г. Гуково, Ростовская область

Учимся программировать в среде разработки Gambas IDE

Разработал: Фаткуллин И.И.,
учитель высшей категории

2019 г.

Ознакомление со средой разработки
и
языком программирования Gambas

г.Гуково, 2015-19г.г.

I. Ознакомление со средой разработки Gambas IDE

Gambas является средой объектно-ориентированного программирования. В процессе разработки создается *проект*, который объединяет в себе программный код и графический интерфейс.

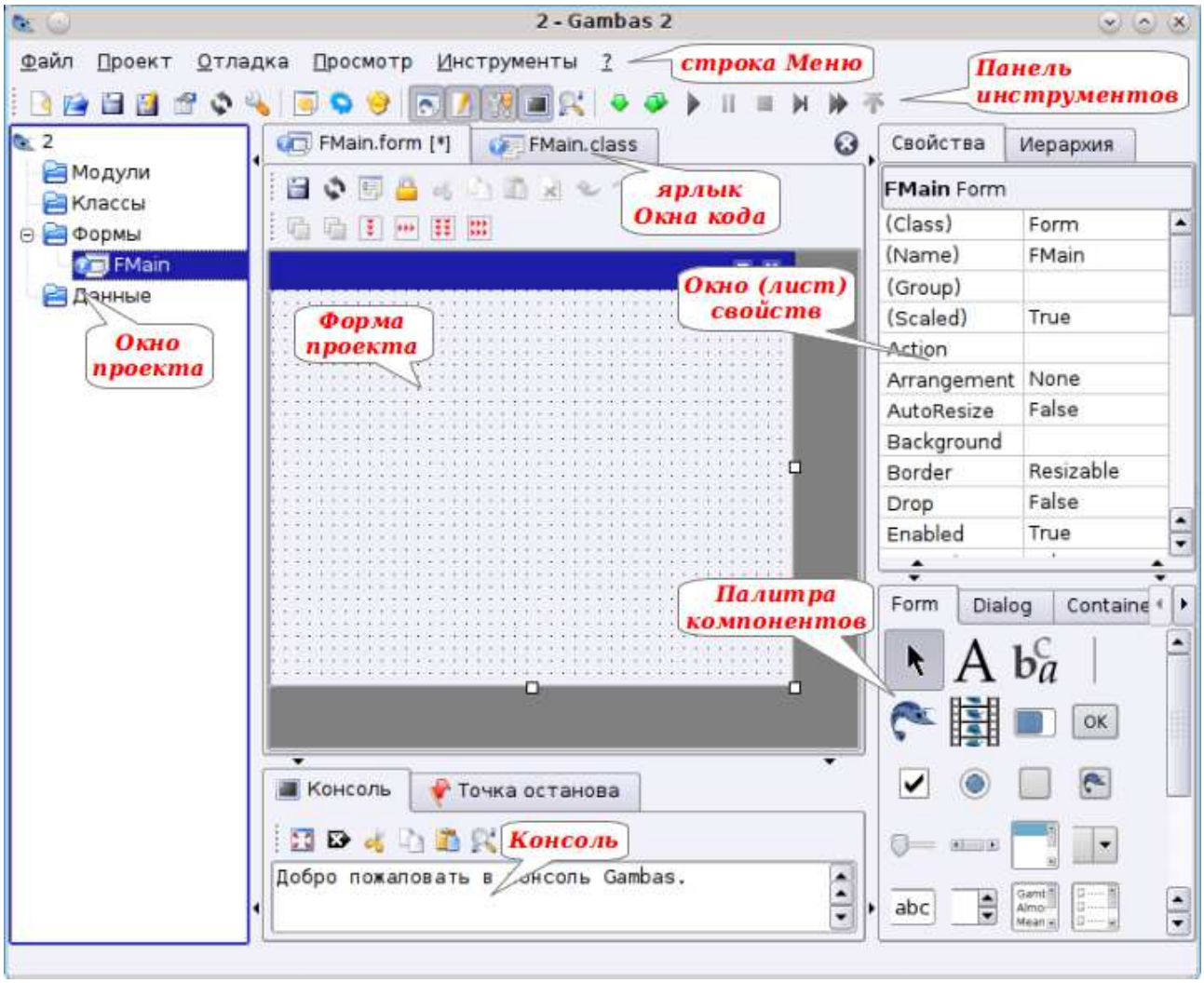
Шаги разработки проекта с графическим интерфейсом:

1. Создание *формы* и размещение на ней *элементов управления* (*компонентов*).

2. Запись *программного кода*.

3. Компиляция, отладка и сохранение *проекта*.

Форма создается при разработке проекта с графическим интерфейсом. Создание проекта начинается командой *<Главное меню [KDE]> - <Разработка> - [<IDE> -] <[Среда разработки] Gambas...>*, выбирается *«Новый проект»* (тип проекта *«Графическое приложение»*). Далее задаются каталог для его сохранения, имя проекта и заголовок. После создания проекта можно видеть следующее окно программы Gambas (*форма* проекта отображается после двойного щелчка на его значке в окне проекта):



Элементы управления на *форме* рисуются протягиванием мыши после выбора их на «*Палитре компонентов*». Затем задаются или изменяются их *параметры*.

Окно редактора кода открывается щелчком на *ярлыке Окна кода*, через *контекстное меню* или двойным щелчком на *элементе управления*, а затем записываются программные коды (*процедуры*) *событий* объекта. Сохранение проекта выполняется соответствующей командой меню или кнопкой панели инструментов. Для компиляции используется команда *Проект - Компилировать*.

1. Конструирование графического интерфейса проекта и разработка программного кода базируются на использовании *объектов*.

1.1. Классы объектов являются «*шаблонами*», определяющими наборы *свойств*, *методов* и *событий*, по которым создаются объекты. *Экземпляры класса* наследуют весь этот набор и имеют уникальное для класса имя. Набор объектов по умолчанию (для библиотек компонентов **gb**, **gb.form**, **gb.gui**) приводится в таблице:

Объект	Комментарий
Form	Форма - на ней размещаются другие компоненты
Label	Метка (надпись) - вывод строки
TextLabel	Текстовая метка - вывод нескольких строк
Button	Кнопка (командная) - пуск программы
CheckBox	Флажок - состояние: включен или выключен
RadioButton	Радиокнопка (Переключатель) - одна радиокнопка включена, остальные выключены
ToggleButton	Флажок-кнопка - как флажок или радиокнопка
ToolButton	Инструментальная кнопка - кнопка-инструмент, как флажок или радиокнопка
TextBox	Текстовое поле - ввод и вывод строк
TextArea	Область текста - ввод и вывод текста
ValueBox	Поле значений - ввод и вывод значений различных типов
PictureBox	Поле рисунка - вывод рисунка (рисовать нельзя!)
DrawingArea	Область рисования - рисование, вывод рисунка
MovieBox	Поле видео
ListBox	Поле со списком - отображение и работа со списком (записями)
ComboBox	Комбинированный список - поле со сворачиваемым списком
SpinBox	Счетчик - установка целых значений от *min* до *max*
Slider	Ползунок - плавное изменение и отображение значений
ProgressBar	Индикатор выполнения - отображение процесса
ScrollBar	Полоса прокрутки - прокручивание областей для отображения
ListView, TreeView, IconView, ColumnView, GridView, DirView, FileView, TableView	Отображение (просмотр) списка, дерева, иконки, столбцов, сетки, каталогов, файлов, таблицы

ColorChooser, DateChooser, DirChooser, FileChooser, FontChooser	Выбор цвета, даты, каталога, файла, шрифта
ColorButton	Кнопка выбора цвета
Frame, HBox, VBox, Panel, HPanel, VPanel, ScrollView, TabStrip, Expander, ListContainer, SidePanel, ToolPanel	Фрейм, Горизонтальное, Вертикальное поля, Панель, Горизонтальная, Вертикальная панели, Прокрутка-отображение, Вкладка (полоса), Расширитель, Контейнер, Скрываемая панель, Панель инструментов - на них можно размещать другие элементы
Separator, HSplit, VSplit	Разделитель, Горизонтальный, Вертикальный разделители
Embedder	Вставка
TrayIcon	Трей-иконка
Timer	Таймер - отсчет времени и запуск команд

Можно добавить другие компоненты используя команду: Проект-Свойства-Компоненты, например, компоненты для правки текста и работы со звуком:

Объект	Библиотека (набор)	Комментарий
TextEdit	**gb.qt.ext**	Редактор текста - для ввода, отображения, редактирования и оформления текста
-	**gb.sdl.sound**	Звук SDL & управление CD-ROM

1.1.2. Каждый *класс объектов* обладает определенным набором ***свойств***. *Свойства объектов* можно задать используя *лист свойств* или *программно* (*Объект.**Свойство** = Значение*). Некоторые *свойства объектов* приводятся ниже:

Свойство	Комментарий
Name	Имя объекта в программном коде
Alignment	Выравнивание (текста)
Background (или BackColor)	Цвет фона объекта
Border	Граница объекта
Enabled	Использование (включено)
Font	Шрифт
Bold	Жирный (начертание шрифта)
Size	Размер (шрифта)
Foreground (или ForeColor)	Цвет (шрифта)
Height (или H)	Высота объекта
Text	Текст, размещаемый на объекте
Visible	Видимость объекта
Width (или W)	Ширина объекта
X (или Left)	Расстояние до объекта слева
Y (или Top)	Расстояние до объекта сверху

1.1.3. *Объекты* могут использовать различные ***методы*** (*Объект.**Метод** арг1, арг2*).

Метод	Комментарий
Clear	Очистить объект
Hide	Скрыть объект
Show	Показать объект
Delete	Удалить объект
Refresh	Обновить объект (перерисовать)
Resize	Изменить размер объекта
Setfocus	Установить фокус (переместить курсор на объект)
Move	Переместить объект в новое место
Add	Добавить (строку)
Lover	Переместить объект в самый нижний слой
Raise	Переместить объект в самый верхний слой

1.1.4. Событие представляет собой *действие*, распознаваемое объектом, для которого можно задать *инструкции* (*набор команд*).

Событие	Комментарий
Activate	Активизация объекта
Change	Изменение значения
Click	Щелчок мышкой
Cursor	Установка указателя
DblClick	Двойной щелчок
MouseDown, MouseUp	Нажатие, отпускание клавиши мыши
MouseDrag	Перетаскивание мыши
MouseMove	Перемещение мыши
Open	Открытие (формы)
Timer	Окончание заданного интервала времени

1.2. В проектах Gambas можно использовать также и *системные компоненты*. Некоторые из них проводятся ниже:

Имя	Метод	Комментарий
Dialog	OpenFile	Диалоговое окно открытия файла
	SaveFile	Диалоговое окно сохранения файла
	SelectColor	Диалоговое окно выбора цвета
	SelectDirectory	Диалоговое окно выбора каталога
	SelectFont	Диалоговое окно выбора параметров шрифта
File	Load	Загрузить файл
	Save	Сохранить файл
Picture	Load	Загрузить картинку

2. Ознакомление с языком программирования Gambas

2.1. Переменные и константы

Для хранения данных в оперативной памяти используются ***переменные***. *Переменная* имеет ***имя***, ***тип*** и ***значение***. В Gambas используются следующие ***типы данных***:

Тип	Комментарий
Byte	Целый - 1 байт (0 .. 255)
Short	Целый - 2 байта (-32.768 .. +32.767)
Integer	Целый - 4 байта (-2.147.483.648 .. +2.147.483.647)
Long	Целый - 8 байт (-9.223.372.036.854.775.808.. +9.223.372.036.854.775.807)
Single	Дробный - 4 байта (-1.7014118E-38 .. +1.7014118E+38)
Float	Дробный - 8 байт (-8.98846567431105E-307 .. +8.98846567431105E+307)
Boolean	Логический - 1 байт (true/false)
Date	Дата/время - 8 байт (пример: 07/19/2005 02:20:08)
String	Строка - память для хранения указателя и всех символов строки
Variant	Произвольный - 8 байт (тип определяется автоматически)

2.1.1. Объявление переменных и констант

При объявлении *переменной* указываются *ключевое слово*, *имя переменной* и ее *тип*. *Имена переменных* обозначаются латинскими буквами и цифрами (впереди должна быть буква). Примеры объявления переменных:

```
DIM a AS Byte
PRIVATE f AS Float
STATIC b AS Boolean
PUBLIC d AS Variant
DIM s AS String
CONST c AS Integer = 400
DIM m[6] AS INTEGER 'объявляется массив из 6 элементов целого типа:
    'm[0], m[1], m[2], m[3], m[4], m[5] (нумерация начинается с нуля).
```

2.2. Операторы и функции

В программном коде (тексте программы) используются инструкции обработки данных (*операторы*, *функции*). Для выполнения математических действий применяются:

2.2.1. Арифметические операторы

Оператор	Операция	Оператор	Операция
+	Сложение	=	Присвоение
-	Вычитание	\ (или DIV)	Целочисленное деление
*	Умножение	MOD	Остаток от деления
/	Деление	^	Возведение в степень

Математические функции

Функция	Комментарий	Функция	Комментарий
Abs(n)	Модуль числа	Log(n)	Натуральный логарифм
Sgn(n)	Знак числа	Log10(n)	Десятичный логарифм
Sqr(n)	Корень	Log2(n)	Логарифм с основанием 2
Sin(x)	Синус	Rnd(n)	Случайное число (от 0 до n)
Cos(x)	Косинус	Int(n)	Округление
Tan(x)	Тангенс	Fix(n)	Отсечение дроби
Asin(x)	Арксинус	Val(s)	Преобразование строки в число
Acos(x)	Арккосинус	Str(n)	Преобразование числа в строку
Atn(x)	Арктангенс	Pi(n)	(Число ПИ) * n

2.2.2. Строковые функции и операторы

Для работы со строками используются *Строковые функции* и *операторы*:

Функция	Комментарий
Len(s)	Длина строки s (количество символов)
Left(s, k)	Подстрока (из строки s) - k символов слева
Right(s, k)	Подстрока - k символов справа
Mid(s, n, k)	Подстрока - k символов начиная с n
Asc(sim)	Код символа sim (только для кодов от 0 до 127)
Chr(kod)	Символ с кодом kod (для кодов от 0 до 127)
LCase(s)	Преобразует символы в строчные (для кодов от 0 до 127)
UCase(s)	Преобразует символы в ПРОПИСНЫЕ (для кодов от 0 до 127)
Trim(s)	Строка с удаленными пробелами слева и справа
LTrim(s)	Строка с удаленными пробелами слева
RTrim(s)	Строка с удаленными пробелами справа
Hex(n)	Представляет десятичное число n в 16-ричном виде
Bin(n)	Представляет десятичное число n в двоичном виде
InStr(s, w)	Позиция первого вхождения подстроки w в строке s
String(n, s)	Строка, состоящая из n повторов подстроки s

& - оператор сцепления (строк).

Пример: "gam" & "bas"

2.2.3. Логические операции

В ветвлениях, операторах выбора, циклах с условиями требуется записать условия с использованием *операторов сравнения*. Значения (результаты) логических операций бывают TRUE (ИСТИНА) или FALSE (ЛОЖЬ).

Оператор	Комментарий	Оператор	Комментарий
=	Равно	<>	Не равно
<	Меньше	<=	Меньше или равно
>	Больше	>=	Больше или равно

Сложные условия получаются из простых с помощью логических операций (при этом простые условия берутся в скобки и между ними ставятся знаки логических операций).

Операция	Комментарий	Операция	Комментарий
AND	И (конъюнкция)	NOT	НЕ (инверсия)
OR	ИЛИ (дизъюнкция)	XOR	Исключающее ИЛИ

Пример записи сложного условия:

((m = 5) AND NOT (n = 5)) OR ((n = 5) AND (m <> 5))

2.2.4. Функции и операторы даты и времени

При работе с *датой* и *временем* применяются соответствующие функции:

Функция	Комментарий
Date	Текущая дата (месяц, число, год)
Time	Текущее время (часы, минуты, секунды)
Timer	Время, прошедшее после запуска программы в секундах
Hour(dt)	Часы (число от 0 до 23) для даты (времени) dt
Minute(dt)	Минуты (число от 0 до 59) для даты (времени) dt
Second(dt)	Секунды (число от 0 до 59) для даты (времени) dt
Day(dt)	День месяца (число от 1 до 31) для даты dt
Month(dt)	Номер месяца (число от 1 до 12) для даты dt
Year(dt)	Год для даты dt
Weekday(dt)	День недели (число от 0 до 6) для даты dt

Операторы Date, Time используются для установления (изменения) системных даты и времени.

2.3. Диалоговые панели

Ввод и **вывод информации** можно осуществлять с помощью функций (диалоговых панелей) **InputBox** и **Message**:

InputBox(Сообщение[, Заголовок, Значение по умолчанию])

Message[.{Вид панели - «Delete», «Error», «Info», «Question» или «Warning»}] (Сообщение[, Текст на кнопке])

2.4. Ветвление и выбор

Для выполнения последовательности команд в зависимости от определенных условий и параметров, используются ветвление и выбор:

2.4.1. Условный оператор IF (ветвление)

Полная форма записи оператора **IF**:

```
IF Условие 1 THEN                    'ЕСЛИ Условие 1 верно, ТО
        Серия 1                      'выполняется Серия операторов 1
ELSE IF Условие 2 THEN               'ИНАЧЕ ЕСЛИ Условие 2 верно, ТО
        Серия 2                      'выполняется Серия операторов 2
ELSE                                 'ИНАЧЕ
        Серия 3                      'выполняется Серия операторов 3
END IF                               'КОНЕЦ (оператора) ЕСЛИ
```

Обычная форма записи оператора **IF**:

```
IF Условие THEN
        Серия 1
ELSE
        Серия 2
END IF
```

Краткая запись оператора **IF**:

```
IF Условие THEN
        Серия 1
END IF
```

2.4.2. Оператор выбора SELECT

```
SELECT CASE Переменная      'ВЫБОР ВАРИАНТА по значению Переменной
        CASE Значение 1     'ВАРИАНТ для Значения 1
                Серия 1     'выполняется Серия операторов 1
        CASE Значение 2     'ВАРИАНТ для Значения 2
                Серия 2     'выполняется Серия операторов2
        CASE Значение 3     'ВАРИАНТ для Значения 3
                Серия 3     'выполняется Серия операторов 3
        CASE ELSE           'ВАРИАНТ для других значений
                Серия 4     'выполняется Серия операторов 4
END SELECT                  'КОНЕЦ ВЫБОРА
```

2.5. Циклы (повторения)

При многократном выполнении команд используются циклы. Циклы бывают со счетчиком, когда серии команд выполняются определенное количество раз, и с условием, в которых число повторов зависит от указываемых условий.

2.5.1. Циклы FOR (со счетчиком)

```
FOR i = n0 TO n2 STEP m     'ДЛЯ Параметр = НачЗнач ДО КонечЗнач ШАГ
        Серия (операторов)  'Серия операторов, которые нужно повторять
NEXT                        'СЛЕДУЮЩИЙ повтор
```

2.5.2. Циклы с постусловием

Цикл REPEAT ... UNTIL (с постусловием)

```
REPEAT                      'ПОВТОРЯТЬ
        Серия операторов    'Серия операторов (которые нужно повторять)
UNTIL Условие               'ДО получения верного Условия
```

Цикл DO ... LOOP UNTIL (с постусловием)

```
DO                          'ДЕЛАТЬ
        Серия операторов    'выполняется Серия операторов
LOOP UNTIL Условие          'ЦИКЛ - ДО получения верного Условия
```

Цикл DO ... LOOP WHILE (с постусловием)

```
DO                          'ДЕЛАТЬ
        Серия операторов    'выполняется Серия операторов
LOOP WHILE Условие          'ЦИКЛ - ПОКА верно Условие
```

2.5.3. Циклы с предусловием

Цикл DO UNTIL ... LOOP (с предусловием)

```
DO UNTIL Условие            'ДЕЛАТЬ ДО получения верного Условия
        Серия операторов    'выполняется Серия операторов
LOOP                        'ЦИКЛ
```

Цикл DO WHILE ... LOOP (с предусловием)

```
DO WHILE Условие              'ДЕЛАТЬ ПОКА верно Условие
      Серия операторов        'выполняется Серия операторов
LOOP                          'ЦИКЛ
```

Цикл WHILE ... WEND (с предусловием - старая версия)

```
WHILE Условие                 'ПОКА верно Условие
      Серия операторов        'выполняется Серия операторов
WEND                          'КОНЕЦ ПОКА
```

2.6. Пользовательские функции и процедуры

Применение функций и процедур пользователя упрощает написание текста программы при наличии в ней повторяющихся блоков. Функция используется когда нужно получить только одно значение, в других случаях применяются процедуры. Для передачи данных в процедуры и функции могут использоваться параметры (аргументы). Примеры записи функции и процедуры:

```
FUNCTION ИмяФункции(Аргумент 1, Аргумент 2 ...) тип переменной
   Инструкции функции
   RETURN Возвращаемое значение
END

PROCEDURE ИмяПроцедуры(Параметр 1, Параметр 2, ...)
   Инструкции процедуры
END
```

2.7. Методы графики

Используя методы графики можно рисовать графические примитивы и вывести текст, рисунок на поверхности формы или в области для рисования. Методы графики:

```
Draw.Begin(ОбъектДляРисования)
    'к объекту ОбъектДляРисования применить метод Начать

Draw.Point(X, Y)
    'нарисовать текущим цветом точку с координатами (X, Y)

Draw.Line(X1, Y1, X2, Y2)
    'нарисовать линию от точки 1(X1, Y1) до точки 2(X2, Y2)

Draw.Rect(X, Y, Width, Height)
    'нарисовать прямоугольник(левый верхний угол: X, Y, Ширина, Высота)

Draw.Ellipse(X, Y, Width, Height[, Start, End])
    'нарисовать эллипс(X, Y, Ширина, Высота[, НачалоДуги, КонецДуги])

Draw.Circle(X, Y, Radius[, Start, End])
    'нарисовать окружность(X, Y, Радиус[, НачалоДуги, КонецДуги])

Draw.Polyline([X1, Y1, X2, Y2, ..., Xn, Yn])
    'нарисовать ломаную (указать массив координат точек)

Draw.Polygon([X1, Y1, X2, Y2, ..., Xn, Yn])
    'нарисовать многоугольник (задать массив координат точек вершин)

Draw.Picture(Picture, X, Y[, Width, Height, SrcX, SrcY, SrcWidth, SrcHeight])
    'вывести рисунок(Рисунок, X, Y
    '[, Ширина, Высота, Параметры вырезания полей рисунка])*

Draw.Text(Text, X, Y)
    'вывести текст в заданной точке

Draw.End
    'метод окончить рисование
```

*Пример вывода рисунка:
Draw.Picture(Picture.Load("Picture1.png"), 10, 10, 400, 300)

Рисунок также можно вывести инструкцией:
Draw.Image(Image, X, Y[, Width, Height, SrcX, SrcY, SrcWidth, SrcHeight]),
например: Draw.Image(Image.Load("Image1.png"), 0, 0)

Перед рисованием можно задать свойства получаемой фигуры: ForeColor - цвет контура, FillColor - цвет заливки, FillStyle - стиль заливки (0 - нет заливки, 1 - сплошная заливка, 2-14 - штриховки). Примеры записи:

Draw.ForeColor = 255 'синий цвет

Draw.FillColor =&H00FF00& 'зеленый цвет

Draw.FillStyle = 1 'сплошная заливка

Свойство LineWidth задает толщину линии, LineStyle указывает стиль линии (1 - сплошная, 2 - штриховая, 3 - пунктирная, 4-5 - штрих-пунктирные линии). Например:

Draw.LineWidth = 4

Draw.LineStyle = 2

Цвет можно задать в виде шестнадцатеричного (в RGB-формате) или десятичного числа:

Цвет	**В 16-чном виде**	**В 10-чном виде**
Черный	&H0&	0
Красный	&HFF0000&	16711680 (= 255 * 2^{16} = 255 * 65536)
Зеленый	&HFF00&	65280 (= 255 * 2^{8} = 255 * 256)
Синий	&HFF&	255
Желтый	&HFFFF00&	16776960 (=255*2^{16}+255*2^{8}=255*65536+255*256)
Голубой	&HFFFF&	65535 (= 255 * 2^{8} + 255 = 255 * 256 +255)
Пурпурный	&HFF00FF&	16711680 (= 255 * 2^{16} + 255 = 255 * 65536 + 255)
Белый	&HFFFFFF&	16777215 (= 255 * 65536 + 255 * 256 + 255)

Остальные цвета можно получить смешивая Красную (Red), Зеленую (Green) и Синию (Blue) составляющие в других (нужных) соотношениях.

2.8. Работа с файлами

Чтобы читать данные из файла, добавлять и записывать информацию в файл используются файловые переменные, которые нужно связать с файлами. Фрагмент программы для работы с файлами с использованием файловых переменных приводится ниже:

```
PUBLIC SUB Main()
  DIM inputLine AS String
  ' Loop until the end of the standard input file stream
  WHILE NOT Eof(File.In)
    ' Read a line from standard input. This is the same as:
    ' LINE INPUT inputLine
    LINE INPUT #File.In, inputLine
    ' Print to standard output. This is the same as:
    ' PRINT inputLine
    PRINT #File.Out, inputLine
    ' Print to standard error
    PRINT #File.Err, inputLine
  WEND
END
```

С файлами также можно работать используя методы File.Load, File.Save, но нужно учитывать, что при этом действие выполняется над всем объемом данных одновременно. Эти методы хороши, когда в проекте используется элемент Область (текста, рисования) (TextArea, DrawingArea):

```
PUBLIC SUB ButtonOpen_Click()
  Dialog.Filter = ["*.txt", "Text Files"]
  IF Dialog.OpenFile() THEN RETURN
  TextAreaEdit.Text = File.Load(Dialog.Path)
CATCH
  Message.Info(Error.Text)
END

PUBLIC SUB ButtonSave_Click()
  Dialog.Filter = ["*.txt", "Text Files"]
  IF Dialog.SaveFile() THEN RETURN
  File.Save(Dialog.Path, TextAreaEdit.Text)
CATCH
  Message.Info(Error.Text)
END
```

Оператор (функция)	**Комментарий**
PIPE	Открывает именованный канал для чтения или записи и создает для него поток
OPEN	Открывает файл для чтения или записи и создает для него поток
CLOSE	Закрывает поток
Lof	Возвращает длину потока
Eof	Возвращает, достигнут ли конец файла
LOCK	Блокирует открытый поток
UNLOCK	Разблокирует открытый поток
INPUT	Считывает данные из текстового потока
LINE INPUT	Читает строки из текстового потока
INPUT FROM	Перенаправляет стандартный ввод
READ	Считывает двоичные данные из потока
SEEK	Изменяет позицию указателя файла потока
Seek	Получает позицию указателя файла потока
PRINT	Выводит выражение в поток
OUTPUT TO	Перенаправляет стандартный вывод
WRITE	Записывает двоичные данные в поток
ERROR	Направляет выражение в стандартный вывод ошибок
ERROR TO	Перенаправляет стандартный вывод ошибок
FLUSH	Сбрасывает выходные данные буферизованного потока

Разработка программ в Gambas IDE

г.Гуково, 2015-19г.г.

1. Запуск и начало работы

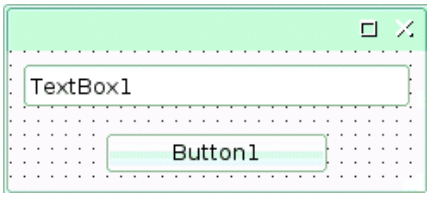
1.1. Запустите программу **Gambas** (<Главное меню> - <Разработка> - [<IDE> -] <[Среда разработки] Gambas...>), выберите "Новый проект", тип проекта: "Графическое приложение". Далее в Домашней папке ("Home") создайте (Create directory) папку "*Projects*" (и выберите её); (далее) задайте имя проекта - "*project1*", заголовок - "*Первая программа*". Двойным щелчком мыши отобразите **Форму проекта** (FMain); кнопкой на панели инструментов - **Палитру компонентов**.

1.2. Уберите и восстановите **окно Проекта**, **окно Формы**, **лист Свойств**, **окно (редактор) Кода**, **Палитру компонентов**. Просмотрите содержимое вкладок *Палитры компонентов*.

1.3. Просмотрите содержимое меню **Файл**, **Просмотр**, **Инструменты**, **Проект**, **Отладка**. Закройте программу Gambas. Изучите меню **<Проект>** - **<Свойства>** и **<Инструменты>** - **<Предпочтения>**.

2. Разработка простейшей программы

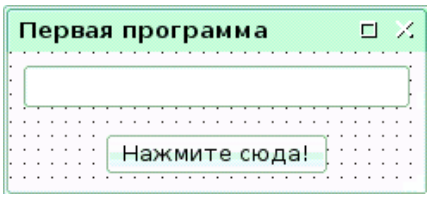
2.1. Запустите программу Gambas и откройте проект «*project1*». Откройте форму и создайте на ней «Текстовое поле» (TextBox1) и «Командную кнопку» (Button1).



2.2. Задайте нужные параметры свойств форме, текстовому полю и командной кнопке:

(Name)	FMain
Text	Первая программа

(Name)	TextBox1	(Name)	Button1
Text		**Text**	Нажмите сюда!



2.3. Запишите процедуру для события «щелчок по командной кнопке»:

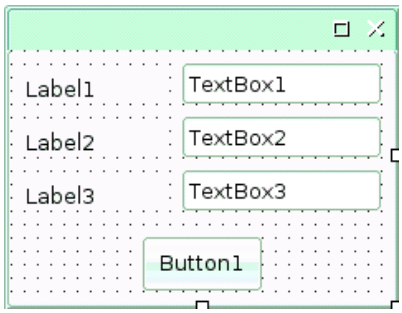
```
' Gambas class file

PUBLIC SUB Button1_Click()
  FMain.Caption = "Ура, получилось!"
  TextBox1.Text = "Моя первая программа"
  Button1.Text = "Спасибо!"
END
```

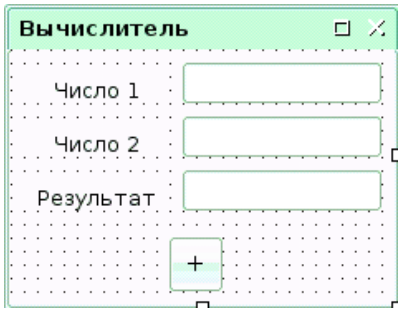
2.4. Запустите проект на исполнение, при необходимости выполните отладку.

2.5. Сохраните изменения проекта. При необходимости соберите исполняемую программу «project1» (Проект - Собрать - Запускаемые [Executable]) и проверьте ее работу. Закройте программу Gambas.

3. Присвоение, сложение чисел (программа Вычислитель — начало)

3.1. Запустите программу Gambas и в папке «Projects» создайте Новый проект - Графическое приложение, Имя проекта «calculator», заголовок - «Вычислитель». Разместите на форме 3 метки (Label1-3), 3 текстовых поля, командную кнопку и задайте их параметры.





(Name)	FMain
Text	Вычислитель

(Name)	Label1
Alignment	Center
Text	Число 1

(Name)	TextBox1
Alignment	Center
Text	

(Name)	Button1
Text	+

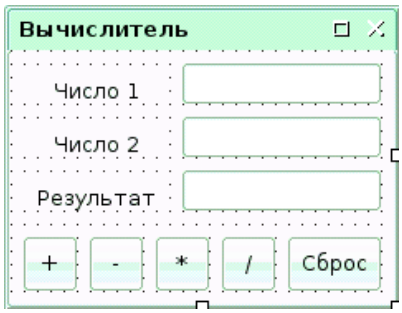
3.2. Запишите процедуру для командной кнопки:

```
PUBLIC SUB Button1_Click()
  TextBox3.Text = Val(TextBox1.Text) + Val(TextBox2.Text)
END
```

3.3. Запустите проект на исполнение, при необходимости выполните отладку. Сохраните изменения проекта и закройте программу.

4. Математические операторы (Вычислитель - завершение)

4.1. Запустите программу Gambas и Откройте проект «calculator». Создайте ещё 4 командные кнопки и задайте их параметры.



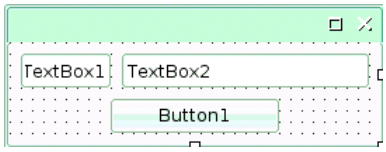
4.2. Запишите процедуры:

```
PUBLIC SUB Button2_Click()
  TextBox3.Text = Val(TextBox1.Text) - Val(TextBox2.Text)
END

PUBLIC SUB Button3_Click()
  TextBox3.Text = Val(TextBox1.Text) * Val(TextBox2.Text)
END

PUBLIC SUB Button4_Click()
  IF Val(TextBox2.Text) <> 0 THEN
    TextBox3.Text = Val(TextBox1.Text) / Val(TextBox2.Text)
  ELSE
    Message("На ноль делить нельзя!", "OK")
    TextBox3.Text = ""
  END IF
END

PUBLIC SUB Button5_Click()
  TextBox1.Text = ""
  TextBox2.Text = ""
  TextBox3.Text = ""
END
```

4.3. Сохраните изменения проекта. Запустите программу на исполнение и проверьте её работу. Закройте программу.

5. Условный оператор (Определитель чисел)

5.1. Запустите программу Gambas и в папке «Projects» создайте Новый проект - Графическое приложение, Имя проекта «defines», заголовок - «Определитель чисел». Разместите на форме 2 текстовых поля, командную кнопку и задайте их параметры.



5.2. Запишите процедуру для события кнопки:

```
PUBLIC SUB Button1_Click()
  IF Val(TextBox1.Text) > 0 THEN
    TextBox2.Text = "Положительное число"
  ELSE IF Val(TextBox1.Text) < 0 THEN
    TextBox2.Text = "Отрицательное число"
  ELSE
    TextBox2.Text = "Число равно 0"
  END IF
END
```

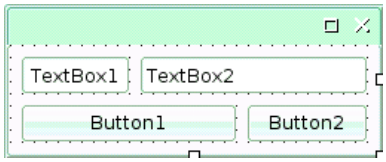
5.3. Сохраните изменения проекта. Запустите программу на исполнение и проверьте её работу. Закройте программу.

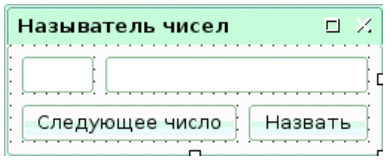
6. Оператор выбора (Называтель чисел)

6.1. Запустите программу Gambas и в папке «Projects» создайте новый проект (Графическое приложение) - имя проекта «named», заголовок - «Называтель чисел». Разместите на форме 2 текстовых поля, 2 командные кнопки и задайте их параметры.

6.2. Объявите переменную, запишите процедуры для кнопок:

```
STATIC n AS Byte

PUBLIC SUB Button1_Click()
  n = n + 1
  TextBox1.Text = n
  TextBox2.Text = ""
END

PUBLIC SUB Button2_Click()
  SELECT CASE TextBox1.Text
    CASE 1
      TextBox2.Text = "Один"
    CASE 2
      TextBox2.Text = "Два"
    ...
    CASE 9
      TextBox2.Text = "Девять"
    CASE 10
    ...
    CASE ELSE
      TextBox2.Text = "Не знаю"
  END SELECT
END

PUBLIC SUB TextBox1_Change()
  IF IsNumber(Val(TextBox1.Text)) THEN n = Val(TextBox1.Text)
END
```

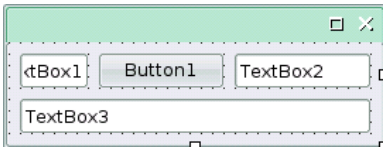


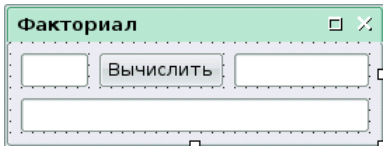


6.3. Сохраните изменения проекта. Запустите программу на исполнение и проверьте её работу. Закройте программу.

7. Циклы (Факториал)

7.1. В программе Gambas создайте проект с именем «factorial», заголовок - «Факториал». Разместите на форме 3 текстовых поля, командную кнопку и задайте их параметры.





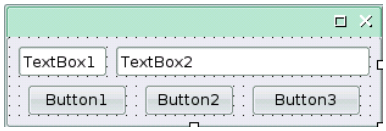
7.2. Объявите переменные и запишите процедуру для события кнопки,:

```
PUBLIC SUB Button1_Click()
  DIM a, i AS Byte
  DIM c AS Long
  DIM s AS String
  TextBox2.Text = ""
  TextBox3.Text = ""
  IF NOT IsNumber(Val(TextBox1.Text)) THEN
    TextBox3.Text = "Это не число!"
  ELSE
    a = TextBox1.Text
    c = 1
    s = 1
    FOR i = 2 TO a
      c = c * i
      s = s & "*" & i
    NEXT
    TextBox2.Text = c
    TextBox3.Text = s
  END IF
END
```

7.3. Сохраните изменения проекта. Запустите программу на исполнение и проверьте её работу. Закройте программу.

8. Циклы и массивы (Квадраты чисел)

8.1. В программе Gambas создайте проект с именем «squares», заголовок - «Квадраты чисел». Разместите на форме 2 текстовых поля, 3 командные кнопки и задайте их параметры.



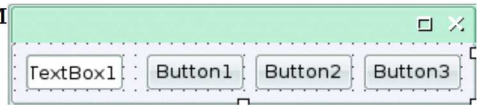
8.2. Объявите переменные, запишите процедуры для кнопок:

```
STATIC B[101] AS Integer
PUBLIC SUB Form_Open()
  Button2.Enabled = FALSE
  Button3.Enabled = FALSE
END

PUBLIC SUB Button1_Click()
  DIM i AS Integer
  FOR i = 1 TO 100
    B[i] = i ^ 2
  NEXT
  Button1.Enabled = FALSE
  Button2.Enabled = TRUE
  Button3.Enabled = TRUE
END

PUBLIC SUB Button2_Click()
  DIM i AS Integer
  FOR i = 1 TO 100
    B[i] = 0
  NEXT
  TextBox1.Text = ""
  TextBox2.Text = ""
  Button1.Enabled = TRUE
  Button2.Enabled = FALSE
  Button3.Enabled = FALSE
END

PUBLIC SUB Button3_Click()
  DIM s AS String
  s = TextBox1.Text
  IF NOT IsNumber(Val(s)) THEN
    Message("Это не число!", "OK")
  ELSE IF Val(s) < 1 OR Val(s) > 100 THEN
    Message("Должно быть число от 1 до 100", "OK")
  ELSE
    TextBox2.Text = B[Val(s)]
  ENDIF
END
```

8.3. Сохраните изменения проекта. Запустите программу на исполнение и проверьте её работу. Закройте программу.

9. Создание программы со случайными числами

9.1. В программе Gambas создайте проект с именем «random», заголовок - «Случайное число». Разместите на форме текстовое поле, 3 командные кнопки и задайте их параметры.



9.2. Запишите процедуры для кнопок:

```
PUBLIC SUB Button1_Click()
  TextBox1.Text = Int(Rnd * 10)
END

PUBLIC SUB Button2_Click()
  TextBox1.Text = Int(Rnd * 100)
END

PUBLIC SUB Button3_Click()
  TextBox1.Text = Int(Rnd * 500)
END
```

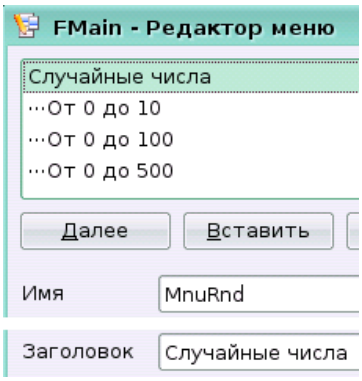
9.3. Сохраните изменения проекта. Запустите программу на исполнение и проверьте её работу.

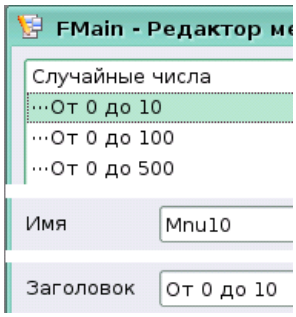
10. Создание меню программы

10.1. На форме проекта «random» создайте (через контекстное меню) меню программы.

10.2. Запишите процедуры для меню:

```
PUBLIC SUB Button1_Click()
  TextBox1.Text = Int(Rnd * 10)
END

PUBLIC SUB Button2_Click()
  TextBox1.Text = Int(Rnd * 100)
END

PUBLIC SUB Button3_Click()
  TextBox1.Text = Int(Rnd * 500)
END SUB

PUBLIC SUB Mnu10_Click()
  Button1_Click
END

PUBLIC SUB Mnu100_Click()
  Button2_Click
END

PUBLIC SUB Mnu500_Click()
  Button3_Click
END
```

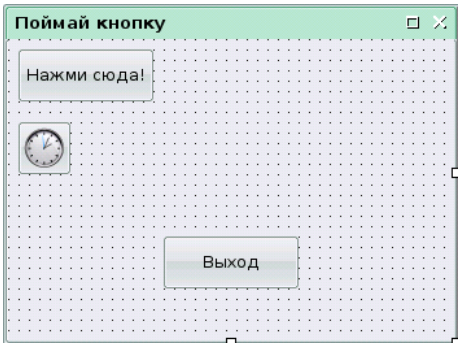




10.3. Сохраните изменения проекта. Запустите программу на исполнение и проверьте её работу. Закройте программу.

11. Создание программы «Поймай кнопку»

11.1. Создайте проект с именем «catch_key», заголовок - «Поймай кнопку». Разместите на форме 2 командные кнопки и таймер, задайте их параметры.



(Name)	Timer1
(Group)	
Delay	400
Enabled	True

(Name)	FMain
Height	232
Text	Поймай кнопку
Width	344

(Name)	Button1
Height	40
Text	Нажми сюда!
Width	104
X	8
Y	8

11.2. Запишите программный код:

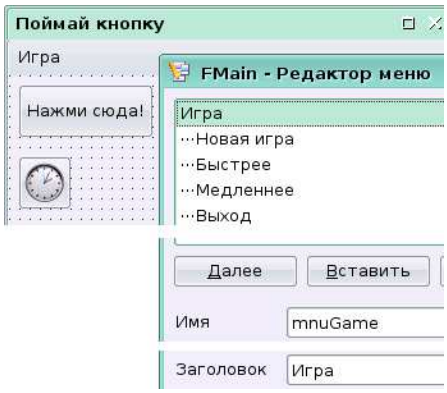
```
STATIC n AS Byte

PUBLIC SUB Timer1_Timer()
  n = Int(Rnd * 9)
  Button1.Left = 8 + 112 * (n DIV 3)
  Button1.Top = 8 + 48 * (n MOD 3)
END
```

```
PUBLIC SUB Button1_MouseDown()
  Timer1.Enabled = FALSE
  Button1.Text = "Конец игры!"
END

PUBLIC SUB Button2_Click()
  QUIT
END
```

11.3. Запустив на исполнение, проверьте работоспособность проекта. При необходимости измените параметры элементов. Сохраните проект.

12. Усовершенствование программы «Поймай кнопку»

12.1. На форме проекта «catch_key» создайте меню программы.



Имя	mnuNewGame
Группа	
Заголовок	Новая игра

Имя	mnuExit
Группа	
Заголовок	Выход

Имя	mnuQuick
Группа	
Заголовок	Быстрее

Имя	mnuSlow
Группа	
Заголовок	Медленнее

12.2. В программном коде добавьте процедуры для пунктов меню:

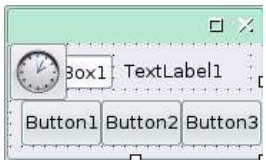
```
STATIC n AS Byte

PUBLIC SUB Timer1_Timer()
  n = Int(Rnd * 9)
  Button1.Left = 8 + 112 * (n DIV 3)
  Button1.Top = 8 + 48 * (n MOD 3)
END

PUBLIC SUB Button1_MouseDown()
  Timer1.Enabled = FALSE
  Button1.Text = "Конец игры!"
END

PUBLIC SUB Button2_Click()
  QUIT
END
```

```
PUBLIC SUB mnuNewGame_Click()
  Timer1.Enabled = TRUE
  Button1.Text = "Нажми сюда!"
END

PUBLIC SUB mnuQuick_Click()
  Timer1.Delay = Timer1.Delay * 0.7
END

PUBLIC SUB mnuSlow_Click()
  Timer1.Delay = Timer1.Delay * 1.4
END

PUBLIC SUB mnuExit_Click()
  QUIT
END
```

12.3. Проверьте работоспособность и сохраните проект.

13. Создание программы «Секунды»

13.1. В программе Gambas создайте проект с именем «seconds», заголовок - «Секунды». Разместите на форме текстовое поле, текстовую метку (TextLabel1), таймер (Timer1), 3 командные кнопки и задайте их параметры.



13.2. Объявите переменную, запишите процедуры для событий таймера и кнопок:

(Name)	Timer1
(Group)	
Delay	1000
Enabled	False

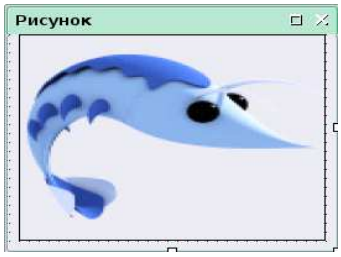
```
STATIC t AS Integer

PUBLIC SUB Timer1_Timer()
  t = t + 1
  TextBox1.Text = t
END

PUBLIC SUB Button1_Click()
  Timer1.Enabled = TRUE
END

PUBLIC SUB Button2_Click()
  Timer1.Enabled = FALSE
END

PUBLIC SUB Button3_Click()
  t = 0
  TextBox1.Text = t
END
```

13.3. Сохраните изменения проекта. Запустите программу на исполнение и проверьте её работу. Закройте программу.

14_1. Работа с графикой - создание программы «Рисунок»

14_1.1. В программе Gambas создайте проект с именем «picture», заголовок - «Рисунок». Разместите на форме поле рисунка (PictureBox1) и задайте параметры.

(Name)	PictureBox1
Stretch	True



14_1.2. Запишите код программы:

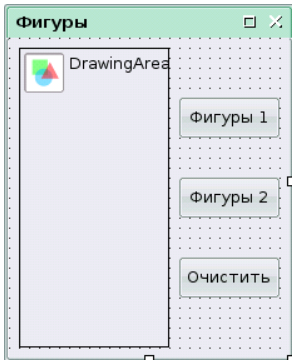
```
PUBLIC SUB PictureBox1_MouseDown()
   PictureBox1.Picture = Picture.Load("12.png")
END
```

14_1.3. Создайте файл (рисунок) «12.png» и поместите его в каталог проекта («Projects/picture»). Щелчком мыши на поле рисунка проверьте работоспособность, и сохраните проект.

14_2. Создание программы «Фигуры»

14_2.1. Создайте проект с именем «figures», заголовок - «Фигуры», состоящей из формы, области рисования (DrawingArea1), 3-х командных кнопок и задайте их параметры.

(Name)	DrawingArea1
Height	240
Width	120



14_2.2. Запишите код программы:

```
PUBLIC SUB Button1_Click()
  DRAW.BEGIN(DrawingArea1)
  DRAW.FillStyle = 1
  DRAW.LineWidth = 3
  DRAW.ForeColor = 255 * 256 * 256
  DRAW.FillColor = 255
  DRAW.FillStyle = 1
  DRAW.ELLIPSE(0, 20, 120, 80)
  DRAW.ForeColor = 255 * 256
  DRAW.LINE(0, 60, 60, 20)
  DRAW.LINE(60, 20, 120, 60)
  DRAW.LINE(120, 60, 60, 100)
  DRAW.LINE(60, 100, 0, 60)
  DRAW.END
END

PUBLIC SUB Button2_Click()
  DRAW.BEGIN(DrawingArea1)
  DRAW.LineWidth = 3
  DRAW.FillStyle = 1
  DRAW.ForeColor = 255
  DRAW.FillColor = 255 * 256 * 256 + 255 * 256
  DRAW.RECT(10, 130, 100, 100)
  DRAW.ForeColor = 255 * 256 * 256 + 255
  DRAW.FillColor = 255 * 256
  DRAW.Circle(60, 180, 50)
  DRAW.END
END

PUBLIC SUB Button3_Click()
  DrawingArea1.Refresh
END
```

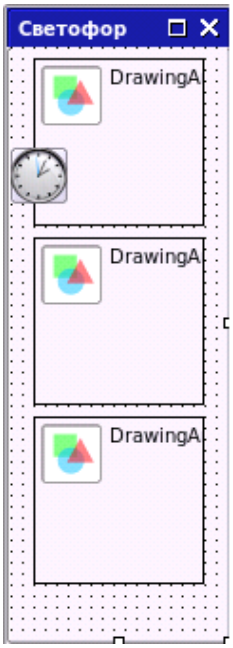
14_2.3. Нажимая на кнопки, проверьте работоспособность проекта и сохраните его.

15. Создание программы «Светофор»

15.1. В программе Gambas создайте проект с именем «light_signal», заголовок - «Светофор». Разместите на форме 3 области рисования (DrawingArea1-3) и таймер. Задайте их параметры.

(Name)	Timer1
(Group)	
Delay	500
Enabled	True

(Name)	DrawingArea1
Height	120
Width	120



15.2. Объявив переменную, запишите процедуры (пользовательскую и для таймера):

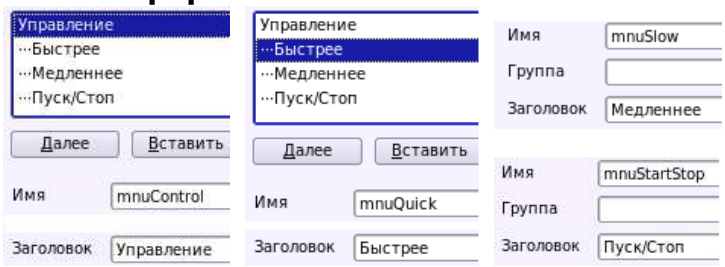
```
STATIC t AS Byte

PUBLIC PROCEDURE OnLight(ob AS Object, on AS Boolean, cl AS Integer)
  IF on THEN
    DRAW.BEGIN(ob)
    DRAW.FillStyle = 1
    DRAW.FillColor = cl
    DRAW.Circle(60, 60, 60)
    DRAW.END
  ELSE
    ob.Refresh
  ENDIF
END
```

```
PUBLIC SUB Timer1_Timer()
  SELECT CASE t
    CASE 0
      OnLight(DrawingArea1, 1, &FF0000)
      OnLight(DrawingArea2, 0, &FF00)
    CASE 8
      OnLight(DrawingArea2, 1, &FF00)
    CASE 12
      OnLight(DrawingArea1, 0, &FF0000)
      OnLight(DrawingArea2, 0, &FF00)
      OnLight(DrawingArea3, 1, &FF000)
    CASE 23
      OnLight(DrawingArea2, 1, &FF00)
      OnLight(DrawingArea3, 0, &FF000)
  END SELECT
  t = (t + 1) MOD 27
END
```

15.3. Запустив на исполнение, проверьте работоспособность проекта. При необходимости измените параметры элементов. Сохраните проект.

16. Совершенствование программы «Светофор»

16.1. На форме проекта «light_signal» создайте (через контекстное меню) меню программы.



16.2. Добавьте инструкции для события таймера и запишите процедуры для меню:

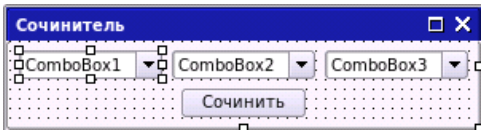
```
PUBLIC SUB Timer1_Timer()
  ...
    CASE 17
      OnLight(DrawingArea3, 0, &FF000)
    CASE 18
      OnLight(DrawingArea3, 1, &FF000)
    CASE 20
      OnLight(DrawingArea3, 0, &FF000)
    CASE 21
      OnLight(DrawingArea3, 1, &FF000)
  ...
END
```

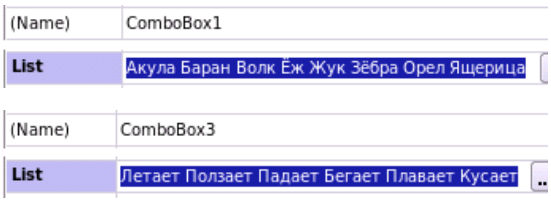
```
PUBLIC SUB mnuQuick_Click()
  Timer1.Delay = Timer1.Delay * 0.7
END

PUBLIC SUB mnuSlow_Click()
  Timer1.Delay = Timer1.Delay * 1.4
END

PUBLIC SUB mnuStartStop_Click()
  IF Timer1.Enabled = TRUE THEN
    Timer1.Enabled = FALSE
  ELSE
    Timer1.Enabled = TRUE
  ENDIF
END
```

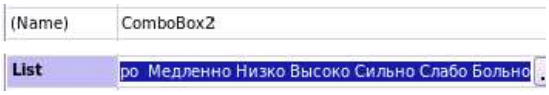
16.3. Проверьте работоспособность и сохраните проект.

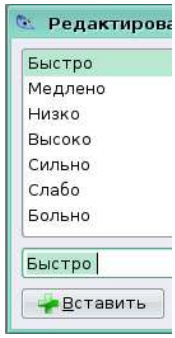
17. Программа «Сочинитель»

17.1. Создайте проект с именем «writer», заголовок - «Сочинитель», состоящий из формы, 3-х комбинированных списков (ComboBox1-3), кнопки и задайте их параметры.









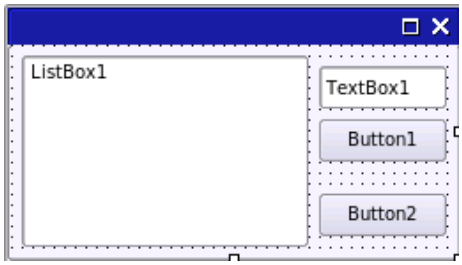
17.2. Запишите код программы:

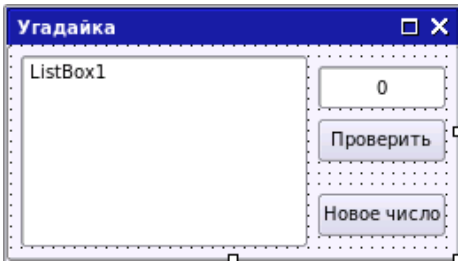
```
PUBLIC SUB Button1_Click()
    ComboBox1.Index = Int(Rnd * ComboBox1.Count)
    ComboBox2.Index = Int(Rnd * ComboBox2.Count)
    ComboBox3.Index = Int(Rnd * ComboBox3.Count)
  END
```

17.3. Сохраните проект, запустите его на исполнение и проверьте работоспособность.

18. Программа «Угадайка»

18.1. Создайте проект с именем «guess», заголовок - «Угадайка», состоящий из формы, текстового поля, списка (ListBox1), 2-х кнопок и задайте их параметры.





18.2. Объявите константу и переменные, запишите код программы:

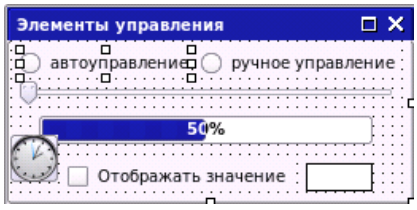
```
CONST c AS Integer = 100

STATIC n AS Integer
STATIC k AS Byte

PUBLIC SUB Form_Open()
  n = Int(Rnd * c)
  Message("Число от 0 до " & c &
    " загадано. Угадайте его.")
END

PUBLIC SUB Button2_Click()
  k = 0
  ListBox1.Clear
  Form_Open
END
```

```
PUBLIC SUB Button1_Click()
  DIM s AS String, m AS Integer
  k = k + 1
  m = Val(TextBox1.Text)
  IF m < n THEN
    s = "загаданное число больше"
  ELSE IF m > n THEN
    s = "загаданное число меньше"
  ELSE
    s = "число угадано за " & k &
      " попыток"
  ENDIF
  Message(s)
  ListBox1.Add(m & ": " & s, k)
END
```

18.3. Сохраните проект, запустите его на исполнение и проверьте работоспособность.

19. Программа «Элементы управления-1»

19.1. Создайте проект с именем «elements1», заголовок - «Элементы управления-1», состоящий из формы, 2-х переключателей (RadioButton1-2), ползунка (Slider1), индикатора выполнения (ProgressBar1), флажка (CheckBox1), таймера, текстовой метки и задайте их параметры.



(Name)	RadioButton1
Enabled	True
Text	автоуправление

(Name)	CheckBox1
Text	Отображать значение

(Name)	Timer1
Delay	100
Enabled	True

(Name)	TextLabel1
Background	TextBackground
Border	Etched
Text	

19.2. Запишите код программы:

```
STATIC t AS Byte

PUBLIC SUB Timer1_Timer()
  t = (t + 1) MOD 101
  Slider1.Value = t
END

PUBLIC SUB RadioButton1_Click()
  IF RadioButton1.Enabled = TRUE THEN
    Timer1.Enabled = TRUE
  ENDIF
END
```

```
PUBLIC SUB RadioButton2_Click()
  IF RadioButton2.Enabled = TRUE THEN
    Timer1.Enabled = FALSE
  ENDIF
END

PUBLIC SUB Slider1_Change()
  t = Slider1.Value
  ProgressBar1.Value = t / 100
  IF CheckBox1.Value = TRUE THEN
    TextLabel1.Text = t
  ENDIF
END
```

19.3. Сохраните проект, запустите его на исполнение и проверьте работоспособность.

20. Программа «Элементы управления-2»

20.1. Создайте проект с именем «elements2», заголовок - «Элементы управления-2», состоящий из формы, 3-х флажков-кнопок (ToggleButton1-3), ползунка (Slider1), индикатора выполнения (ProgressBar1), таймера, текстовой метки и задайте их параметры.

(Name)	ToggleButton1
Radio	True
Text	Автоуправление
Value	True

(Name)	ToggleButton2
Radio	True
Text	Ручное управление

(Name)	ToggleButton3
Radio	False
Text	Вкл./выкл. показ значений

(Name)	Timer1
Delay	100
Enabled	True

20.2. Запишите код программы:

```
STATIC t AS Byte

PUBLIC SUB Timer1_Timer()
  t = (t + 1) MOD 101
  Slider1.Value = t
END

PUBLIC SUB ToggleButton1_Click()
  IF ToggleButton1.Enabled = TRUE THEN
    Timer1.Enabled = TRUE
  ENDIF
END
```

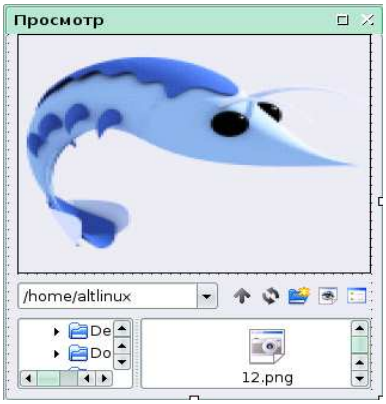
```
PUBLIC SUB ToggleButton2_Click()
  IF ToggleButton2.Enabled = TRUE THEN
    Timer1.Enabled = FALSE
  ENDIF
END

PUBLIC SUB Slider1_Change()
  t = Slider1.Value
  ProgressBar1.Value = t / 100
  IF ToggleButton3.Value = TRUE THEN
    TextLabel1.Text = t
  ENDIF
END
```

20.3. Сохраните проект, запустите его на исполнение и проверьте работоспособность.

21_1. Создание программы «Просмотр»

21_1.1. Создайте проект с именем «viewing», заголовок - «Просмотр», состоящий из формы, поле рисунка (PictureBox1), диалогового элемента ВыборФайла (FileChooser1) и задайте их параметры.

(Name)	PictureBox1
Stretch	True



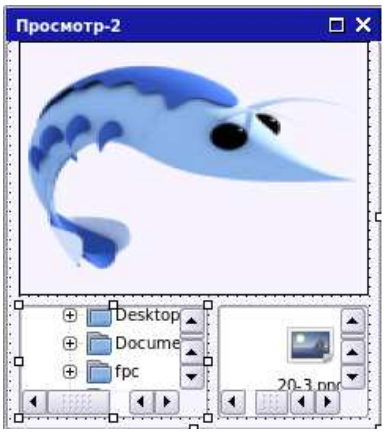
21_1.2. Запишите код программы:

```
PUBLIC SUB FileChooser1_Activate()
  PictureBox1.Picture = ""
  IF ERROR THEN
    FINALLY
  ELSE
    PictureBox1.Picture = picture.Load(FileChooser1.Value)
  ENDIF
END
```

21_1.3. Меняя каталоги и выбирая файлы для просмотра, проверьте работу проекта и сохраните его.

21_2. Создание программы «Просмотр2»

21_2.1. Создайте проект с именем «viewing2», заголовок - «Просмотр2», состоящий из формы, поле рисунка, элементов ПросмотрКаталогов (DirView1) и ПросмотрФайлов (FileView1) задайте их параметры.



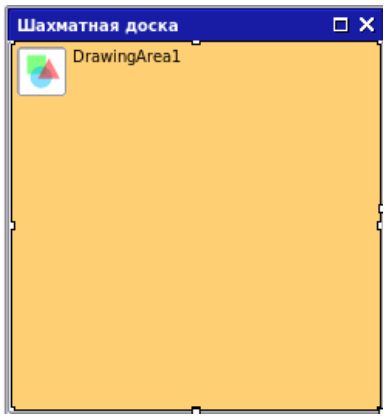
21_2.2. Запишите код программы:

```
PUBLIC SUB DirView1_Click()
  FileView1.Dir = DirView1.Current
END

PUBLIC SUB FileView1_Click()
  PictureBox1.Picture = ""
  IF ERROR THEN
    FINALLY
  ELSE
    PictureBox1.Picture = picture.Load(DirView1.Current & "/" & FileView1.Current)
  ENDIF
END
```

21_2.3. Проверьте работоспособность проекта (как в предыдущем задании) и сохраните его.

22. Создание программы «Шахматная доска»

22.1. Создайте проект с именем «chess», заголовок - «Шахматная доска», состоящий из формы, области рисования (DrawingArea1) и задайте их параметры.



(Name)	DrawingArea1
Background	&HFFCC77&
Height	240
Width	240

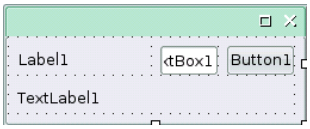
22.2. Запишите код программы:

```
PUBLIC SUB DrawingArea1_DblClick()
  DIM i, j AS Byte
  DRAW.BEGIN(DrawingArea1)
  DRAW.FillStyle = 1
  DRAW.FillColor = &773300
  FOR i = 1 TO 4
    FOR j = 1 TO 4
      DRAW.RECT(i * 60 - 30, j * 60 - 60, 30, 30)
      DRAW.RECT(i * 60 - 60, j * 60 - 30, 30, 30)
    NEXT
  NEXT
  DRAW.END
END
```

22.3. Запустив на исполнение, проверьте работу проекта (при необходимости измените параметры элементов). Сохраните проект.

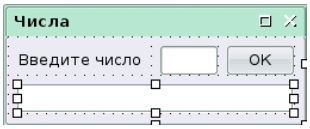
23. Начало создания программы «Числа»

23.1. В программе Gambas создайте проект с именем «numbers», заголовок - «Числа». Разместите на форме метку (надпись), текстовое поле, кнопку, текстовую метку и задайте их параметры.



23.2. Запишите процедуру для кнопки:

(Name)	TextLabel1
Background	TextBackground
Border	Etched




```
PUBLIC SUB Button1_Click()
  IF IsNumber(Val(TextBox1.Text)) THEN
    IF Val(TextBox1.Text) > 0 AND Val(TextBox1.Text) < 1000 THEN
      TextLabel1.Text = Rezult(TextBox1.Text)
    ELSE
      TextLabel1.Text = TextBox1.Text
      Message("Для преобразования введите целое число от 1 до 1000 !")
    ENDIF
  ELSE
    TextLabel1.Text = ""
    Message("Необходимо ввести целое положительное число до 1000 !")
  ENDIF
END
```

23.3. Просмотрите код программы и сохраните проект (запускать еще рано!).

24. Завершение создания программы «Числа»

24.1. Запишите пользовательские функции программы:

```
FUNCTION Rezult(n AS Integer) AS String
  DIM s1, s2, s3 AS String
  s1 = Str(n MOD 10)
  s2 = Str((n DIV 10) MOD 10)
  s3 = Str((n DIV 100) MOD 10)
  RETURN Trim(Hunds(s3)) & " " & Trim(GrTen(s2) & " " & Units(s2, s1))
END
```

```
FUNCTION Units(s2 AS String, s1 AS String) AS String
  IF s2 <> 1 THEN
    SELECT CASE s1
      CASE 1
        RETURN "один"
      CASE 2
        RETURN "два"
  …
      CASE 9
        RETURN "девять"
    END SELECT
  ELSE
    SELECT CASE s1
      CASE 0
        RETURN "десять"
      CASE 1
        RETURN "одиннадцать"
  …
      CASE 9
        RETURN "девятнадцать"
    END SELECT
  ENDIF
END
```

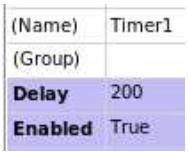
```
FUNCTION GrTen(s2 AS String) AS String
  SELECT CASE s2
    CASE 2
      RETURN "двадцать"
    CASE 3
      RETURN "тридцать"
  …
      RETURN "девяносто"
  END SELECT
END
```

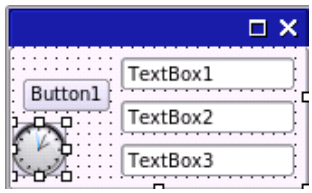
```
FUNCTION Hunds(s3 AS String) AS String
  SELECT CASE s3
    CASE 1
      RETURN "сто"
    CASE 2
      RETURN "двести"
  …
    CASE 9
      RETURN "девятьсот"
  END SELECT
END
```

24.2. Проверьте работоспособность и сохраните проект.

25. Программа «Вычислитель-2»

25.1. Создайте проект с именем «calculator2», заголовок - «Вычислитель-2», состоящий из формы, 3 текстовых полей, кнопки, таймера и задайте их параметры.





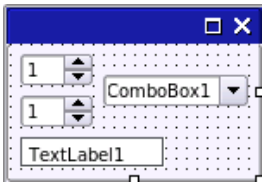
25.2. Запишите код программы:

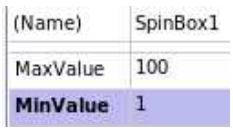
```
STATIC d AS Byte
PUBLIC SUB Button1_Click()
  d = (d + 1) MOD 4
  Timer1_Timer
END
PUBLIC SUB Timer1_Timer()
  IF TextBox1.Text <> "" AND TextBox2.Text <> "" THEN
    SELECT CASE d
      CASE 0
        Button1.Text = "+"
        TextBox3.Text = Val(TextBox1.Text) + Val(TextBox2.Text)
      CASE 1
        Button1.Text = "-"
        TextBox3.Text = Val(TextBox1.Text) - Val(TextBox2.Text)
      CASE 2
        Button1.Text = "*"
        TextBox3.Text = Val(TextBox1.Text) * Val(TextBox2.Text)
      CASE 3
        IF Val(TextBox2.Text) <> 0
          Button1.Text = "/"
          TextBox3.Text = Val(TextBox1.Text) / Val(TextBox2.Text)
        ENDIF
    END SELECT
  ELSE
    TextBox3.Text = ""
  ENDIF
END
```

25.3. Проверьте работу и сохраните проект.

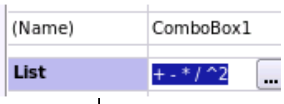
26. Программа «Вычислитель-3»

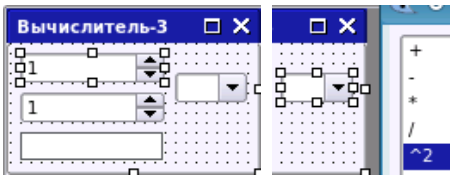
26.1. Создайте проект с именем «calculator3», заголовок - «Вычислитель-3», состоящий из формы, 2-х счетчиков (SpinBox1-2), текстовой метки, комбинированного списка (ComboBox1) и задайте их параметры.





26.2. Запишите код программы:





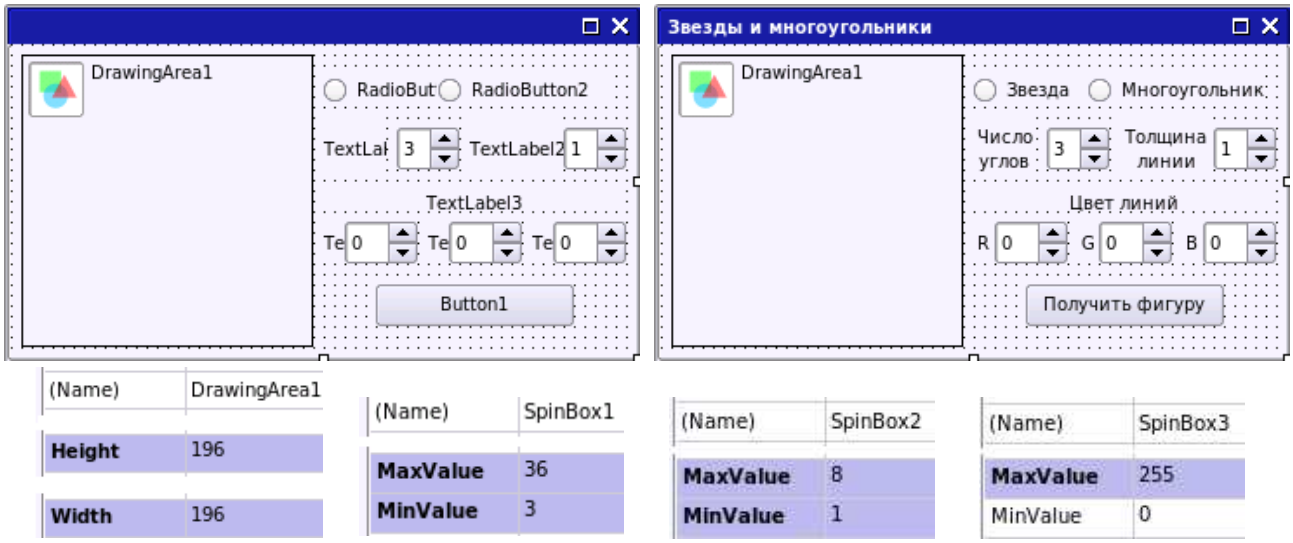

```
PUBLIC SUB ComboBox1_Change()
  DIM a, b AS Byte
  a = SpinBox1.Text
  b = SpinBox2.Text
  SELECT CASE ComboBox1.Text
    CASE "+"
      TextLabel1.Text = a + b
    CASE "-"
      TextLabel1.Text = a - b
    CASE "*"
      TextLabel1.Text = a * b
    CASE "/"
      TextLabel1.Text = a / b
    CASE ELSE
      SpinBox2.Value = a
      TextLabel1.Text = a ^ 2
  END SELECT
END
```

```
PUBLIC SUB SpinBox1_Change()
  ComboBox1_Change
  IF ComboBox1.Text = "^2" THEN
    SpinBox2.Value = SpinBox1.Text
  ENDIF
END
PUBLIC SUB SpinBox2_Change()
  ComboBox1_Change
  IF ComboBox1.Text = "^2" THEN
    SpinBox1.Value = SpinBox2.Text
  ENDIF
END
```

26.3. Проверьте работу и сохраните проект.

27. Создание интерфейса программы «Звезды и многоугольники»

27.1. Создайте проект с именем «stars_polygons», заголовок - «Звезды и многоугольники», состоящий из формы, области рисования (DrawingArea1), 2-х переключателей (RadioButton1-2), 6-и текстовых меток, 5-и счетчиков (SpinBox1-5), кнопки.



(Name)	DrawingArea1
Height	196
Width	196

(Name)	SpinBox1
MaxValue	36
MinValue	3

(Name)	SpinBox2
MaxValue	8
MinValue	1

(Name)	SpinBox3
MaxValue	255
MinValue	0

27.2. Задайте параметры компонентов. Сохраните проект.

28. Завершение создания программы «Звезды и многоугольники»

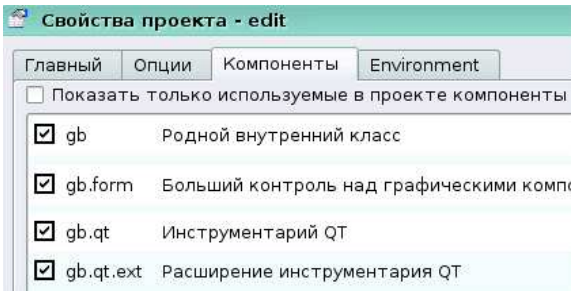
28.1. В проекте «stars_polygons» запишите код программы, используя константы, переменные и пользовательскую процедуру «star»:

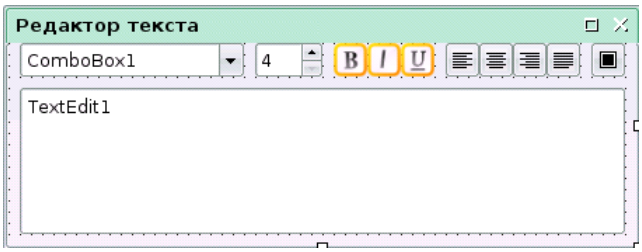
```
CONST r AS Byte = 80

CONST o AS Byte = 98

PROCEDURE star(z AS Byte, nk AS Byte, kr AS Float)
  DIM i AS Byte
  nk = nk * z
  FOR i = 0 TO nk - 1 STEP 2
    Draw.Line(o + r * Sin(2 * Pi * i / nk), o - r * Cos(2 * Pi * i / nk),
      o + r * kr * Sin(2 * Pi * (i + 1) / nk), o - r * kr * Cos(2 * Pi * (i + 1) / nk))
    Draw.Line(o + r * Sin(2 * Pi * (i + 2) / nk), o - r * Cos(2 * Pi * (i + 2) / nk),
      o + r * kr * Sin(2 * Pi * (i + 1) / nk), o - r * kr * Cos(2 * Pi * (i + 1) / nk))
  NEXT
END

PUBLIC SUB Button1_Click()
  DrawingArea1.Clear
  Draw.Begin(DrawingArea1)
  Draw.LineWidth = SpinBox2.Text
  Draw.ForeColor = SpinBox3.Text * 65536 + SpinBox4.Text * 256 + SpinBox5.Text
  IF RadioButton1.Value = TRUE THEN
    star(2, SpinBox1.Text, 0.4)
  ELSE IF RadioButton2.Value = TRUE THEN
    star(1, SpinBox1.Text, 1)
  ENDIF
  Draw.End()
END
```

28.2. Запустив на исполнение, проверьте работу проекта (при необходимости измените параметры элементов, константы). Сохраните проект.

29. Создание интерфейса программы «Редактор текста»

29.1. Создайте проект с именем «edit», заголовок - «Редактор текста». Добавьте компоненты: «gb.qt - Инструментарий QT», «gb.qt.ext - Расширение инструментария QT» (Проект-Свойства-Компоненты). Разместите на форме комбинированный список, счетчик, 7 инструментальных кнопок (ToolButton1-3, ToolButton4-7), кнопку (выбора) цвета (ColorButton1), поле редактора текста (TextEdit1).





(Name)	SpinBox1
MaxValue	72
MinValue	4

(Name)	ToolButton1
Picture	icon:/24/text-bold
Radio	False
Toggle	True

(Name)	ToolButton4
Picture	icon:/16/text-left
Radio	True
Toggle	True

29.2. Задайте параметры компонентов. Сохраните проект.

30. Завершение создания программы «Редактор текста»

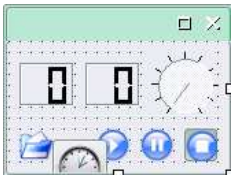
30.1. Запишите код программы:

```
PUBLIC SUB Form_Open()
  DIM s AS String
  FOR EACH s IN Fonts
    ComboBox1.add(s)
  NEXT
END

PUBLIC SUB ComboBox1_Click()
  TextEdit1.Format.Font.Name = LAST.Text
END

PUBLIC SUB SpinBox1_Change()
  TextEdit1.Format.Font.Size = LAST.Value
END

PUBLIC SUB ToolButton1_Click()
  TextEdit1.Format.Font.Bold = LAST.Value
END

PUBLIC SUB ToolButton2_Click()
  TextEdit1.Format.Font.Italic = LAST.Value
END

PUBLIC SUB ToolButton3_Click()
  TextEdit1.Format.Font.Underline = LAST.Value
END

PUBLIC SUB ToolButton4_Click()
  TextEdit1.Format.Alignment = Align.Left
END

PUBLIC SUB ToolButton5_Click()
  TextEdit1.Format.Alignment = Align.Center
END

PUBLIC SUB ToolButton6_Click()
  TextEdit1.Format.Alignment = Align.Right
END

PUBLIC SUB ToolButton7_Click()
  TextEdit1.Format.Alignment = Align.Justify
END

PUBLIC SUB ColorButton1_Change()
  TextEdit1.Format.Color = LAST.Value
END

PUBLIC SUB TextEdit1_Cursor()
  ToolButton1.Value = TextEdit1.Format.Font.Bold
  ToolButton2.Value = TextEdit1.Format.Font.Italic
  ToolButton3.Value =
TextEdit1.Format.Font.Underline
  SELECT CASE TextEdit1.Format.Alignment
    CASE Align.Left
      ToolButton4.Value = TRUE
    CASE Align.Right
      ToolButton5.Value = TRUE
    CASE Align.Center
      ToolButton6.Value = TRUE
    CASE Align.Justify
      ToolButton7.Value = TRUE
  END SELECT
  ColorButton1.Color = TextEdit1.Format.Color
  ComboBox1.Text = TextEdit1.Format.Font.Name
  SpinBox1.Value = TextEdit1.Format.Font.Size
END
```

30.2. Запустите проект и наберите текст в поле редактора. Проверьте работу проекта (меняйте форматы выделенных фрагментов текста с помощью элементов управления). Сохраните проект.

31. Программа «Музыкальный проигрыватель»

31.1. Создайте проект с именем «music», заголовок - «Музыкальный проигрыватель» с компонентами: «gb», «gb.form», «gb.qt», «gb.qt.ext», «gb.stl.sound». Разместите на форме 2 «жидкокристаллических числа» (LCDNumber1-2), круговой бегунок (Dial1), таймер, 4 инструментальных кнопки (ToolButton1-4) и задайте их параметры.



(Name)	LCDNumber1
Border	Etched
Digits	2
Style	Flat

(Name)	Dial1
Step	10

(Name)	Timer1
Delay	1000
Enabled	True

(Name)	ToolButton1
Picture	icon:/24/open
Radio	False

(Name)	ToolButton2
Picture	icon:/24/play
Radio	True
Toggle	True

(Name)	ToolButton3
Picture	icon:/24/pause
Radio	True
Toggle	True

(Name)	ToolButton4
Picture	icon:/24/stop
Radio	True
Toggle	True

31.2. Запишите код программы:

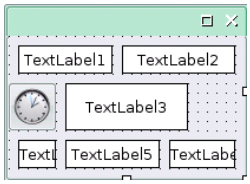
```
STATIC t AS Integer

PUBLIC SUB Timer1_Timer()
  t = Music.Pos
  LCDNumber1.Value = t \ 60
  LCDNumber2.Value = t MOD 60
END

PUBLIC SUB ToolButton1_Click()
  Dialog.OpenFile
  IF NOT ERROR THEN
    Music.Load(Dialog.Path)
    ToolButton2.Value = TRUE
    ToolButton2_Click
  ELSE
    FINALLY
  ENDIF
END
```

```
PUBLIC SUB
ToolButton2_Click()
  Music.Play
  Dial1_Change
END

PUBLIC SUB ToolButton3_Click()
  Music.Pause
END

PUBLIC SUB ToolButton4_Click()
  Music.Stop
END

PUBLIC SUB Dial1_Change()
  Music.Volume = Dial1.Value / 100
END

PUBLIC SUB Form_Open()
  Dial1.Value = 25
END
```

31.3. Поместите звуковой файл (в формате mp3 или wav) в каталог проекта («Projects/music»). Запустите и проверьте проект проигрывая звуковой файл. Сохраните проект.

32. Программа «Дата-Время»

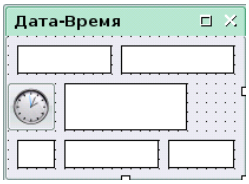
32.1. Создайте проект с именем «date_time», заголовок - «Дата-Время», состоящий из формы, таймера, 6 текстовых меток и задайте их параметры.



(Name)	TextLabel1
Alignment	Center
Background	TextBackground
Border	Plain

(Name)	TextLabel3
Alignment	Center
Background	TextBackground
Border	Plain
Font	+3

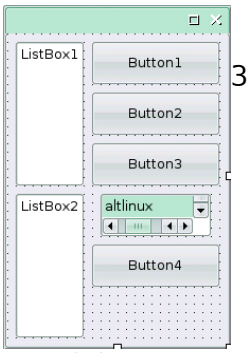
(Name)	Timer1
Delay	1000
Enabled	True



32.2. Запишите код программы:

```
PUBLIC SUB Timer1_Timer()
  TextLabel1.Text = Date
  SELECT CASE WeekDay(Date)
    CASE 0
      TextLabel2.Text = "воскресенье"
    CASE 1
      TextLabel2.Text = "понедельник"
    ...
    CASE 6
      TextLabel2.Text = "суббота"
  END SELECT
  TextLabel3.Text = Format(Time)
  TextLabel4.Text = Day(Date)
  SELECT CASE Month(Date)
    CASE 1
      TextLabel5.Text = "января"
    CASE 2
      TextLabel5.Text = "февраля"
    ...
    CASE 12
      TextLabel5.Text = "декабря"
  END SELECT
  TextLabel6.Text = Year(Date) & "г."
END
```

32.3. Проверьте и сохраните проект.



3

33. Создание программы «Сохранить-загрузить»

Создайте проект с именем «save_load», заголовок - «Сохранить-загрузить». Разместите на форме 2 поля со списками, 4 кнопки, элемент - просмотр каталога и задайте их параметры.

33.2. Запишите код программы:

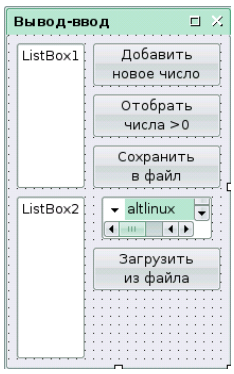
```
PUBLIC SUB
Button1_Click()
  DIM n AS Byte
  n = Rnd(100) + 1
  ListBox1.Add(n)
END
PUBLIC SUB Button2_Click()
  DIM i, j, n AS Byte
  FOR i = 0 TO ListBox1.List.Count - 2
    FOR j = i + 1 TO ListBox1.List.Count - 1
      IF Val(ListBox1.List[i]) > Val(ListBox1.List[j]) THEN
         n = Val(ListBox1.List[i])
         ListBox1.Remove(i)
         ListBox1.Add(ListBox1.List[j - 1], i)
         ListBox1.Remove(j)
         ListBox1.Add(n, j)
      ENDIF
    NEXT
  NEXT
END
```

```
PUBLIC SUB Button3_Click()
  DIM s AS String
  DIM i AS Byte
  FOR i = 0 TO ListBox1.List.Count - 1
    s = s & ListBox1.List[i] & Chr(13)
  NEXT
  File.Save(DirView1.Current & "/text.txt", s)
END
PUBLIC SUB Button4_Click()
  DIM s, st AS String
  DIM i AS Byte
  ListBox2.Clear
  st = File.Load(DirView1.Current & "/text.txt")
  FOR i = 1 TO Len(st)
    IF Mid(st, i, 1) <> Chr(13) THEN
      s = s & Mid(st, i, 1)
    ELSE
      ListBox2.Add(s)
      s = ""
    ENDIF
  NEXT
END
```

33.3. Проверьте и сохраните проект.

34. Создание программы «Вывод-ввод»

34.1. Создайте проект с именем «out_input», заголовок - «Вывод-ввод». Разместите на форме 2 поля со списками, 4 кнопки, элемент - просмотр каталога и задайте их параметры.



34.2. Запишите код программы:

```
PUBLIC SUB Button1_Click()
  DIM n AS Integer
  n = Rnd(80) - 30
  ListBox1.Add(n)
END
PUBLIC SUB Button2_Click()
  DIM i AS Byte
  FOR i = ListBox1.List.Count - 1 TO
    0 STEP -1
      IF Val(ListBox1.List[i]) <= 0 THEN
        ListBox1.Remove(i)
      ENDIF
  NEXT
END
PUBLIC SUB Button3_Click()
  DIM i AS Byte
  DIM fOut AS File
  DIM s AS String
  s = DirView1.Current & "/text2.txt"
  File.Save(s, "")
  fOut = OPEN s FOR OUTPUT
  FOR i = 0 TO ListBox1.List.Count - 1
    PRINT #fOut, ListBox1.List[i]
  NEXT
END
PUBLIC SUB Button4_Click()
  DIM i AS Byte
  DIM fIn AS File
  DIM s AS String
  ListBox2.Clear
  s = DirView1.Current & "/text2.txt"
  fIn = OPEN s FOR INPUT
  DO UNTIL Eof(fIn)
    LINE INPUT #fIn, s
    ListBox2.Add(s)
  LOOP
END
```

34.3. Проверьте и сохраните проект.

35. Создание программы «Часы»

35.1. В программе Gambas создайте проект с именем «clock», заголовок - «Часы». Разместите на форме область рисования, таймер и задайте их параметры.

(Name)	DrawingArea1
Height	240
Width	240

(Name)	Timer1
Delay	1000
Enabled	True

35.2. Запишите процедуру для таймера:

```
PUBLIC SUB Timer1_Timer()
  DIM i, s, m, h AS Byte
  s = Second(Time)
  m = Minute(Time)
  h = Hour(Time)
  IF s MOD 10 = 0 THEN DrawingArea1.Refresh
'или IF s MOD 10 = 0 THEN DrawingArea1.Clear
  DRAW.BEGIN(DrawingArea1)
    DRAW.ForeColor = &H000077&
    DRAW.LineWidth = 4
    FOR i = 0 TO 11
      DRAW.Line(120 + 105 * Cos(Pi(i / 6)),
        120 + 105 * Sin(Pi(i / 6)), 120 + 115 *
        Cos(Pi(i / 6)), 120 + 115 * Sin(Pi(i / 6)))
    NEXT
    DRAW.ForeColor = &HFF7700&
    DRAW.LineWidth = 2
    DRAW.Line(120, 120, 120 + 100 * Sin(
      Pi(s / 30)), 120 - 100 * Cos(Pi(s / 30)))
    DRAW.ForeColor = &H0000FF&
    DRAW.LineWidth = 4
    DRAW.Line(120, 120, 120 + 80 * Sin(Pi(
      m / 30)), 120 - 80 * Cos(Pi(m / 30)))
    DRAW.ForeColor = &H770077&
    DRAW.LineWidth = 6
    DRAW.Line(120, 120, 120 + 65 * Sin(Pi(
      h / 6 + m / 360)), 120 - 65 * Cos(Pi(
      h / 6 + m / 360)))
  DRAW.END
END
```

35.3. Проверьте работу, при необходимости измените параметры элементов и сохраните проект.

36. Создание программы «Часы-2»

36.1. В программе Gambas создайте проект с именем «clock2», заголовок - «Часы-2». Разместите на форме область рисования, таймер и задайте их параметры.

(Name)	DrawingArea1
Height	240
Width	240

(Name)	Timer1
Delay	1000
Enabled	True

36.2. Запишите процедуру для таймера:

```
PUBLIC SUB Timer1_Timer()
  DIM i, s, m, h AS Byte
  s = Second(Time)
  m = Minute(Time)
  h = Hour(Time)
  DRAW.BEGIN(DrawingArea1)
    DRAW.Picture(Picture.Load("clock.png"), 0, 0)
    DRAW.ForeColor = &HFF7700&
    DRAW.LineWidth = 2
    DRAW.Line(120, 120, 120 + 100 * Sin(
      Pi(s / 30)), 120 - 100 * Cos(Pi(s / 30)))
    DRAW.ForeColor = &H0000FF&
    DRAW.LineWidth = 4
    DRAW.Line(120, 120, 120 + 80 * Sin(Pi(
      m / 30)), 120 - 80 * Cos(Pi(m / 30)))
    DRAW.ForeColor = &H770077&
    DRAW.LineWidth = 6
    DRAW.Line(120, 120, 120 + 65 * Sin(Pi(
      h / 6 + m / 360)), 120 - 65 * Cos(Pi(
      h / 6 + m / 360)))
  DRAW.END
END
```

36.3. Создайте рисунок - циферблат часов и сохраните его в каталоге проекта («Projects/clock2») под именем «clock.png». Проверьте работу, при необходимости измените параметры элементов и сохраните проект.