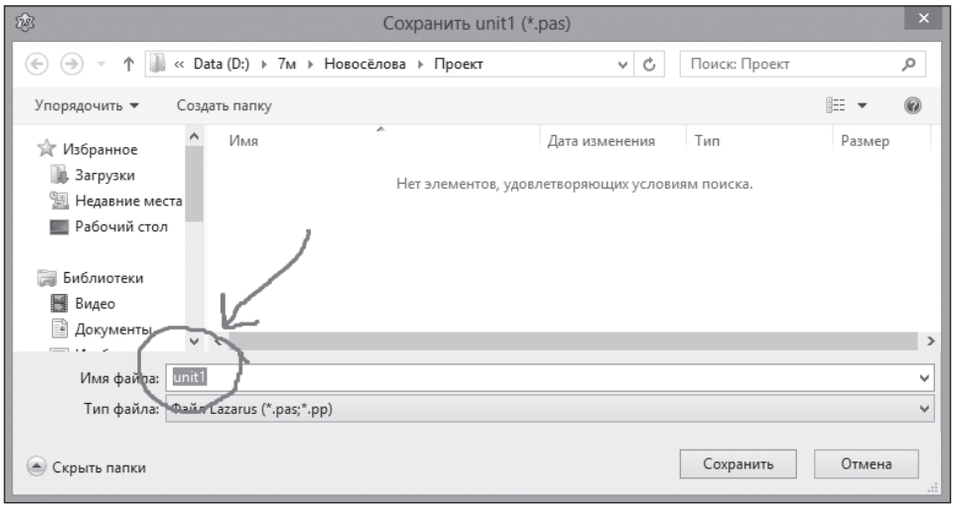


Рис. 3

2. Программирование

2.1. Линейные алгоритмы

Как запрограммировать свойство объекта? Объект.Свойство := Значение;	
Close;	Выход из проекта
Label1.color := clRed;	Изменение цвета объекта Label1 на красный
Form1.color := $00FAD1F3;	Изменение фонового цвета формы на цвет из палитры цветов
Shape1.brush.color := clGreen; Shape1.brush.color := $00FAA143;	Изменение цвета объекта Shape1 на зеленый и на цвет из палитры
Label1.caption := Edit1.text;	Отображение в текстовой метке Label1 содержания текстового окна Edit1
Label1.caption := ' Hello ' + Edit1.text;	Отображение в текстовой метке Label1 слова Hello и содержания текстового окна Edit1
Image1.visible := true;	Картинка отображается на форме при значении true и скрыта при значении false
Как перевести число в строку или строку в число?	
a := IntToStr(<число>);	Функция: переводит число в строку (переменная a типа string)

a := StrToInt(<строка>);	Функция: переводит строку в число (переменная a может быть типа integer)
Val(St, X);	Процедура: переводит строку St в число X (St: string; X: integer)
Str(X, St);	Процедура: переводит число X в строку St (St: string; X: integer)
Как запрограммировать цвет?	
x := RGBtoColor(a,b,c);	Функция: смешивает цвет из трех цветов (красного, зеленого, синего) x — типа color; a, b, c — типа byte
Как запрограммировать событие объекта?	
1. Открыть вкладку события в окне инспектора объектов. 2. Выбрать необходимое событие. 3. Открыть окно редактора исходного кода: двойным щелчком мыши или при помощи кнопки . 4. Записать программный код	
Как запрограммировать метод? Объект.метод;	
Form1.show;	Отобразить форму
Form1.hide;	Скрыть форму
Как отобразить окно сообщения?	
ShowMessage('Сообщение для вывода');	
Как задать случайное число?	
b := random;	Задает в переменную b случайное вещественное число в промежутке [0; 1)
b := random*5;	Задает в переменную b случайное вещественное число в промежутке [0; 5)
b := random(10);	Задает в переменную b случайное целое число в промежутке [0; 10)
b := random(10)–5;	Задает в переменную b случайное целое число в промежутке [–5; 5)

2.2. Алгоритмы ветвления

If <условие> Then <действие>;
 	
If <условие> Then
Begin
<действие1> ;
<действие2> ;
<действие3>
End;
	Неполное условие.
	
	Если при истинности условия необходимо выполнить несколько действий, то эти действия заключаются в операторные скобки Begin...End

If Edit1.text = 'Колобок' Then
Begin
k := k + 1;
Edit1.color := clGreen
End;
	Пример организации подсчета баллов за правильный ответ «Колобок» в поле Edit1 и изменение цвета текстового поля на зеленый

If <условие> Then <действие1>
Else <действие2>;
 	
If <условие> Then
Begin
<действие1>;
<действие2>;
<действие3>
End
Else
Begin
<действие4>;
<действие5>;
<действие6>
End;
	Полное условие.
	
	Если при истинности или ложности условия необходимо выполнить несколько действий, то эти действия заключаются в операторные скобки Begin...End.
	
	Перед словом Else никогда не ставится точка с запятой

If Edit1.text = '46' Then
Label1.caption := 'Ответ верный'
Else
Label1.caption := 'Ответ неверный';
	Пример, в котором проверяется, ввели ли в текстовое поле Edit1 число 46; если ввели именно это число, то в текстовой метке Label1 появится словосочетание «Ответ верный», если ввели другое число — «Ответ неверный»

If (<усл.1>) and (<усл.2>) and (<усл3>) Then
<действие> ;
If (<усл.1>) or (<усл.2>) or (<усл3>) Then
<действие> ;
If not (<условие>) Then <действие>;
	Составное сложное условие

If (Edit1.text = '7') and (Edit2.text = '7') and (Edit3.text = '7') Then Image1.Visible := true;	**Пример**, в котором организовано появление изображения, если в трех текстовых окнах одновременно будет стоять цифра 7

If (Edit1.text = 'репка') or (Edit1.text = 'Репка') or (Edit1.text = 'РЕПКА') Then
Label1.caption := 'Ответ верный'
Else
Label1.caption := 'Ответ неверный';
	Пример, в котором проверяют все варианты верного ответа — слова «Репка»

If <условие1> Then If <условие 2> Then <действие1> Else <действие2> Else <действие3>;	Вложенное условие
If k = 3 Then Label1.caption := 'отлично' Else If k = 2 Then Label1.caption := 'хорошо' Else Label1.caption := 'плохо';	**Пример** проверки количества баллов: если 3 балла — «отлично», если 2 балла — «хорошо», если 1 балл или 0 баллов — «плохо»

3. Практические работы

Практическая работа 1 «Флаг» (программирование цвета: Label1.color)

Задание.

Создайте проект, в котором на форме первоначально располагается кнопка с надписью «Флаг», а затем при нажатии на указанную кнопку появляется флаг Российской Федерации.

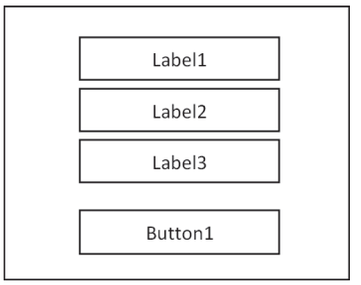


Рис. 4. Схема

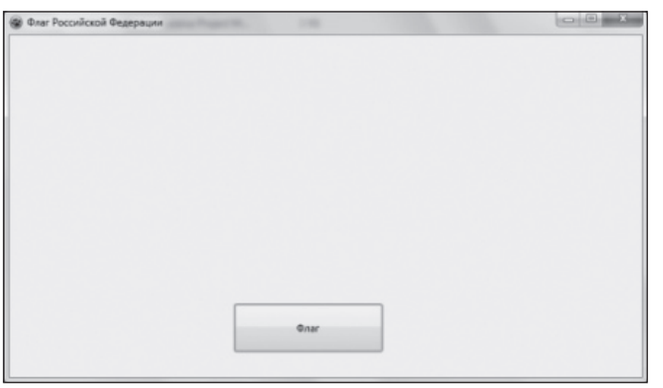


Рис. 5. Форма до выполнения исходного кода

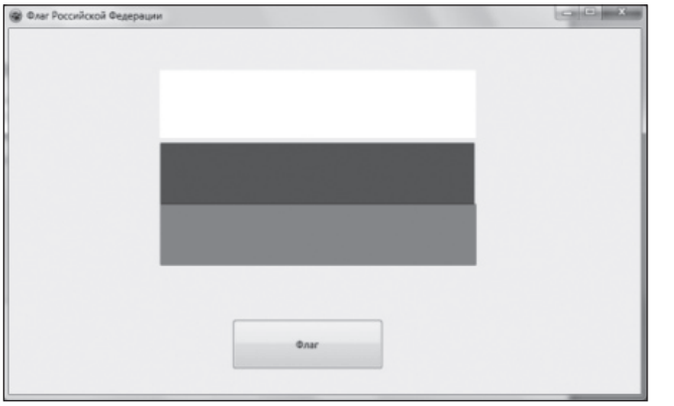


Рис. 6. Форма после выполнения исходного кода

Рекомендации к выполнению.

1. Создайте новый проект в среде Lazarus.
2. Сохраните проект в отдельную папку.
3. Расположите три объекта Label и одну кнопку Button согласно схеме.
4. Настройте свойства объектов в окне *Инспектор объектов*: свойство AutoSize — False, свойства Caption у объектов Label пустое, у объекта Button — Флаг.
5. Запрограммируйте изменение цвета текстовых меток Label в кнопке Button1, для этого два раза щелкните на кнопке — откроется окно *Редактор исходного кода*.
6. *Важно!* Код программы вписывайте там, где мигает курсор!
   Label1.color := clWhite;
   Label2.color:=clBlue;
   Label3.color:=clRed;
7. Запустите проект. Если проект успешно собран и работает верно, покажите результат учителю. Если в коде проекта присутствуют ошибки, исправьте их и запустите проект еще раз.

Практическая работа 2 «Приветствие» (программирование объекта Edit)

Задание.

Создайте проект, в котором просят ввести имя в соответствующее поле, а затем при нажатии на кнопку «Поздороваться» выводится приветственное сообщение с именем пользователя.

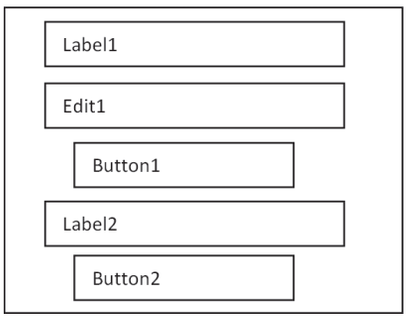


Рис. 7. Схема

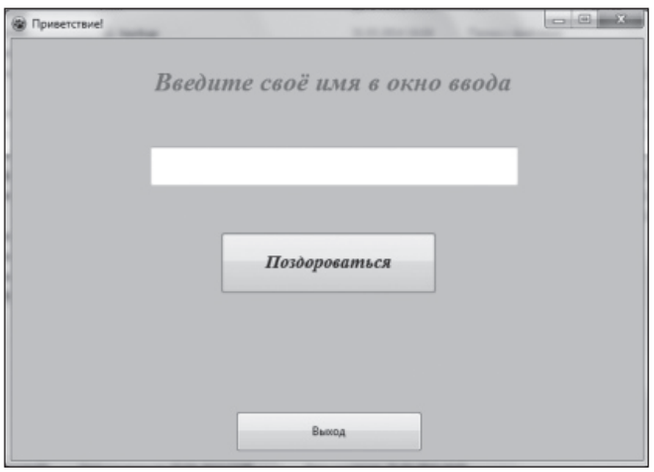


Рис. 8. Форма до выполнения исходного кода

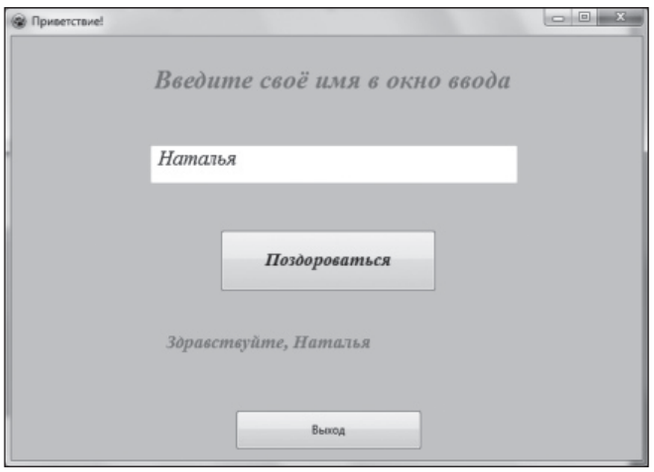


Рис. 9. Форма после выполнения исходного кода

Рекомендации к выполнению.

1. Оформите интерфейс проекта — расположите объекты на форме.
2. Запрограммируйте кнопку «Поздороваться» (см. п. 2.1).
3. Запрограммируйте кнопку «Выход».

Практическая работа 3 «Времена года» (свойство Visible)

Задание.

Создайте проект, в котором приветствуют пользователя, а затем предлагают выбрать его любимое время года. При нажатии на кнопку с названием времени года появляется соответствующее изображение.

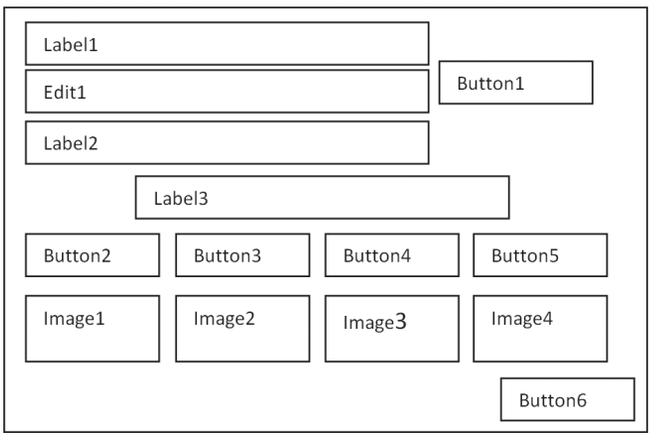


Рис. 10. Схема

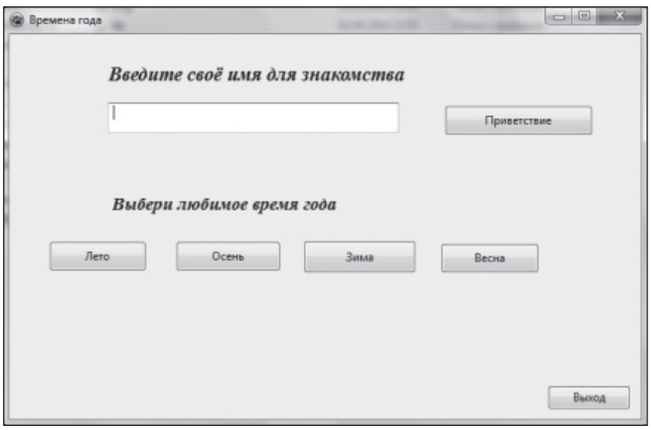


Рис. 11. Форма до выполнения исходного кода

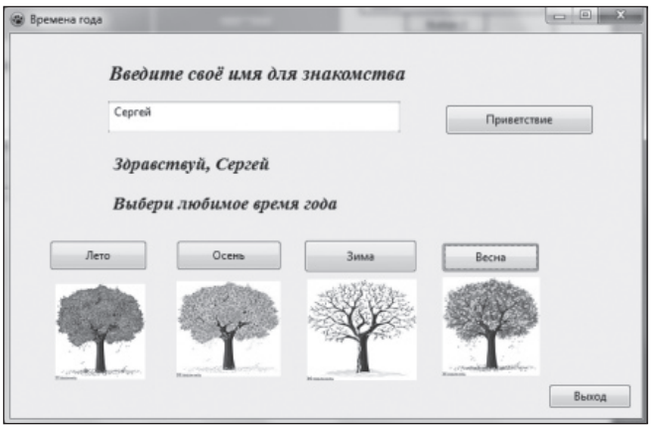


Рис. 12. Форма после выполнения исходного кода

Рекомендации к выполнению.

1. Оформите интерфейс проекта — расположите объекты на форме.
2. Запрограммируйте приветствие пользователя.

3. Запрограммируйте кнопки, отвечающие за появление изображения с указанным временем года (см. п. 2.1).

Практическая работа 4 «Калькулятор» (функции StrToInt и IntToStr, сложение)

Задание.

Создайте проект, который выполняет простейшие арифметические операции: сложение, вычитание, умножение и деление.

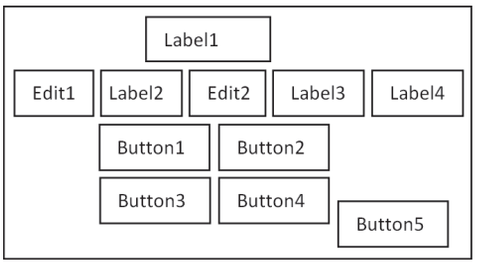


Рис. 13. Схема

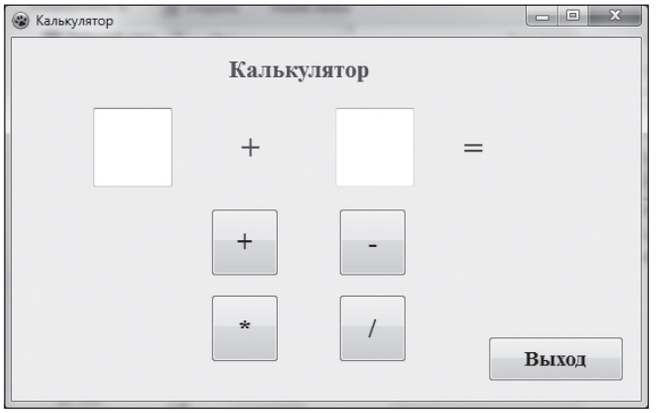


Рис. 14. Форма до выполнения исходного кода

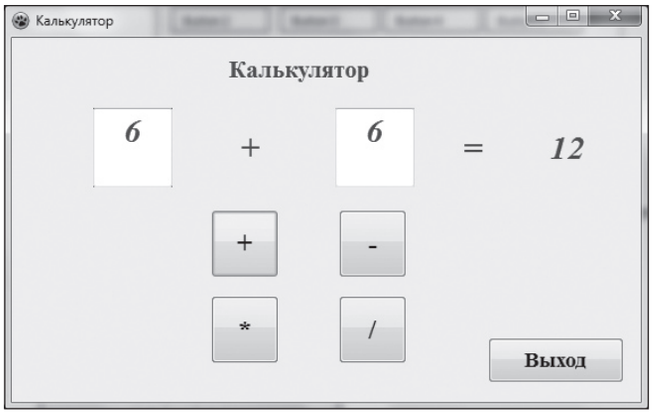


Рис. 15. Форма после выполнения исходного кода

Рекомендации к выполнению.

1. Оформите интерфейс проекта — расположите объекты на форме.
2. Запрограммируйте кнопки сложения, вычитания, умножения и деления:
   - при помощи функции StrToInt переведите строковые значения объектов Edit в числовые;
   - выполните арифметическую операцию;
   - переведите числовое значение в строковое при помощи функции IntToStr;
   - в значении объекта Label2 укажите знак арифметической операции.

Практическая работа 5 «Пример» (алгоритм ветвления)

Задание.

Создайте проект, в котором приветствуют пользователя и предлагают ему выполнить простейшую арифметическую операцию. При нажатии на кнопку «Проверить!» выводится сообщение о том, что введенный ответ верен/неверен.

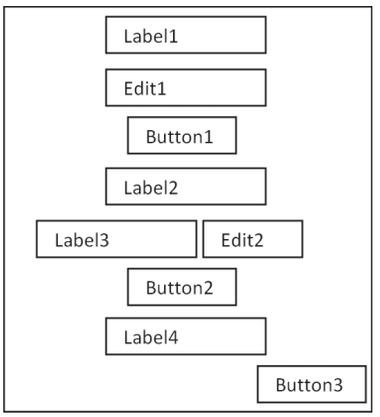


Рис. 16. Схема

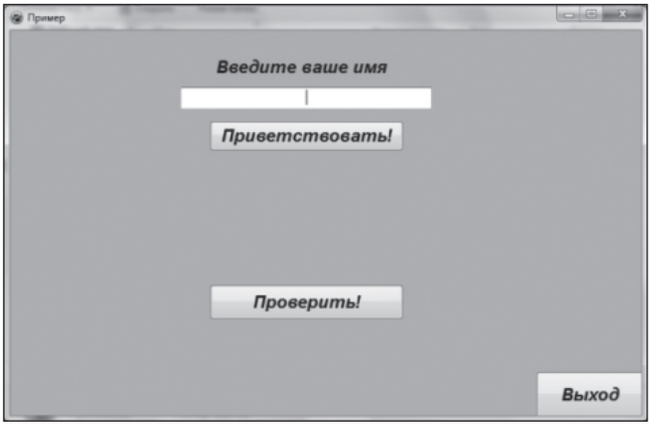


Рис. 17. Форма до выполнения исходного кода

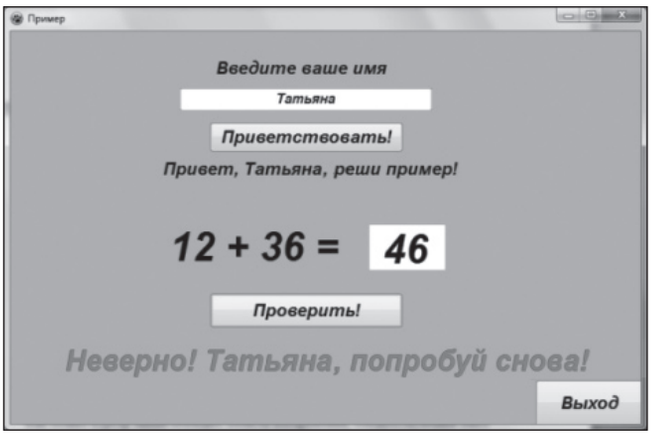


Рис. 18. Форма после выполнения исходного кода

Рекомендации к выполнению.

1. Оформите интерфейс проекта — расположите объекты на форме.
2. Запрограммируйте блок приветствия.
3. Запрограммируйте кнопку «Проверить!» на правильность ответа, введенного в текстовое поле (см. п. 2.2).

Практическая работа 6 «Пример со случайным числом» (задание случайного числа)

Задание.

Создайте проект, в котором проверяется правильность выполнения арифметической операции пользователем. При нажатии на кнопку «Новый пример» появляются новые, случайным образом заданные числа. Кнопка «Проверка» проверяет, верен ли ответ.

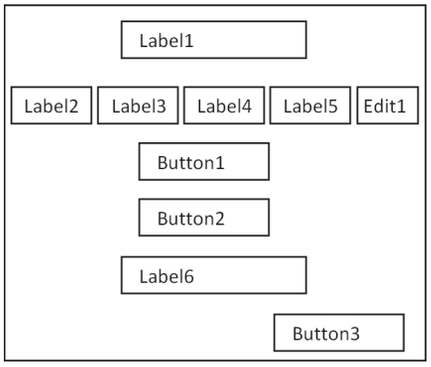


Рис. 19. Схема

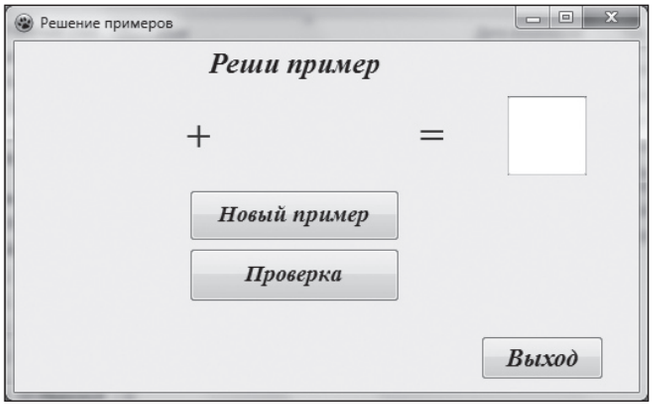


Рис. 20. Форма до выполнения исходного кода

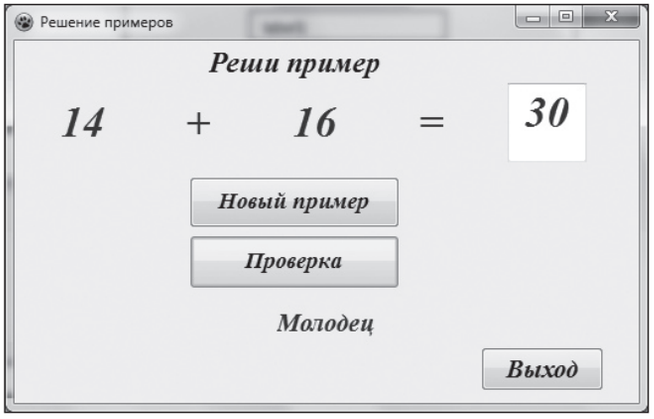


Рис. 21. Форма после выполнения исходного кода

Рекомендации к выполнению.

1. Оформите интерфейс проекта — расположите объекты на форме.
2. Запрограммируйте кнопку «Новый пример», в коде которой значению текстовой метки подбирается случайно заданное число (см. п. 2.1 «Линейный алгоритм. Как задать случайное число»).
3. Запрограммируйте кнопку «Проверка».

4. Проектные работы

Проектная работа 1 «Поймай меня» (случайное задание свойств Top, Left; объект Timer; подсчет количества)

Задание.

Создайте игру, в которой на поле в случайно заданном месте появляется прямоугольник. Пользователю необходимо поймать указателем мыши данный прямоугольник. Через определенное время прямоугольник исчезает, а пользователю выводится сообщение о набранных очках.

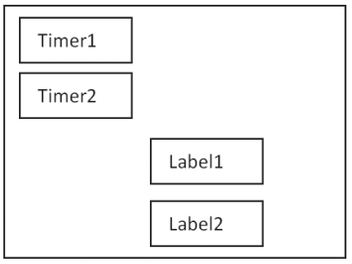


Рис. 22. Схема

Рис. 23. Форма до выполнения исходного кода

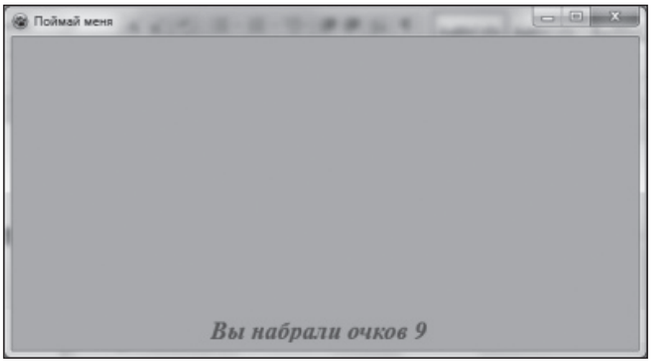


Рис. 24. Форма после выполнения исходного кода

Рекомендации к выполнению.

1. Оформите интерфейс проекта — расположите объекты на форме.
2. Запрограммируйте в коде объекта Timer1 появление прямоугольника в случайно выбранном месте поля и укажите интервал времени перемещения.
3. Запрограммируйте подсчет очков при нажатии на прямоугольник мышью.
4. Запрограммируйте в коде объекта Timer2 по истечении определенного времени выключение объекта Timer1, появление подсчитанных очков и исчезновение прямоугольника.

Проектная работа 2
«Погода»
(алгоритмы ветвления; вывод окна сообщения)

Задание.

Создайте проект, в котором организовано приветствие пользователя, а также ему необходимо ответить на три вопроса:

- «Какова температура воздуха?»
- «Ожидаются ли осадки?»
- «Какова скорость ветра?»

Если температура воздуха ниже 15 градусов, то при нажатии на кнопку «Совет» появляется окно с сообщением: «Надень куртку», если температура выше — сообщение: «Можно куртку не надевать сегодня».

Если ожидаются осадки — сообщение: «Возьми зонтик», если дождя нет — сообщение: «Зонтик можно оставить дома».

Если скорость ветра больше 20 м/с — сообщение: «Останься дома!», при другой скорости ветра — сообщение: «Все в порядке, можно идти в школу».

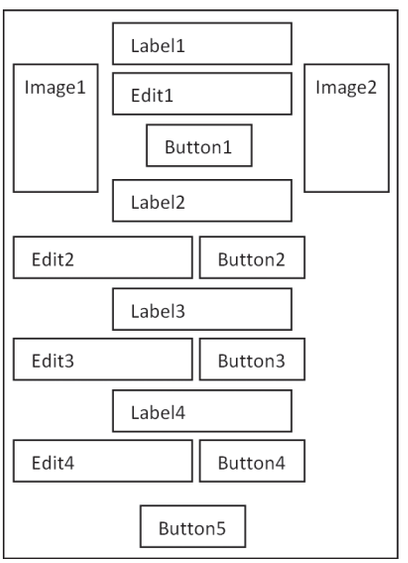


Рис. 25. Схема

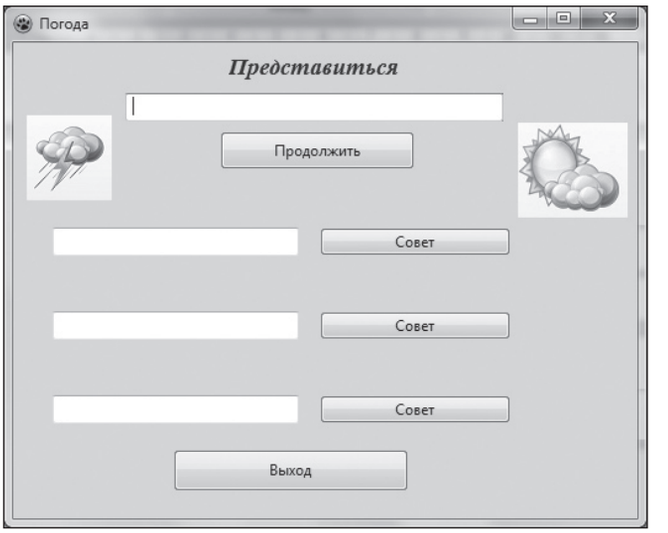


Рис. 26. Форма до выполнения исходного кода

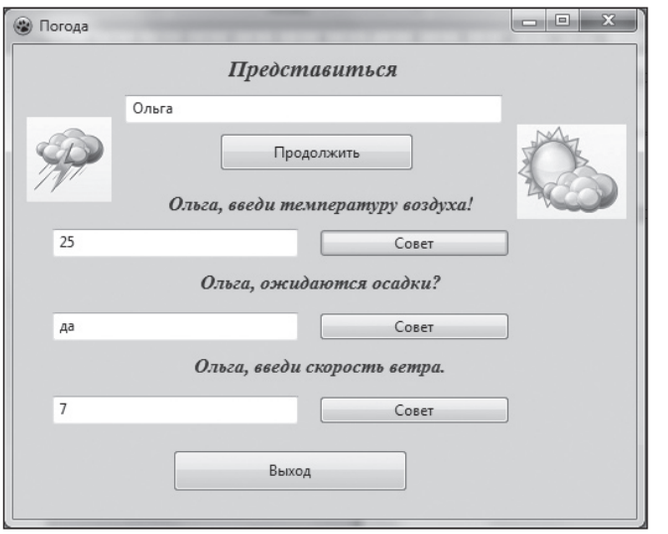


Рис. 27. Форма после выполнения исходного кода

Литературные и интернет-источники

1. *Алексеев Е. Р., Чеснокова О. В., Кучер Т. В.* Free Pascal и Lazarus: учебник по программированию. М.: ALT Linux; Издательский дом ДМК-пресс, 2010.

2. *Ачкасов В.* Программирование на Lazarus // НОУ ИНТУИТ. http://www.intuit.ru/studies/courses/13745/1221/info

3. *Гусева О. Л.* Практикум по Visual Basic. М.: Финансы и статистика, 2007.

4. *Макарова Н. В.* Информатика и ИКТ. Практикум по программированию. 10–11 классы. Базовый уровень / под ред. проф. Н. В. Макаровой. СПб.: Питер, 2007.

5. *Мансуров К. Т.* Основы программирования в среде Lazarus. http://www.freepascal.ru/

6. *Паутова А. Г.* Visual Basic. Творческое проектирование в школе и дома. В 3 ч. Ч. 1. М.: Классикс Стиль, 2003.