**СОДЕРЖАНИЕ**

[1 Анализ литературных источников 2](#_Toc131278513)

[1.1 Анализ существующих аналогов 2](#_Toc131278514)

[1.2 Анализ средств разработки программного средства 6](#_Toc131278515)

[1.2.1 Класс 6](#_Toc131278516)

[1.2.2 Динамический массив 6](#_Toc131278517)

[1.2.3 Дерево 6](#_Toc131278518)

[1.2.4 Стек.](#_Toc131278519)6

[1.2.5 Файл 7](#_Toc131278520)

[2 Постановка задачи 9](#_Toc131278521)

[2.1 Описание предметной области 9](#_Toc131278522)

[2.2 Информационная база проекта 9](#_Toc131278523)

[2.3 Функциональное назначение программного средства 10](#_Toc131278524)

# Анализ литературных источников

## Анализ существующих аналогов

**Structurizer** – это бесплатный графический редактор для создания схем Насси-Шнейдермана. Программа была разработана компанией H.J. Schulz & Co. и предлагает ряд функций, которые делают ее привлекательной для пользователей, занимающихся программированием и проектированием.

Среди основных функций Structurizer можно выделить:

* создание блоков и условных переходов, а также циклов;
* возможность настройки цвета, шрифта и размера элементов диаграммы;
* поддержка импорта и экспорта диаграмм в различных форматах, включая PNG, GIF, JPEG и другие.
* возможность использования дополнительных символов и иконок.

Достоинств Structurizer:

* простота и удобство использования программы;
* бесплатность программы;
* возможность импорта и экспорта диаграмм в различных форматах;
* наличие дополнительных символов и иконок.

Недостатки Structurizer:

* ограниченный функционал, необходимый только для создания схем Насси-Шнейдермана;
* не всегда стабильная работа программы;
* yстаревшее ПО. Structurizer не обновляется уже много лет и не поддерживается разработчиками.

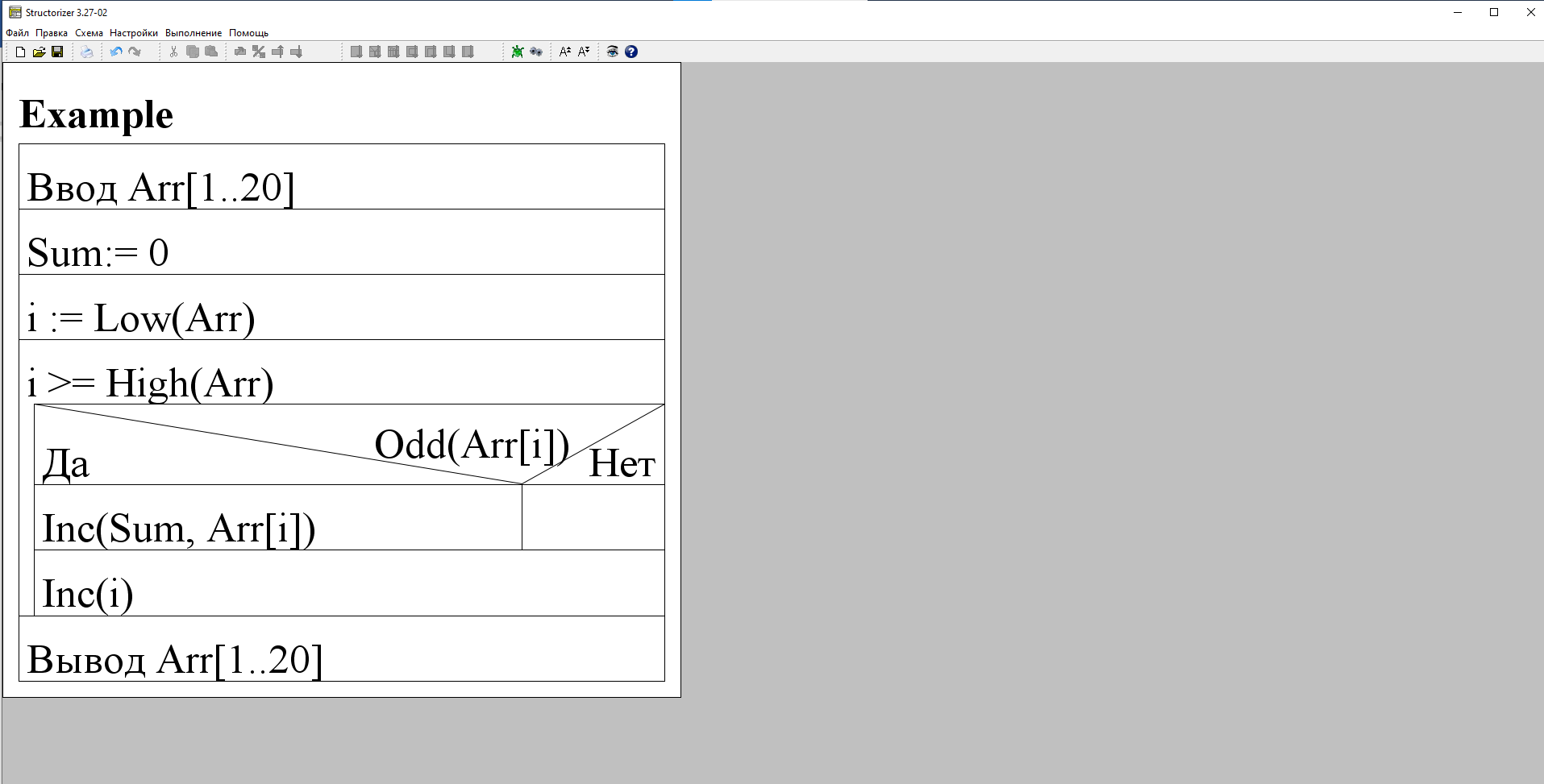


Рисунок 1.1 – Программное средство Structorizer

Structurizer – это простой и удобный инструмент для создания схем Насси-Шнейдермана. Программа имеет достаточно широкий набор функций для решения задач данного типа, и при этом является бесплатной и простой в использовании. Однако, несмотря на эти достоинства, программа иногда может работать нестабильно, что может привести к неудобствам при работе с ней.

**Edrawmax** – это мощный графический редактор, который предоставляет пользователю возможность создавать широкий спектр диаграмм и схем, включая схемы Насси-Шнейдермана. Редактор создан компанией Edrawsoft и имеет ряд функций, которые делают его популярным среди пользователей, занимающихся программированием и проектированием.

Основные функции Edrawmax:

* создание блоков и условных переходов, а также циклов;
* поддержка различных типов соединений и рисунков;
* возможность редактирования цвета, размера, формы и других свойств элементов диаграммы;
* поддержка импорта и экспорта диаграмм в различных форматах, включая PNG, GIF, JPEG, SVG, PDF и другие;
* возможность использования дополнительных символов, шаблонов и шрифтов;
* встроенный набор готовых шаблонов и элементов для быстрого создания диаграмм.

Достоинства EdrawMax:

* мощный и многофункциональный редактор, позволяющий создавать широкий спектр диаграмм;
* большой выбор готовых шаблонов и элементов;
* возможность импорта и экспорта диаграмм в различных форматах;
* наличие дополнительных символов, шаблонов и шрифтов.

К сожалению, Edrawmax не является оптимальным инструментом для построения диаграмм Насси-Шнейдермана. Несмотря на то, что программа имеет некоторые функции, которые могут быть полезны при создании таких диаграмм, у нее есть ряд ограничений и недостатков, которые могут сделать этот процесс менее удобным и эффективным:

* неудобный интерфейс для создания диаграмм Насси-Шнейдермана. Edrawmax не имеет специализированных инструментов для создания диаграмм Насси-Шнейдермана;
* ограниченный функционал для создания диаграмм Насси-Шнейдермана. Edrawmax не имеет специализированных функций для создания диаграмм Насси-Шнейдермана.

Также стоит отметить, что это коммерческое программное обеспечение и для получения полного функционала и доступа к расширенным возможностям, пользователи должны приобрести платную версию программы.

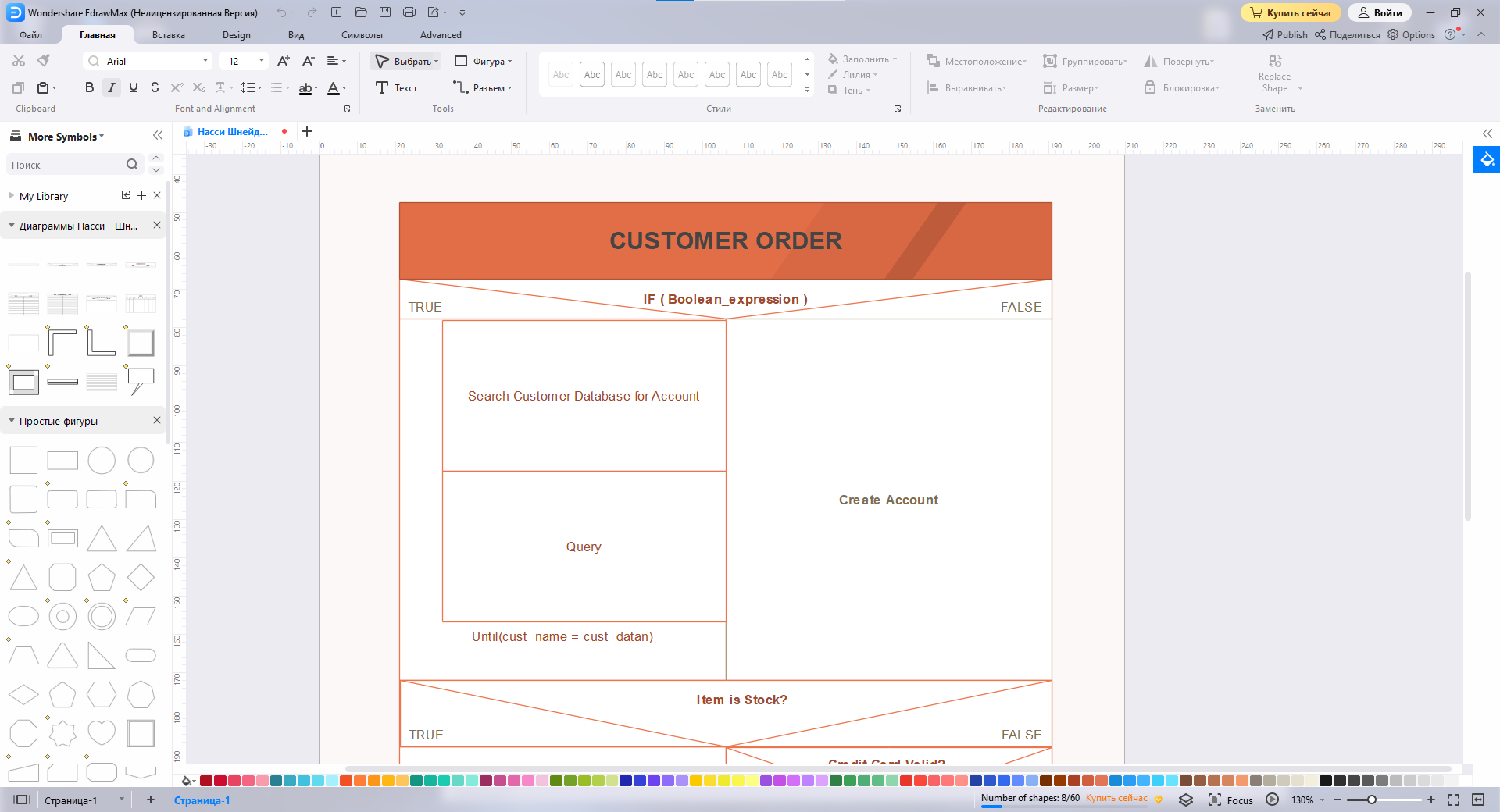


Рисунок 1.2 – Программное средство EdrawMax

В целом, хотя Edrawmax предлагает широкий набор инструментов для создания различных типов диаграмм, программа не является оптимальным выбором для создания диаграмм Насси-Шнейдермана.

**Smartdraw** не только представлен в виде десктопного приложения, но и доступен в виде онлайн-версии, что обеспечивает более гибкую работу с программой и возможность доступа к проектам из любой точки с доступом в интернет. Кроме того, онлайн-версия Smartdraw позволяет работать в реальном времени с другими пользователями, обмениваться комментариями и совместно редактировать документы, что делает программу удобной для работы в коллективе.

Среди основных функций SmartDraw можно выделить:

* редактор предоставляет широкий набор инструментов для создания профессиональных диаграмм и схем;
* интеграция с другими приложениями, такими как Microsoft Word, Excel и PowerPoint, а также с Google Workspace, Jira и другими инструментами;
* предоставляет широкий выбор шаблонов для различных типов диаграмм и схем, что упрощает и ускоряет процесс создания;
* предоставляет возможность онлайн-совместной работы, что упрощает совместное использование диаграмм и схем с другими пользователями;
* возможность импорта и экспорта в различных форматах

Достоинства SmartDraw:

* обширный набор функций, необходимых для создания схем Насси-Шнейдермана;
* возможность настройки цвета, шрифта и размера элементов диаграммы;
* поддержка импорта и экспорта диаграмм в различных форматах;
* легкий интерфейс, понятный даже для новичков;
* возможность использования дополнительных символов и иконок;

Недостатки SmartDraw:

* платное программное обеспечение, необходимо приобретать лицензию для получения доступа к полному функционалу;
* отсутствие возможности редактирования схем в реальном времени, только локальное сохранение и загрузка диаграмм.

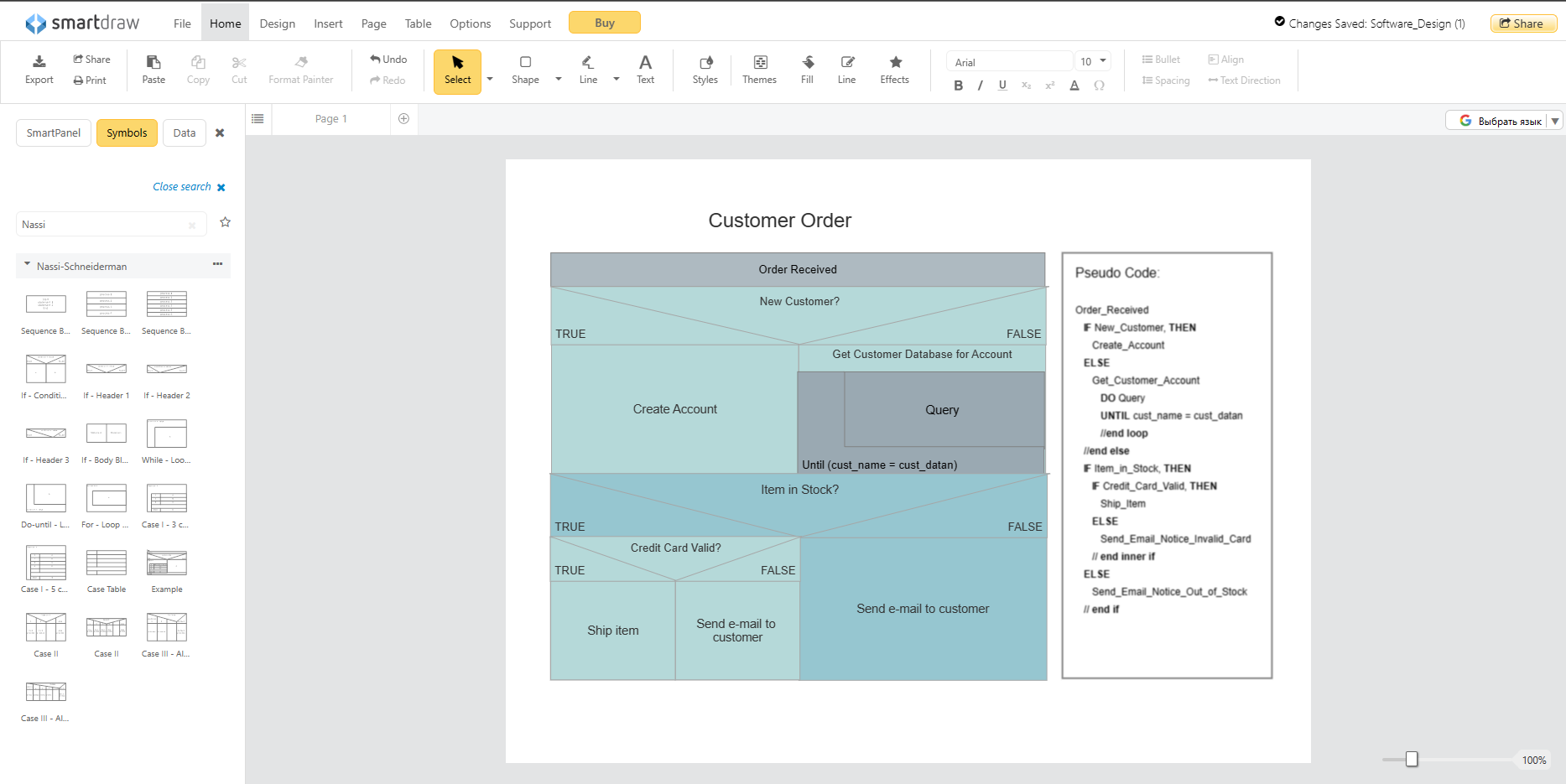


Рисунок 1.3 – Программное средство EdrawMax

Smartdraw – это многофункциональный графический редактор, который может использоваться для создания разнообразных диаграмм и схем, включая схемы Насси-Шнейдермана. Однако, программа не является специализированным инструментом для создания схем Насси-Шнейдермана, поэтому ее функциональность в этой области может быть ограничена.

## Анализ средств разработки программного средства

Для создания программного средства будут использоваться следующие структуры данных:

* класс;
* динамический массив;
* стек;
* Двусвязанный список
* файл.

### Класс

Класс – это шаблон для создания объектов, объединяющий свойства и методы для работы с этими объектами. Классы позволяют абстрагироваться от конкретных реализаций и описывать сущности, свойства и возможные действия над ними. Основными элементами класса являются поля (свойства) и методы. Поля класса описывают его состояние, а методы – действия, которые могут быть выполнены с объектами этого класса. Классы также позволяют наследоваться от других классов и реализовывать интерфейсы.

Классы в программном средстве представляют собой иерархию блоков, каждый из которых имеет определенное назначение и может содержать другие блоки. Каждый класс представляет отдельный тип блока, который может быть добавлен на схему. Классы также могут содержать другие блоки внутри себя.

То есть классы в данной программе используются для создания иерархии блоков, которые могут быть расположены на схеме и связаны между собой. Пользователь может выбирать нужный класс блока и добавлять его на схему, затем настраивать его свойства. Таким образом, классы обеспечивают гибкость и удобство создания схемы по методу Насси-Шнейдермана.

### Динамический массив

Динамический массив – это структура данных, которая представляет собой массив, размер которого может быть изменен во время выполнения программы. Для реализации динамического массива обычно используется выделение памяти в куче. Когда массив заполнен, выделяется новый блок памяти большего размера, данные из старого массива копируются в новый, и старый массив освобождается. Динамический массив позволяет эффективно управлять памятью и обрабатывать большие объемы данных.

В программном средстве для построения схем по методу Насси-Шнейдермана, динамический массив используется для хранения всех блоков, которые были созданы пользователем. Этот массив обеспечивает быстрый доступ к любому элементу благодаря своей структуре, что позволяет легко добавлять, удалять и изменять блоки.

Бинарный поиск используется для поиска определенного блока в массиве. Когда пользователь выбирает блок для изменения или удаления, программа использует бинарный поиск для быстрого поиска соответствующего блока в массиве. Это позволяет существенно ускорить процесс поиска, особенно когда массив содержит много элементов.

Сочетание динамического массива и бинарного поиска обеспечивает высокую производительность программного средства, позволяя пользователю быстро создавать и изменять схемы по методу Насси-Шнейдермана

### Стек

Стек – это структура данных, которая работает по принципу «последним пришел, первым ушел». В стек можно добавлять элементы только на вершину, а удалять – только верхний элемент. Стек используется в программировании для реализации операции отката, когда нужно отменить последнее действие. Для этого в программе можно использовать стек для хранения истории действий пользователя. Каждое действие представляется как элемент стека, и при нажатии на кнопку "Отменить" из стека извлекается последнее действие и выполняется обратное действие, чтобы отменить его эффект. Таким образом, стек позволяет эффективно реализовывать операцию отмены действий в программе.

В программном средстве для построения схем по методу Насси-Шнейдермана используется стек для поддержки отмены (undo) и повтора (redo) действий пользователя при создании и редактировании схемы. Каждый раз, когда пользователь выполняет какое-либо действие, такое как добавление блока или изменение связей между блоками, состояние схемы сохраняется в стеке. Если пользователь захочет отменить последнее действие, программа извлечет состояние схемы из стека и вернет ее в предыдущее состояние.

Кроме того, стек используется для создания новых условий в блоке case. При создании нового условия в блоке case, программа помещает его в вершину стека. Если пользователь захочет удалить условие, программа просто извлечет его из стека.

Для реализации стека в программе используется список, который позволяет легко добавлять новые элементы и удалять уже существующие.

### Двусвязанный список

Двусвязанный список - это структура данных, которая состоит из узлов, каждый из которых содержит ссылки на предыдущий и следующий узел в списке. Таким образом, двусвязный список позволяет двигаться вперед и назад по списку, начиная с любого узла.

Каждый узел в двусвязном списке содержит данные и ссылки на предыдущий и следующий узлы. Первый узел в списке называется головным, а последний - хвостовым.

Одним из преимуществ двусвязного списка является то, что он позволяет эффективно вставлять и удалять элементы в середине списка, в отличие от массива, где это может быть дорогостоящей операцией. Однако, двусвязный список имеет более высокую памятьовую стоимость, чем массив, так как каждый узел должен хранить две ссылки на другие узлы.

### Файл

Файлы могут быть разделены на две основные категории: логические и физические файлы. Логический файл – это файл, который представляет собой логически связанные данные, имеющие определенную структуру. Он может содержать различные типы данных, такие как текстовые документы, изображения, аудио- и видеофайлы, базы данных и другие. Логический файл определяет формат, структуру и ограничения для данных, которые он содержит. Физический файл, с другой стороны, это непосредственно файл, хранящийся на жестком диске или другом устройстве хранения информации. Физический файл содержит набор битов, которые могут быть интерпретированы как данные, которые он представляет.

Логические и физические файлы тесно связаны друг с другом. Логический файл описывает формат и структуру данных, которые он содержит, а физический файл представляет собой место хранения этих данных на устройстве. Когда данные записываются в логический файл, они сохраняются в соответствующем физическом файле, который затем может быть прочитан для получения этих данных. При чтении данных из логического файла, система оперирует на физическом файле, считывая данные из определенного участка жесткого диска, и затем интерпретирует их в соответствии с форматом логического файла.

Существует три типа файлов: типизированные файлы, текстовые файлы и не типизированные файлы. Типизированные файлы связываются с файловыми переменными, объявленными как "file of <Тип>". Файл считается состоящим из элементов, каждый из которых имеет тип <Тип>. Не типизированные файлы могут быть связаны только с файловыми переменными, которые были объявлены как "file". Файл считается состоящим из элементов, размер которых определяется при открытии файла. Текстовый файл представляет собой последовательность символов, которая может быть разделена на строки. Строки могут быть различной длины (в том числе пустые). В конце каждой строки помещается специальный управляющий символ: возврат каретки (#13 или М международное обозначение CR) и перехода новую строку (#10 или международное обозначение LF). С наличием этого маркера связана логическая функция Eoln (End of line). Эта функция возвращает значение True, если текущая позиция в файле находится в конце строки (т.е. перед символом перехода на новую строку). Текстовые файлы могут быть открыты в одном из двух режимов: для чтения или для записи. Когда файл открывается для чтения, указатель позиции устанавливается на начало файла. Когда файл открывается для записи, содержимое файла удаляется, а указатель позиции устанавливается на начало файла.

Кроме того, текстовые файлы могут быть открыты в режиме добавления, который позволяет добавлять данные в конец файла без удаления его содержимого. Файл, открытый в режиме добавления, всегда открывается для записи, но указатель позиции устанавливается на конец файла.

Важно отметить, что для работы с файлами в программировании нужно уметь открывать и закрывать файлы, читать и записывать данные в файлы, а также обрабатывать ошибки, связанные с файлами, такие как отсутствие файла или ошибка доступа. Также нужно следить за использованием ресурсов компьютера при работе с файлами, чтобы избежать проблем с памятью или производительностью.

Можно сделать вывод о том, что файлы являются важными элементами программирования, которые позволяют хранить и загружать данные в различных форматах. Функции чтения и записи данных в файлы, а также функции управления файлами позволяют программистам создавать приложения, которые сохраняют пользовательские настройки и данные, что делает их более удобными и гибкими в использовании. Типизированные файлы позволяют сохранять данные в определенном формате, что упрощает работу с ними в дальнейшем. Кроме того, использование файлов позволяет пользователям сохранять свою работу и продолжать работать с ней в будущем, что является важным элементом при создании любых приложений.

## Спецификация функциональных требований

Программное обеспечение обеспечивает пользователю возможность создавать и редактировать схемы Насси-Шнейдермана.

Функциональные требования – это определенные задачи и функции, которые программа должна выполнять для обеспечения желаемого функционала. Функциональные требования определяются на основе бизнес- и пользовательских требований.

Разрабатываемое программное обеспечение обеспечивает создание приложения, которое позволяет пользователю создавать схемы по методу Дамке и сохранять их в нужном формате. Бизнес-требования представляют собой общее видение, не включающее детализации поведения системы и технических характеристик.

Функциональные требования к разрабатываемому ПС приведены в таблице 1.

Таблица 1 – Функциональные требования к программному средству

|  |  |
| --- | --- |
| Идентификатор | Требование |
| ФТ-1 | Создавание схем по методу Насси-Шнейдермана из каталога блоков |
| ФТ-2 | Сохранение истории изменение схемы и отката к предыдущим версиям |
| ФТ-3 | Редактирование свойств блоков |
| ФТ-4 | Отображение схемы |
| ФТ-5 | Создание и загрузка файлов схем |
| ФТ-6 | Экспорт в различные форматы |
| ФТ-7 | Масштабирование схемы |
| ФТ-8 | Управление свойствами проекта |

Также программное также обеспечение должно иметь интуитивно понятный и простой интерфейс, обеспечивающий удобство работы с ним. Это может быть достигнуто с помощью использования понятных иконок и кнопок, простой навигации, интуитивно понятного меню.