Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Основы алгоритмизации и программирования (ОАиП)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе

на тему

ПРОГРАММНОЕ СРЕДСТВО ПОСТРОЕНИЯ И ОТОБРАЖЕНИЯ СЕТЕВЫХ И ДРЕВОВИДНЫХ СТРУКТУР

БГУИР КР 1-40 01 01 19 ПЗ

Студент: гр. 951001 Северин К.М.

Руководитель:

асс. Фадеева Е.Е.

Минск 2020

ЗАДАНИЕ  
Программное средство построения и отображения сетевых и древовидных структур

**СОДЕРЖАНИЕ**

[**ВВЕДЕНИЕ** 4](#_Toc41986520)

[**1.** **АНАЛИЗ ПРОТОТИПОВ, ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПРОГРАММНОМУ СРЕДСТВУ** 5](#_Toc41986521)

[**1.1** **Анализ программных средств построения и отображения сетей и древовидных структур** 5](#_Toc41986522)

[**1.1.1** **Microsoft Visio** 5](#_Toc41986523)

[**1.1.2** **EdrawMax** 6](#_Toc41986524)

[**1.1.3** **GoVisual Diagram Editor** 7](#_Toc41986525)

[**1.2** **Формирование требований к проектируемому программному средству** 8](#_Toc41986526)

[**2.** **АНАЛИЗ ТРЕБОВАНИЙ К ПРОГРАММНОМУ СРЕДСТВУ И РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ** 8](#_Toc41986527)

[**2.1** **Описание функциональности программного средства** 8](#_Toc41986528)

[**3.** **ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА** 10](#_Toc41986529)

[**3.1** **Обобщенный алгоритм работы программного средства** 10](#_Toc41986530)

[**3.2** **Разработка класса для хранения данных** 10](#_Toc41986531)

[**3.3** **Разработка классов для хранения данных о созданных пользователем блоков** 12](#_Toc41986532)

[**3.3.1** **Описание класса TControlObject** 12](#_Toc41986533)

[**3.3.2** **Описание класса TText** 14](#_Toc41986534)

[**3.3.3** **Описание класса TElement** 15](#_Toc41986535)

[**3.3.4** **Описание класса TLine** 17](#_Toc41986536)

[**3.3.5** **Описание структуры TConnecor** 18](#_Toc41986537)

[**4.** **СОЗДАНИЕ (КОНСТРУИРОВАНИЕ) ПРОГРАММНОГО СРЕДСТВА** 18](#_Toc41986538)

[**4.1** **Разработка главной формы программного средства** 18](#_Toc41986539)

[**4.2** **Разработка модального окна изменения параметров документа** 24](#_Toc41986540)

[**5.** **ТЕСТИРОВАНИЕ ПРОВЕРКА РАБОТОСПОСОБНОСТИ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ** 25](#_Toc41986541)

[**6.** **РУКОВОДСТВО ПО УСТАНОВКЕ И ИСПОЛЬЗОВАНИЮ** 28](#_Toc41986542)

[**ЗАКЛЮЧЕНИЕ** 32](#_Toc41986543)

[**СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ** 33](#_Toc41986544)

[**ВЕДОМОСТЬ** 36](#_Toc41986545)

[**Формат А1** 36](#_Toc41986546)

[**Формат А1** 36](#_Toc41986547)

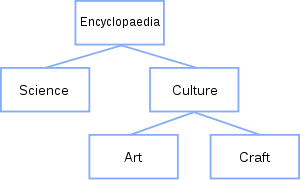
[**Формат А1** 36](#_Toc41986548)

# **ВВЕДЕНИЕ**

В наше время существует большое количество различных программных средств, в алгоритмах которых используется теория графов. Например, Google maps. Существует так же много известных алгоритмов использующие деревья, как структуру данных

**Граф** — это абстрактное представление множества объектов и связей между ними.

Деревья и сети являются частным случаем графа.

**Древовидная структура** является одним из способов представления иерархической структуры в графическом виде.

Простейшим примером графа может стать карта метро, где станции будут составлять множество вершин, а дороги между станциями будут составлять множество ребер.

Целью курсовой работу является разработка программного средства, в котором можно строить графы и его подвиды в графическом отображении.

Данная пояснительная записка содержит следующие основные разделы. В первом разделе выполнен анализ прототипов, литературных источников. Во втором разделе сформированы функциональные требования к проектируемому программному средству. Третий раздел посвящен проектированию программного средства. Четвертый раздел описывает этапы создания программного средства. Пятый радел содержит набор тестов и сценарии тестирования. В шестом разделе описано руководство по установке и использованию. Завершающий раздел содержит выводы.

# **АНАЛИЗ ПРОТОТИПОВ, ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОМУ ПРОГРАММНОМУ СРЕДСТВУ**

## **Анализ программных средств построения и отображения сетей и древовидных структур**

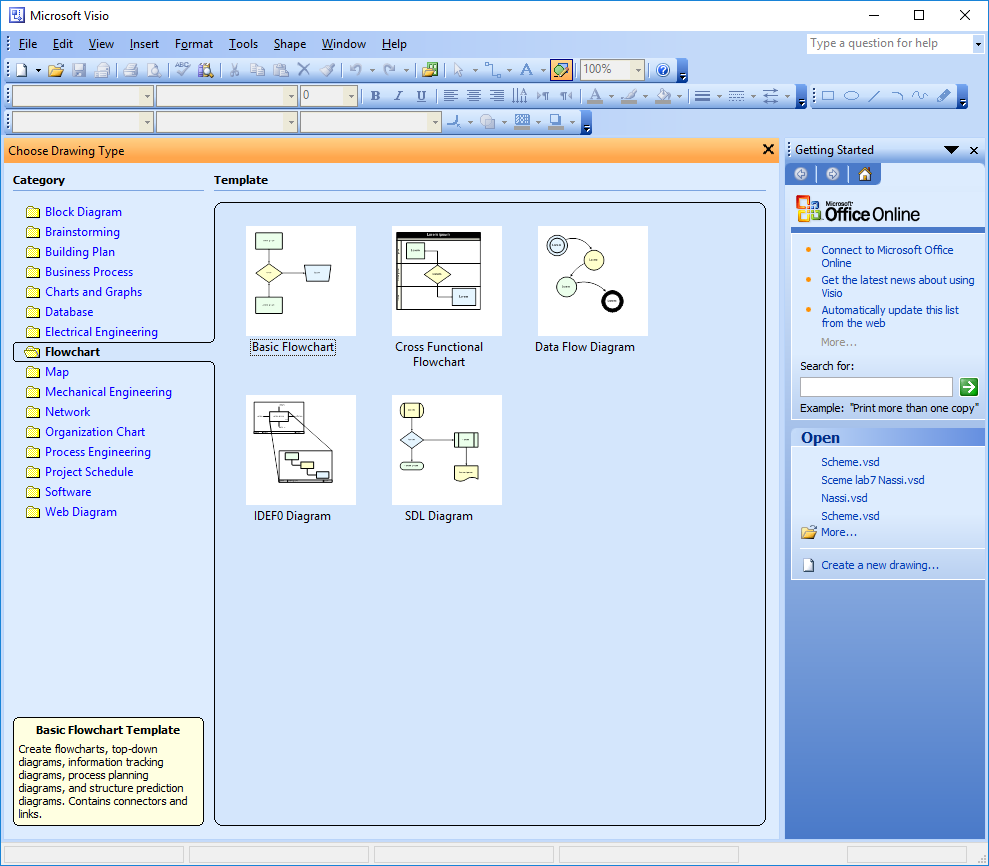
В настоящее время большинство программ, в которых можно построить некоторую схему или граф, позиционируют себя как средство для создания различного уровня диаграмм и имеют большое количество шаблонов.

Общепринятого формата файла для таких программ нет, т.е. каждая компания создает свой тип файла.

## **Microsoft Visio**

Microsoft Visio предоставляет возможности для быстрого создания деловой графики различной степени сложности: схем бизнес процесса, технических, инженерных рисунков, презентаций, разнообразных вариантов организационных, маркетинговых и технических диаграмм, электронных схем, систем транспортных коммуникаций и т. д.

Программа имеет традиционный интерфейс для приложений Windows.



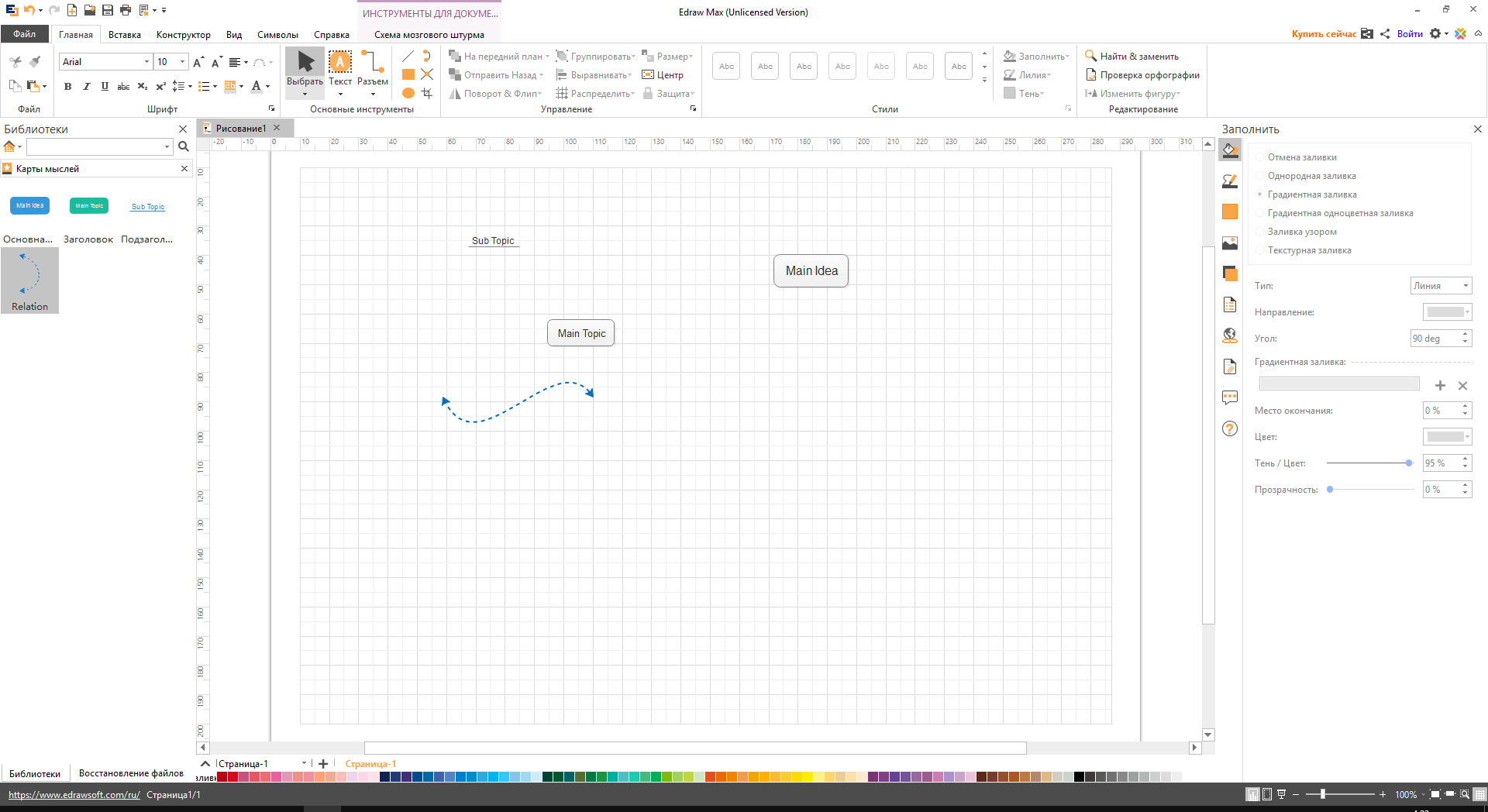
Достоинства:

* Векторный графический редактор
* Имеет достаточно большую библиотеку различных шаблонов
* Шаблоны собраны по категориям
* Традиционный для приложений Windows интерфейс
* Поддерживает большое количество форматов. (В основном это форматы поддерживаемые исключительно программой Visio. VSD, VSS, VST, VDX, VSX, VTX, VSL, VSDX, VSDM)
* Возможность сохранение рабочей области в большинство форматов графических изображений (JPEG, PNG, GIF, BMP и др.)
* Возможность изменять вид блоков

Недостатки:

* Не самый удобный способ выбора графического элемента. (Каждый раз приходиться искать нужный блок).
* Жёсткая политика лицензирования. Продукт стоит достаточно дорого, (около $345)
* Доступна только на операционной системе Windows

## **EdrawMax**

Edraw Max - приложение для деловой графики, в котором удобно создавать схемы, диаграммы, инфографику, иллюстрации для презентационных нужд и деловой литературы. 

Достоинства:

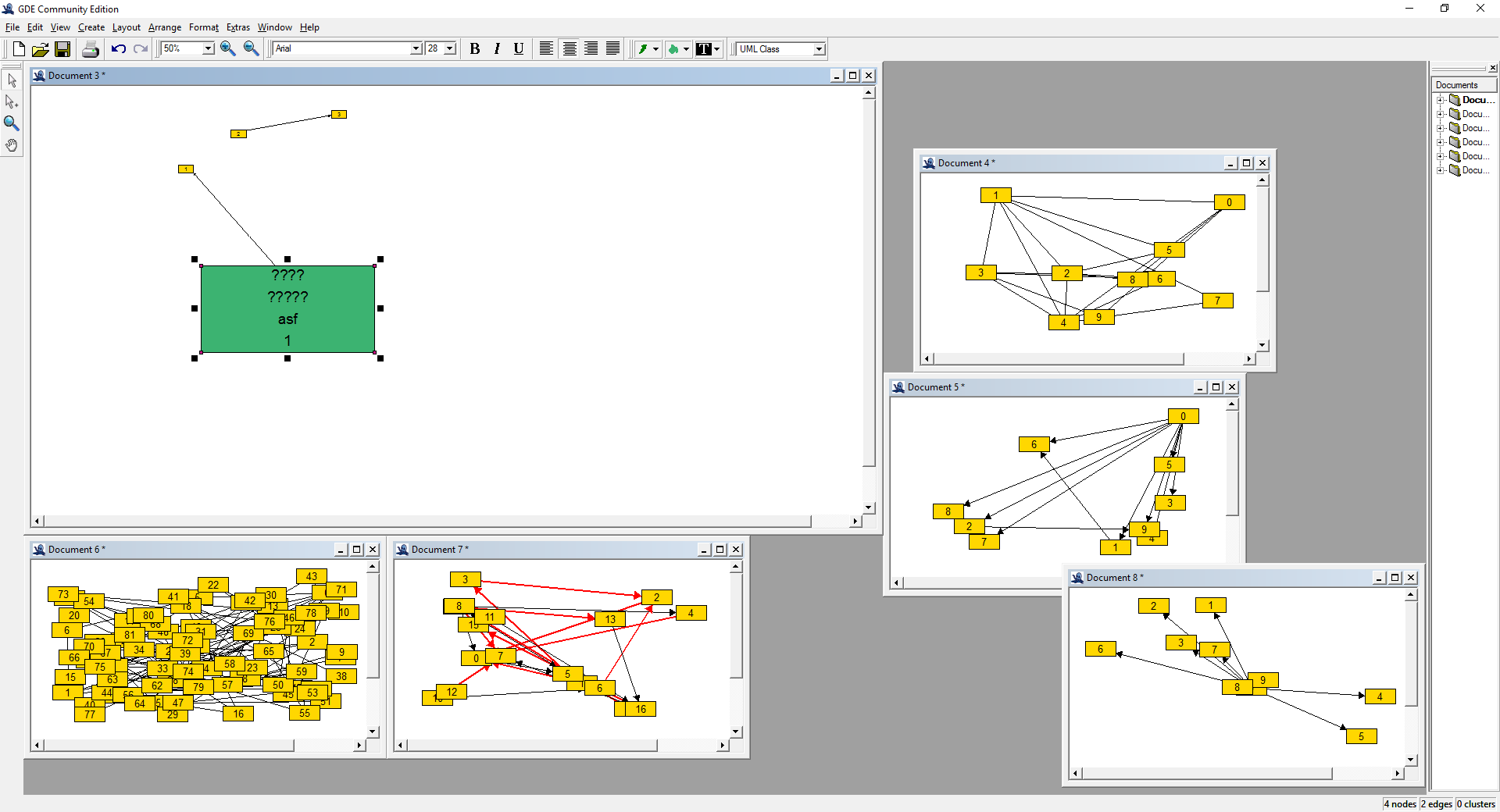
* Доступна на операционных системах Windows, Linux, MacOS
* Векторный графический редактор
* Включает в себя 280 типов диаграмм
* Поддерживает такие форматы Edraw XML файлов .eddx, .edx
* Есть пробная версия
* Возможность изменять вид блоков
* Cтандартный интерфейс приложений из популярного пакета MS Office.
* Возможность сохранение рабочей области в большинство форматов графических изображений (JPEG, PNG, GIF, BMP и др.)

Недостатки:

* Пробная версия доступна в течении 1 месяца. Лицензия же стоит $170
* Из-за большой библиотеки типов диаграмм, много времени уходит на поиск нужного шаблона
* Поиск и выбор графического элемента происходит по большому количеству категорий, что занимает много времени.

## **GoVisual Diagram Editor**

Редактор диаграмм GoVisual (GDE) предоставляет мощные функции для редактирования и автоматического размещения диаграмм. Диаграммы представлены в виде графиков и кластерных графов.

Достоинства:

* Является полностью бесплатной программой
* Доступна на таких платформах как Windows, Linux
* Удобный интерфейс
* Быстрое создание диаграмм, графов, деревьев
* Есть функции генерирования случайных графов, деревьев по определенным параметрам
* Возможность работы сразу с несколькими проектам в одном приложении
* Поддерживает единственный формат файла .gml
* Возможность экспорта в JPEG, PNG, BMP SCG, CSV

Недостатки:

* Поддержка программного средства закончилась в 2004 году
* Нет поддержки Unicode. (Отсутствует кириллица)
* Есть лишь один тип графических элементов: прямоугольник
* Ограниченные возможности изменения графических элементов (Цвет блока, текст, размер)

## **Формирование требований к проектируемому программному средству**

Согласно заданию на курсовой проект необходимо разработать программное средство для построения и отображения древовидных структур и сетей. Основные функциональные возможности:

* Выполнять экспорт рабочей области в популярные форматы изображений BMP, JPEG и др.
* Поддерживать специфический формат файла, конкретно данного программного средства
* Иметь удобный графический пользовательский интерфейс
* Поддерживать несколько видов инструментов создания схем
* Изменение параметров рабочей области
* Поддерживать несколько форм графических блоков

В качестве языка программирования было предложено использовать язык Delphi. Платформой разработки является операционная система Windows, так как большая часть аудитории пользуется именно ей.

# **АНАЛИЗ ТРЕБОВАНИЙ К ПРОГРАММНОМУ СРЕДСТВУ И РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ**

## **Описание функциональности программного средства**

Выполнение экспорта рабочей области в форматы изображений я решил объединить с сохранением в специфический формат файла.

Для сохранения в какой-либо поддерживаемый формат файла пользователю необходимо пройти по меню «File –Save as» или воспользоваться горячей клавишей Ctrl+S, в результате чего откроется диалоговое окно c выбором места сохранения, имени и расширения файла.

Для открытия специфичного файла, пользователь должен нажать в главном меню программы «File», а затем «Open», или воспользоваться горячей клавишей Ctrl+O. Далее пользователь увидит диалоговое окно с выбором открываемого файла.

Выбор инструмента создания схемы или вида графического элемента производиться через кнопки на боковой панели.

При выборе некоторого инструмента и клику по рабочей области происходит взаимодействие с блоками.

Для изменения параметров рабочей области можно пройти по следующим пунктам главного меню программы «Image – Resolution»

* 1. Спецификация функциональных требований

Среди функциональных требований есть поддержка нескольких видов инструментов. В программном средстве реализованы следующие инструменты:

* «Мышь»
  + Пользователь должен иметь возможность выделять созданные объекты для дальнейшего взаимодействия
  + Пользователь должен иметь возможность перемещать созданные объекты
* «Прямоугольник»
  + Пользователь должен иметь возможность создать элемент с формой прямоугольника
* «Эллипс»
  + Пользователь должен иметь возможность создать элемент с формой эллипса
* «Круг»
  + Пользователь должен иметь возможность создать элемент с формой круга
* «Линия»
  + Пользователь должен иметь возможность создавать линии на рабочем пространстве
* «Текст»
  + Пользователь должен иметь возможность напечатать текст на рабочем пространстве

В программном средстве должны быть реализованы следующие формы графических блоков:

* Прямоугольник
* Эллипс
* Круг
* Линия
* Текст

Для графических элементов «Прямоугольник», «Эллипс», «Круг»:

* Пользователь должен иметь возможность изменять внешний вид блоков:
  + Изменять размер блока
  + Изменять точно положение блока
  + Менять цвет фона блока
  + Писать текст внутри блока
  + Менять шрифт текста
  + Менять цвет текста
  + Менять кегль текста
  + Менять цвет линии обводки
  + Менять толщину линии обводки

Для графического элемента «Линия»:

* Пользователь должен иметь возможность менять внешний вид линии
  + Менять цвет линии
  + Менять толщину линии
* Должна быть реализована привязки линии к блокам «Прямоугольник», «Эллипс», «Круг»

Для графического элемента «Текст»

* Пользователь должен иметь возможность менять внешний вид текста
  + Изменять размер поля с текста
  + Изменять точно положение текста
  + Менять шрифт текста
  + Менять цвет текста
  + Менять кегль текста

Спецификация функции экспорта рабочей области в форматы изображений:

* Поддерживаемые форматы:
  + JPEG
  + PNG
  + BMP
* Пользователь должен видеть привычное диалоговое окно сохранения файла в операционной системе Windows

Сохранение и открытие специфичного формата файла:

* Расширение файла .tmf (TreeMakerFile)
* Должна производиться запись/чтение из файла:
  + Размера рабочей области
  + Информации о количестве блоков, о параметрах каждого блока (к параметрам относятся: размер; положение; цвет фона; цвет и толщина линии обводки; шрифт, цвет, кегль и размер текста)
  + Информации о положении линии и о возможной привязке линии к какому-нибудь блоку.

Изменение параметров рабочей области производиться через модальное окно. В модальном окне можно изменить два параметра: ширину и высоту рабочей области. Открытие модального окна происходит через пункт меню «Image - Resolution»

# **ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО СРЕДСТВА**

## **Обобщенный алгоритм работы программного средства**

Схема алгоритма программы представлена в приложении 1

## **Разработка класса для хранения данных**

Для хранения созданных пользователем блоков мною было решено использование динамических массивов в качестве структуры данных.

Класс TVector<T> отвечает за реализацию этой структуры данных.

Таблица 3.1 - Поля класса TVector<T>

|  |  |  |
| --- | --- | --- |
| **Идентификатор поля** | **Назначение** | **Тип поля** |
| FData | Динамический массив для хранения данных | Array of T; |
| FDataSize | Размер(количество) используемых ячеек массива | Integer |
| FReservedSize | Фактический размер динамического массива | Integer |

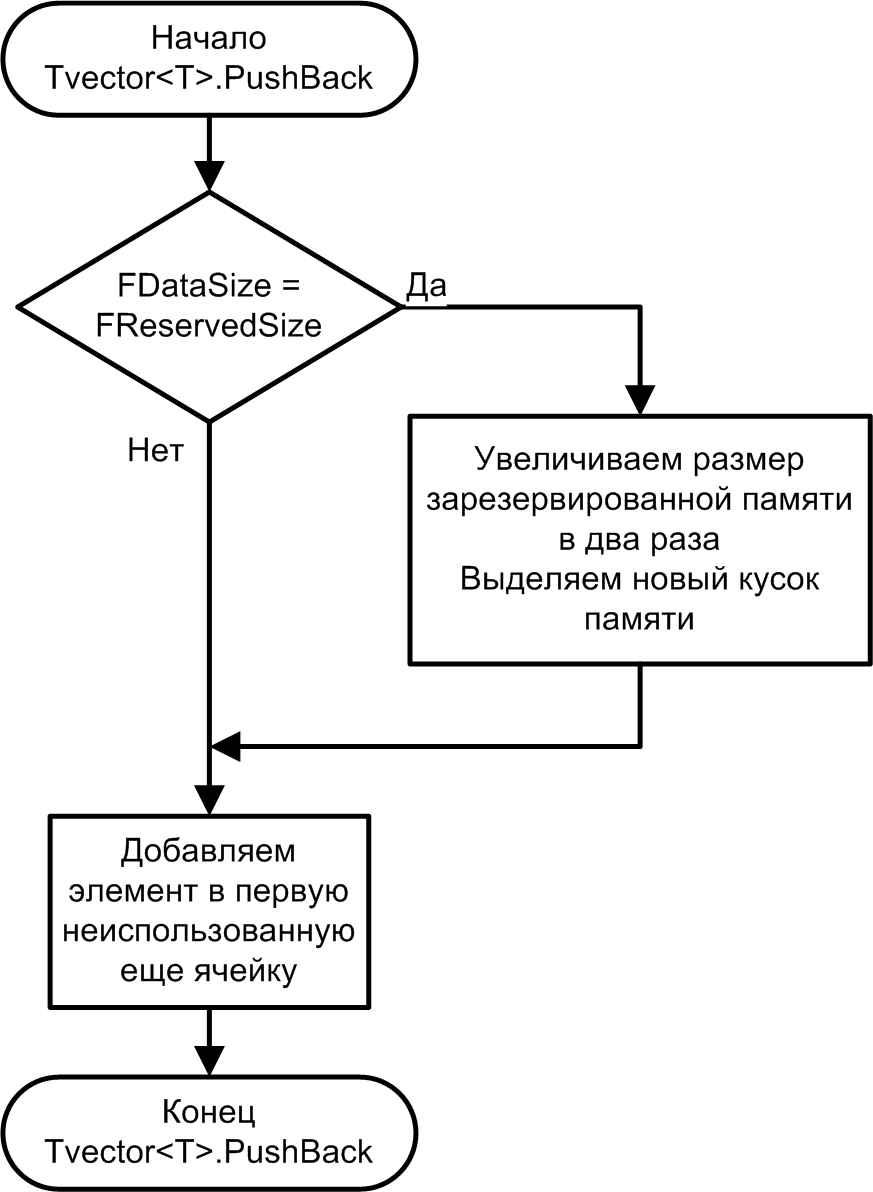


Рисунок 3.1 – Блок-схема добавления элемента в конец динамического массива

Для возможности увеличения в два зарезервированной памяти мною было решено хранить два значения, объем зарезервированной памяти и заполненной мною памяти. Это нужно для того чтобы улучшить производительность программного средства. Если использовать обычный алгоритм добавления элемента в конец массива, то есть увеличивать размер динамического массива на единицу, то мы получим на N запросов добавления элемента в конец асимптотике O(N^2), так как каждый раз будет производиться поиск нового куска памяти нужного нам размера, а затем поэлементно копироваться в новую память. В свою очередь, если мы будем увеличивать размер динамического массива вдвое, то это позволит нам достичь асимптотики в O(N logN), что значительно ускоряет работу с программой при больших объемах данных. Схема алгоритма добавления элемента в массив представлена на рисунке 3.1.

## **Разработка классов для хранения данных о созданных пользователем блоков**

Для хранения информации о блоках, расположенных на рабочем пространстве было решено создать следующие классы:

1. TControlObject
2. TText наследующий TControlObject
3. TElement наследущий TControlObject
4. TLine

Также была создана структура TConnector для описания позиции конца линии

### **Описание класса TControlObject**

В таблице 3.2 приведены поля класса TControlObject

В таблице 3.3 приведено описание основных методов класса TControlObject

Таблица 3.2 - Поля класса TControlObject

|  |  |  |
| --- | --- | --- |
| **Идентификатор поля** | **Назначение** | **Тип поля** |
| Fid | Уникальный номер блока | Integer |
| FIsVisible | Флаг, означающий является ли элемент видимым | Boolean |
| FIsSelected | Флаг, означающий выделен ли сейчас элемент | Boolean |
| FCanvas | Объект класса TCanvas. Нужен для отрисовки блока | TCanvas |
| FLeft | Положение левой границы блока относительно левой границы документа | Integer |
| FTop | Положение верхней границы блока относительно верхней границы документа | Integer |
| FWidth | Ширина блока | Integer |
| FHeigth | Высота блока | Integer |

Таблица 3.3 – Основные методы класса TControlObject

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Идентификатор метода** | **Назначение** | **Заголовок метода** | **Идентификатор параметра** | **Назначение параметра** |
| SetPosition | Устанавливает блок в позицию X, Y | procedure SetPosition(const X, Y: Integer); | X | Новое значение положения левой границы блока |
| Y | Новое значение положения верхней границы блока |
| Move | Перемещает блок на DeltaX единиц по горизонтали и на DeltaY единиц по вертикали | procedure Move(const DeltaX, DeltaY: Integer); | DeltaX | Перемещение по горизонтали |
| DeltaY | Перемещение по вертикали |
| SetSize | Устанавливает ширину блока в AWidth единиц и высоту блока в AHeigth единиц | procedure SetSize(const AWidth, AHeigth: Integer); | AWidth | Новое значение ширины блока |
| AHeigth | Новое значение высоты блока |

### **Описание класса TText**

В таблице 3.4 приведены поля класса TText

В таблице 3.5 приведены основные методы класса TText

Таблица 3.4 - Поля класса TText

|  |  |  |
| --- | --- | --- |
| **Идентификатор поля** | **Назначение** | **Тип поля** |
| Унаследованные поля от класса TControlObject (смотри таблицу 3.2) | | |
| FCaption | Текст, отображаемый на блоке | String |
| FBrush | Кисть, которой отрисовывается фон блока | TBrush |
| FFont | Шрифт текста | TFont |
| FTextFormat | Положение текста | TTextFormat |

Таблица 3.5 – Основные методы класса TText

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Идентификатор метода** | **Назначение** | **Заголовок метода** | **Идентификатор параметра** | **Назначение параметра** |
| Draw | Отрисовка текста | procedure Draw; | - | - |
| IsInside | Выдает истину, если точка c координатами X, Y находиться внутри блока, иначе ложь | function IsInside(const X, Y: Integer): Boolean; | X | Положение точки по горизонтали |
| Y | Положение точки по вертикали |

### **Описание класса TElement**

В таблице 3.6 приведены поля класса TElement

В таблице 3.7 приведены основные методы класса TElement

На рисунке 3.2 приведен алгоритм отрисовки объекта класса TElement

Таблица 3.6 – Поля класса TElement

|  |  |  |
| --- | --- | --- |
| **Идентификатор поля** | **Назначение** | **Тип поля** |
| Унаследованные поля от класса TControlObject (смотри таблицу 3.2) | | |
| FShape | Форма текущего блока (Эллипс, прямоугольник и др.) | TShapeType |
| FBrush | Кисть, которой отрисовывается фон блока | TBrush |
| FPen | Ручка, которой отрисовываются линии блока | TPen |
| FCaption | Текст, отображаемый на блоке | String |
| FFont | Шрифт текста | TFont |
| FTextFormat | Положение текста | TTextFormat |

Таблица 3.7 – Основные методы класса TElement

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Идентификатор метода** | **Назначение** | **Заголовок метода** | **Идентификатор параметра** | **Назначение параметра** |
| Draw | Отрисовка текста | procedure Draw; | - | - |
| IsInside | Выдает истину, если точка c координатами X, Y находиться внутри блока, иначе ложь | function IsInside(const X, Y: Integer): Boolean; | X | Положение точки по горизонтали |
| Y | Положение точки по вертикали |

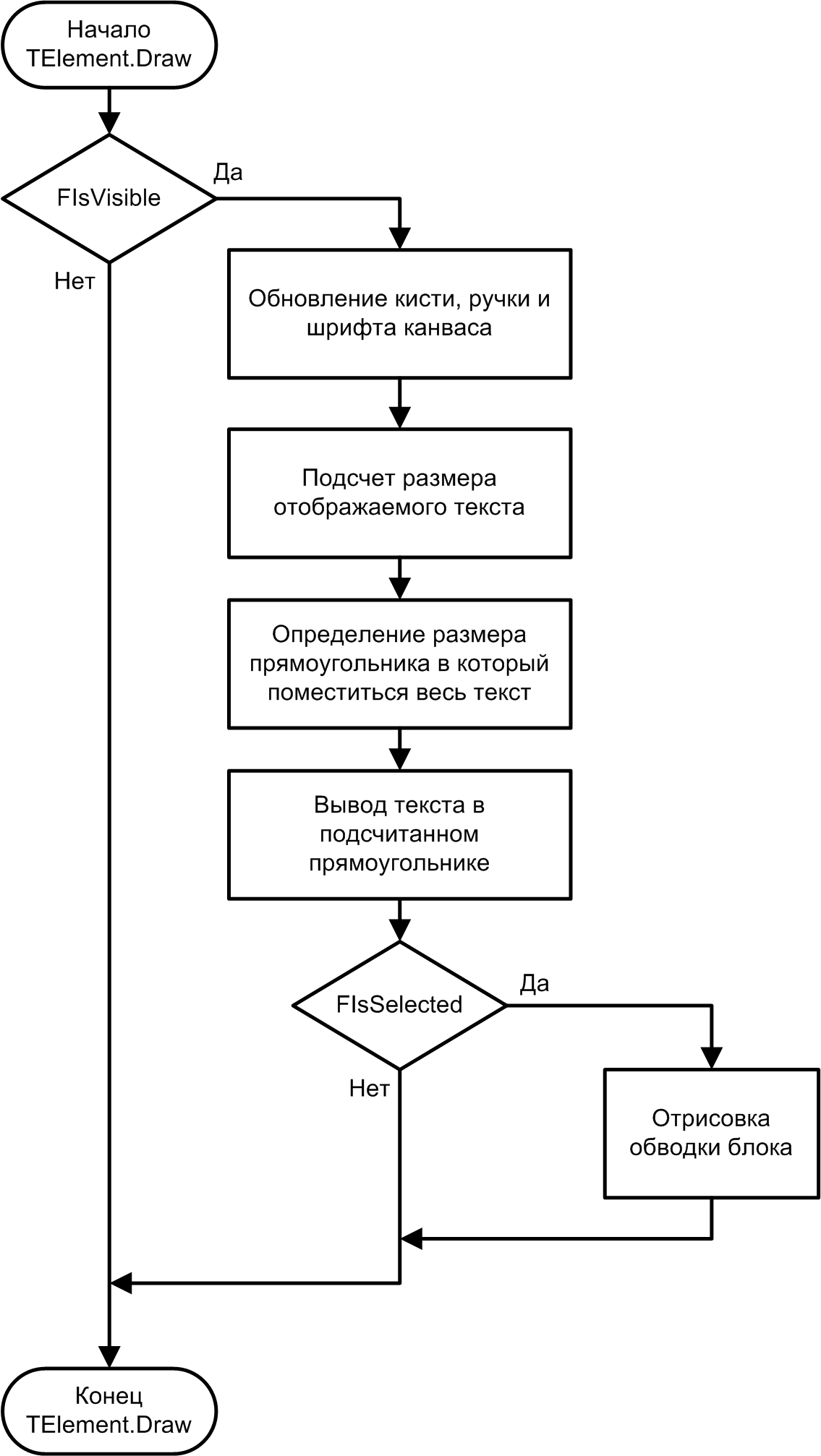


Рисунок 3.2 – Блок-схема отрисовки объекта класса TElement

### **Описание класса TLine**

В таблице 3.8 приведены поля класса TLine

В таблице 3.9 приведены основные методы класса TLine

Таблица 3.8 – Поля класса TLine

|  |  |  |
| --- | --- | --- |
| **Идентификатор поля** | **Назначение** | **Тип поля** |
| FText | Текст, отображаемый на линии | String |
| FIsSelected | Флаг, означающий выбрана ли линия | Boolean |
| FPen | Ручка, которой отрисовывается линия | TPen |
| FCanvas | Объект класса TCanvas. Нужен для отрисовки блока | TCanvas |
| FStart | Позиция начала линии | TConnector |
| FFinish | Позиция конца линии | TConnector |

Таблица 3.9 – Основные методы класса TLine

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Идентификатор метода** | **Назначение** | **Заголовок метода** | **Идентификатор параметра** | **Назначение параметра** |
| Draw | Отрисовка текста | procedure Draw; | - | - |
| IsInside | Выдает истину, если точка c координатами X, Y находиться внутри блока, иначе ложь | function IsInside(const X, Y: Integer): Boolean; | X | Положение точки по горизонтали |
| Y | Положение точки по вертикали |

### **Описание структуры TConnecor**

В таблице 3.10 приведены поля структуры TConnector

Таблица 3.10 – Поля структуры TConnector

|  |  |  |
| --- | --- | --- |
| **Идентификатор поля** | **Назначение** | **Тип поля** |
| BindToElement | Флаг, означающий привязан ли конец линии к блоку | Boolean |
| Element | Если конец линии привязан к блоку, то задает объект класса TElement, к которому привязан | TElement |
| Pos | Если конец линии не привязан к блоку, то задает точку | TPoint |

# **СОЗДАНИЕ (КОНСТРУИРОВАНИЕ) ПРОГРАММНОГО СРЕДСТВА**

Разработка программы выполнялась на основе спецификации функциональных требований.

В программном средстве используются три формы:

1. Главная форма
2. Форма изменения размеров документа
3. Форма окна «About»

## **Разработка главной формы программного средства**

В таблице 4.1 представлены основные поля главной формы.

В таблице 4.2 представлены основные методы, вызываемые при обработке событий.

В таблице 4.3 представлены основные методы главной формы

Таблица 4.1 – Основные поля главной формы

|  |  |  |
| --- | --- | --- |
| **Идентификатор поля** | **Назначение** | **Тип поля** |
| ImageWidth | Ширина документа | Integer |
| ImageHeigth | Высота документа | Integer |
| TextTmp | Временная переменная объекта класса TText. Текст, с которым взаимодействует пользователь | TText |
| ElementTmp | Временная переменная объекта класса TElement. Элемент, с которым взаимодействует пользователь | TElement |
| LineTmp | Временная переменная объекта класса TLine | TLine |
| ConnectorTmp | Временная переменная указатель на структуру TConnector. Указывает на тип соединения конца линии, с которой взаимодействует пользователь | ^TConnector |
| SelectionState | Флаг, указывающий состояние выделения блока, с которым пользователь ведет взаимодействие | Boolean |
| CurrentTool | Текущий выбранный инструмент | TTools |
| StartPoint | Точка начала нажатия | TPoint |
| Elements | Динамический массив элементов | TVector<TElement> |
| Texts | Динамический массив текстов | TVector<TText> |
| Lines | Динамический массив линий | TVector<TLine> |

Таблица 4.2 – Основные методы-обработчики событий

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Описание** | **Заголовок подпрограммы** | **Имя параметра** | **Назначение параметра** |
| WorkspaceMouseDown | Реакция программы на нажатие клавиши мыши над документом | procedure WorkspaceMouseDown(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer); | Sender | Объект, который сгенерировал событие |
| Button | Параметр, определяющий какая кнопка мыши была нажата |
| Shift | Параметр определяющий нажатия управляющих клавиш |
| X | Положение курсора по горизонтали |
| Y | Положение курсора по вертикали |
| WorkspaceMouseMove | Реакция программы на движение мыши над документом | procedure WorkspaceMouseMove(Sender: TObject; Shift: TShiftState; X, Y: Integer); | Sender | Объект, который сгенерировал событие |
| Shift | Параметр определяющий нажатия управляющих клавиш |
| X | Положение курсора по горизонтали |
| Y | Положение курсора по вертикали |
| WorkspaceMouseUp | Реакция программы на отпуск клавиши мыши над документоом | procedure WorkspaceMouseUp(Sender: TObject; Button: TMouseButton; Shift: TShiftState; X, Y: Integer); | Sender | Объект, который сгенерировал событие |
| Button | Параметр, определяющий какая кнопка мыши была нажата |
| Shift | Параметр определяющий нажатия управляющих клавиш |
| X | Положение курсора по горизонтали |
| Y | Положение курсора по вертикали |
| FormCreate | Реакция программы на создание формы | procedure FormCreate(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| toolButtonClick | Реакция программы на нажатие на инструмент | procedure toolButtonClick(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| ImageResolutionExecute | Реакция программы на нажатие пункта меню «Image-Resolution» | procedure ImageResolutionExecute(Sender: TObject); | Sender | Объект, который сгенерировал событие |

Таблица 4.3 – Основные методы формы

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Идентификатор метода** | **Назначение** | **Заголовок метода** | **Идентификатор параметра** | **Назначение параметра** |
| IsClickedElements | Возвращает элемент, на который было произведено нажатие, если такого нет, то возвращает nil | function IsClickedElements(const X, Y: Integer): TElement; | X | Положение курсора по горизонтали |
| Y | Положение курсора по вертикали |
| IsClickedLines | Возвращает линию, на которую было произведено нажатие, если такого нет, то возвращает nil | function IsClickedLines(const X, Y: Integer): TLine; | X | Положение курсора по горизонтали |
| Y | Положение курсора по вертикали |
| IsClickedTexts | Возвращает текст, на который было произведено нажатие, если такого нет, то возвращает nil | function IsClickedTexts(const X, Y: Integer): TText; | X | Положение курсора по горизонтали |
| Y | Положение курсора по вертикали |
| ReDraw | Процедура перерисовки документа | procedure ReDraw; | - | - |
| ClearWorkspace | Процедура очистки документа | procedure ClearWorkspace; | - | - |
| ShowPanel | Процедура, обновляющая значения боковой панели с параметрами выделенных элементов | procedure ShowPanel; | - | - |
| UpdateResolution | Процедура, обновляющая размер документа | procedure UpdateResolution; | - | - |
| SelectAll | Процедура выделения всех объектов рабочей области | procedure SelectAll; | - | - |
| DeselectAll | Процедура снятия выделения со всех объектов рабочей области | procedure DeselectAll; | - | - |

## **Разработка модального окна изменения параметров документа**

В таблице 4.4 приведены основные методы, вызываемые при обработке событий

В таблице 4.5 приведены основные методы формы модального окна

Таблица 4.4 – Основные методы-обработчики событий модального окна

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Имя подпрограммы** | **Описание** | **Заголовок подпрограммы** | **Имя параметра** | **Назначение параметра** |
| btnCancelClick | Реакция программы на нажатие клавиши мыши над документом | procedure btnOkClick(Sender: TObject); | Sender | Объект, который сгенерировал событие |
| btnCancelClick | Реакция программы на движение мыши над документом | procedure btnCancelClick(Sender: TObject); | Sender | Объект, который сгенерировал событие |

Таблица 4.5 – Основные методы формы модального окна

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Идентификатор метода** | **Назначение** | **Заголовок метода** | **Идентификатор параметра** | **Назначение параметра** |
| ChangeResolution | Возвращает результат выполнения модального окна | function ChangeResolution(const AWidth, AHeight: Integer): TModalResult; | AWidth | Текущее значение ширины документа |
| AHeigth | Текущее значение высоты документа |

# **ТЕСТИРОВАНИЕ ПРОВЕРКА РАБОТОСПОСОБНОСТИ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ**

Функциональные тесты, проведенные над программой представлены в таблице 5.1

Таблица 5.1 – Результаты функционального тестирования

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Номер теста** | **Тестируемая функциональность** | **Последовательность действий** | **Ожидаемый результат** | **Полученный результат** |
| 1 | Создание блока с формой прямоугольник | 1. Выбрать инструмент на панели 2. Нажать на свободную область документа | В месте нажатия, появляется блок с формой прямоугольника | Тест пройден |
| 2 | Изменение текста блока | 1. Выделить блок, нажав правой кнопкой мыши по нему инструментом мышь 2. Изменить текст на панели справа | В центре выделенного блока появляется печатаемый текст | Тест пройден |
| 3 | Изменение цвета фона, ширины линии и цвета линии | 1. Выделить блок, нажав правой кнопкой мыши по нему инструментом мышь 2. Изменить необходимые параметры на панели справа | Все параметры изменяются у выделенного блока в соответствии с выбранными значениями | Тест пройден |
| 4 | Изменение шрифта у блока | 1. Выделить блок, нажав правой кнопкой мыши по нему инструментом мышь 2. Вызвать диалоговое окно справа на панели | Шрифт у выделенных блоков изменился в соответствии с выбранным | Тест пройден |
| 5 | Привязка линии к блокам | 1. Создать еще блок, с любой формой 2. Взять инструмент линия 3. Зажать левую кнопку мыши на первом блоке 4. Протянуть курсор до второго блока и отпустить | Между блоками образовалась линия связи. | Тест пройден |
| 6 | Проверка на привязку линии при передвижении блока | 1. Выбрать инструмент мышь 2. Перемещать блок, к которому привязана линия | Линия передвигается вместе с блоком | Тест пройден |
| 7 | Проверка на правильность сохранения данных в специфичный файл | 1. Выбрать пункт меню «File – Save as» 2. Ввести имя файла для сохранения 3. Закрыть программу 4. Открыть программу 5. Открыть файл | Вся проделанная нами работа ранее сохранена и ничего не было потеряно | Тест пройден |

# **РУКОВОДСТВО ПО УСТАНОВКЕ И ИСПОЛЬЗОВАНИЮ**

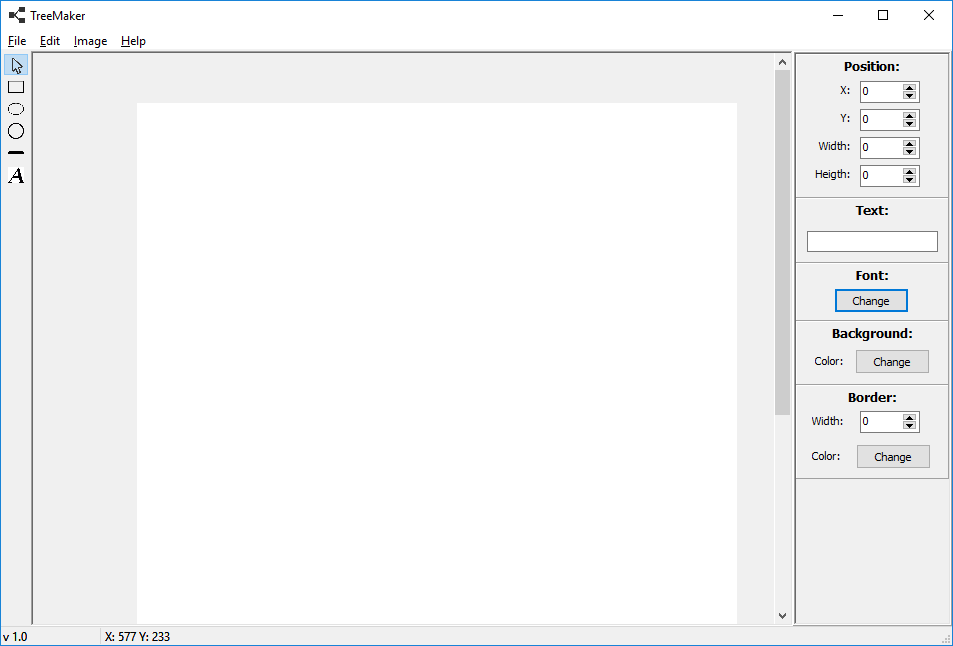
 Для того, чтобы начать использовать программное средство, необходимо запустить файл TreeMaker.exe. После открытия программы появится окно показанное на рисунке 6.1

Рисунок 6.1 – Главное окно программы

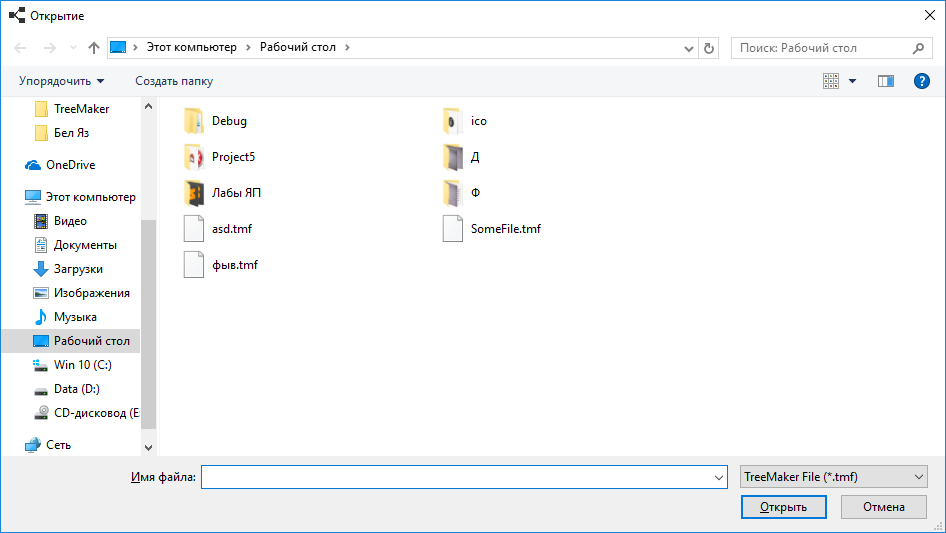
Для открытия файла с расширением .tmf пользователю необходимо выбрать пункт меню «File - Open» или же нажать сочетание клавиш Ctrl + O, после чего откроется диалоговое окно, выбора файла. (рис 6.2)

Рисунок 6.2 – Диалоговое окно открытия файла

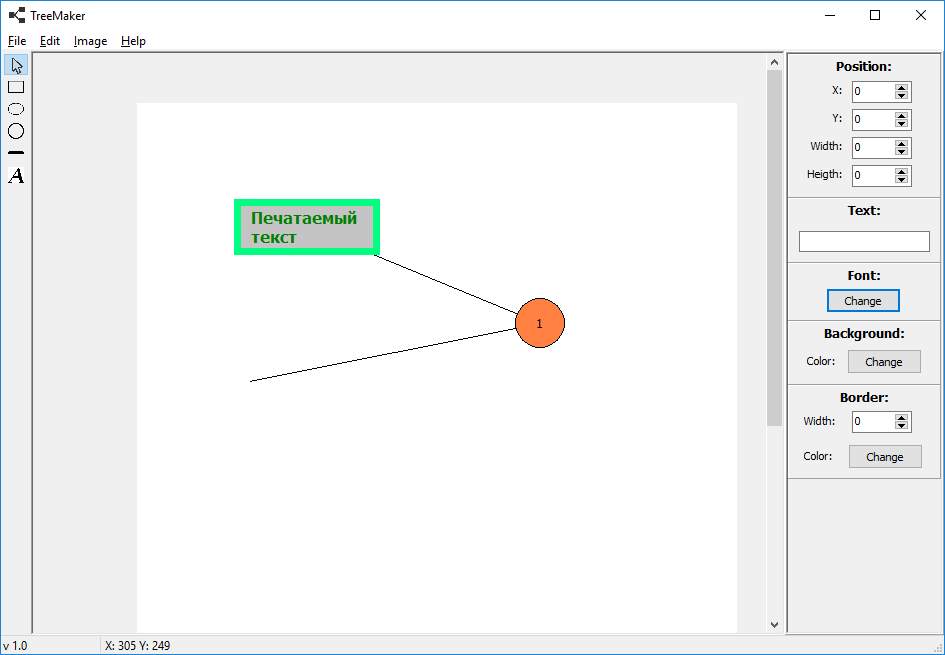
После выбора файла, у нас отобразится его содержимое в рабочей области (рис 6.3).

Рисунок 6.3 – Главное окно программы после открытия файла

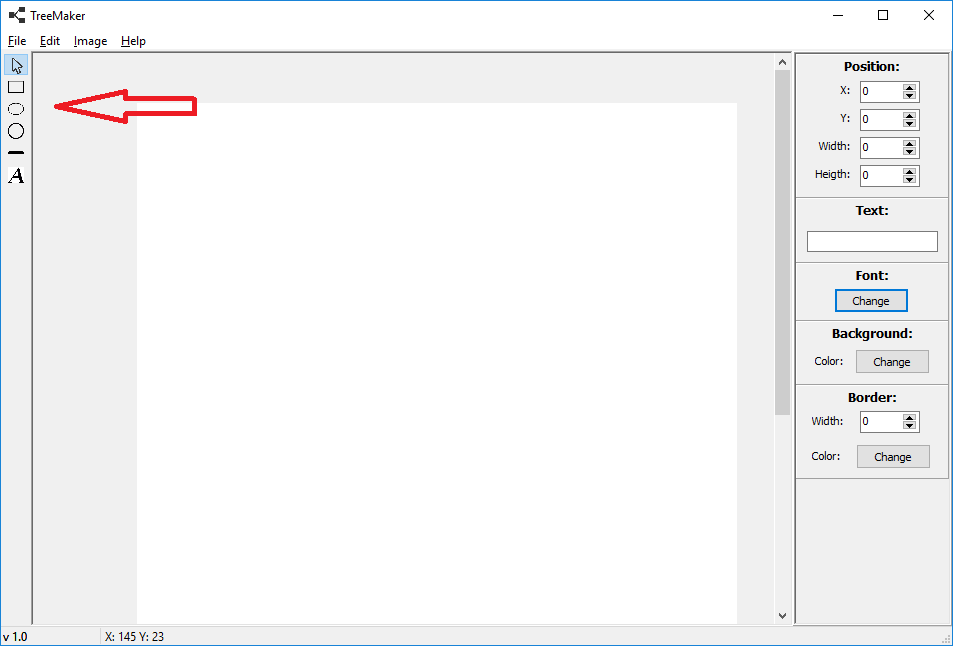
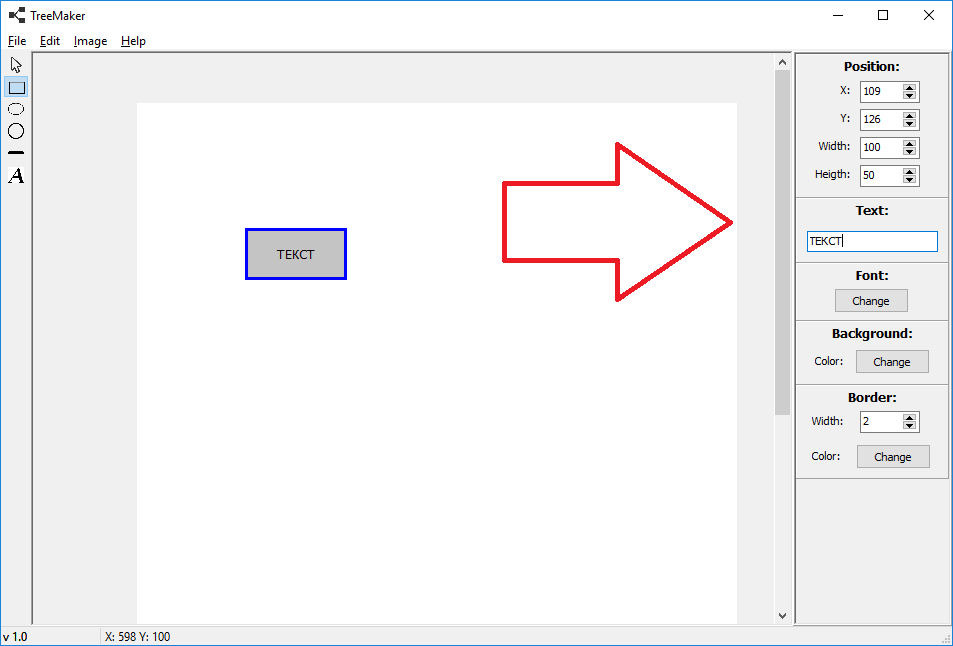
Слева находиться панель инструментов (рис 6.4)

Рисунок 6.4 – Панель инструментов

Справа находиться все свойства блоков и линий (рис 6.5)

Рисунок 6.5 – Панель свойств

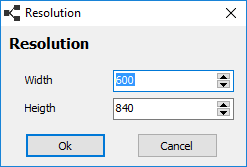
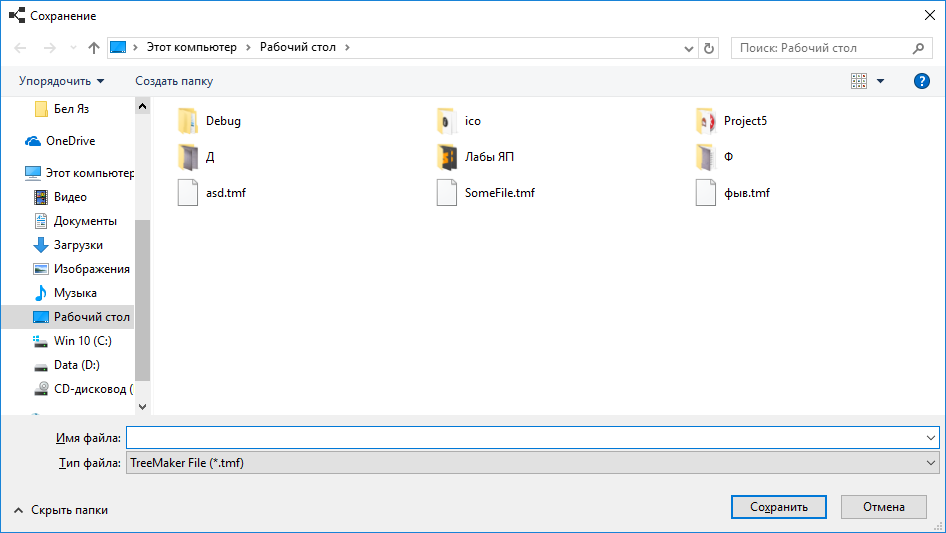
Для изменения размера документа нужно выбрать пункт меню Image – Resolution, после чего откроется модальное окно с выбором размеров документа (рис 6.6).

Рисунок 6.6 – Окно изменения размеров документа

После того, как мы поработали, надо сохранить результат. Для этого следует перейти по пункту меню File – Save as, в результате чего откроется окно с выбором расширения и имени сохраняемого файла (рис 6.7).

Рисунок 6.7 – Окно сохранения файла

# **ЗАКЛЮЧЕНИЕ**

В результате работы над курсовым проектом было создано исправно работающее программное средство для построения и отображения древовидных структур и сете, которое может пригодиться людям, чья жизнь связана с информатикой.

Разработка приложения включала в себя решение множества задач и проблем, как итог было изучено большое количество приложений для построения различных схем, проведен их анализ, и сформированы оптимальные требования для приложений подобного рода.

Далее были изучены некоторые возможности создания приложений в Delphi и формирование конкретных функциональных требований к программе на основе возможностей языка

Затем были разработаны структуры данных, разработана примерная архитектура приложения. Далее были детализированы все функции.

Программа была отлажен и протестирована сначала разработчиком, а затем несколько раз обычным пользователем. После испытаний были внесены корректировки в интерфейсе, работе некоторых функций.

Пройдя все вышеперечисленные этапы на выходе получилось корректно работающее программное средство для построения и отображения древовидных структур и сетей.

# **СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**

1. Древовидная структура [Электронный ресурс] – Режим доступа: <https://ru.wikipedia.org/wiki/Древовидная_структура>
2. Microsoft Visio [Электронный ресурс] – Режим доступа: <https://ru.wikipedia.org/wiki/Microsoft_Visio>
3. Microsoft Visio [Электронный ресурс] – Режим доступа: <http://visio.microsoft.com/>
4. EDraw [Электронный ресурс] – Режим доступа: <https://www.edrawsoft.com/>
5. GDE GoVisual Diagram Editor[Электронный ресурс] – Режим доступа: <https://download.cnet.com/GDE-GoVisual-Diagram-Editor/3000-2075_4-10305236.html>
6. Основы Delphi [Электронный ресурс] – Режим доступа: <http://www.delphibasics.ru/>
7. Embarcadero/IDERA Product Documentation [Электронный ресурс] – Режим доступа: <http://docs.embarcadero.com/>
8. Delphi Russian Knowledge Base [Электронный ресурс] – Режим доступа: <https://drkb.ru/>
9. ГОСТ 19.701–90 (ИСО 5807–85) [Текст]. – Единая система программной документации: Сб. ГОСТов. - М.: Стандартинформ, 2005 с.

**ПРИЛОЖЕНИЕ 1**

**СХЕМА ТВОЮ Ж МАТь**

**ПРИЛОЖЕНИЕ 2**

**Текст главного программного модуля**

**ПРИЛОЖЕНИЕ 3**

# **ВЕДОМОСТЬ**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Обозначение | | | | Наименование | | | | Дополнительные сведения | | | |
|  | | | | Текстовые документы | | | |  | | | |
|  | | | |  | | | |  | | | |
| БГУИР КР 1–40 01 01 НОМЕР ПЗ | | | | Пояснительная записка | | | | 32 с. | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | | Графические документы | | | |  | | | |
|  | | | |  | | | |  | | | |
| ГУИР ГРУППА НОМЕР ПД | | | | См. название графического материала в листе задания | | | | **Формат А1** | | | |
|  | | | |  | | | |  | | | |
| ГУИР ГРУППА НОМЕР ПД | | | | См. название графического материала в листе задания | | | | **Формат А1** | | | |
|  | | | |  | | | |  | | | |
| ГУИР ГРУППА НОМЕР ПД | | | | См. название графического материала в листе задания | | | | **Формат А1** | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  | | | |  | | | |  | | | |
|  |  |  |  |  | **БГУИР КР 1-40 01 01 19 Д1** | | | | | | |
|  |  |  |  |  |
|  |  |  |  |  |
| Изм. | Л. | № докум. | Подп. | Дата | Программное средство построения и отображения сетевых и древовидных структур  Ведомость курсовой  работы |  | | | | Лист | Листов |
| Разраб. | | Северин К.М. |  | 02.06.20 | Т |  | |  | 32 | 32 |
| Пров. | | Фадеева Е.Е. |  | 02.06.20 | Кафедра ПОИТ  гр. 951001 | | | | | |
|  | |  |  |  |
|  | |  |  |  |
|  | |  |  |  |